타일드-디스플레이 시스템에서 실시간 동영상 상영기의 설계 및 구현

(Design and Implementation of a Realtime Video Player on Tiled-Display System)

최 기 석 † 유 정 수 † 최 정 훈 † 낭 종 호 ††
(Giseok Choe) (Jeongsoo Yu) (Jeonghooni Choi) (Jongho Nang)

요 약 본 논문에서는 높은 해상도를 제공하기 위하여 여러 대의 PC와 모니터를 연결한 타일드-디스플레이(Tiled-Display) 시스템 상에서 동작하는 실시간 동영상 상영기를 설계 및 구현하였다. 제안한 동영상 상영기는 하나의 동영상을 기가 비트(Giga bit) 폐쇄 이더넷으로 연결된 여러 PC에 UDP 멀티캐스트를 사용하여 전송하고 각 수신기는 받은 동영상 데이타의 압축을 푼 후 이미지를 자신의 영역에 분할하여 시간적인 동기화를 맞추어서 재생할 수 있도록 설계되었다. 본 시스템은 미디어 데이타의 전송 중 발생하는 패킷 손실 및 지터(jitter) 문제를 동영상의 비트레이트에 따라서 방송량을 결정하는 흐름 제어 방법과 필요한 만큼 미리 받은 뒤 재생을 시작하는 버퍼링 방법을 통하여 해결하였으며, 서로 다른 PC의 상영기 간의 동기화를 위하여 별도의 오버헤드 없이 시작 시간만 동기화 하고 각 PC의 상영기들의 리퍼런스 클럭의 속도를 동일하도록 하여 안정적인 실시간 스트리밍 및 상영이 가능하도록 하였다. 또한 여러 전송 포맷 및 압축 포맷을 지원하기 위하여 Microsoft DirectShow 구조상에서 구현되었다.

키워드 : 디스플레이, 실시간 동영상 상영기, DirectShow

Abstract This paper presents a design and implementation of realtime video player that operates on a tiled-display system consisting of multiple PCs to provide a very large and high resolution display. In the proposed system, the master process transmits a compressed video stream to multiple PCs using UDP multicast. All slaves(PC) receive the same video stream, decompress, clip their designated areas from the decompressed video frame, and display it to their displays while being synchronized with each other. A simple synchronization mechanism based on the H/W clock of each slave is proposed to avoid the skew between the tiles of the display, and a flow-control mechanism based on the bit-rate of the video stream and a pre-buffering scheme are proposed to prevent the jitter. The proposed system is implemented with Microsoft DirectX filter technology in order to decouple the video/audio codec from the player.

Key words : Tiled-Display, Realtime Video Player, DirectShow

· 이 논문은 2007 한국컴퓨터종합학술대회에서 '메타데이타를 비휘발성 램에 유지하는 플래시 파일시스템에서 가비지 컬렉션 수행에 대한 효율성 분석'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 서강대학교 컴퓨터학과
brix@sogang.ac.kr
yjs@mlneptune.sogang.ac.kr
drumist@mlneptune.sogang.ac.kr
†† 종신회원 : 서강대학교 컴퓨터학과 교수
jhnang@sogang.ac.kr

논문접수 : 2007년 9월 27일
심사완료 : 2008년 1월 23일

정보과학회논문지 : 시스템 및 이론 제35권 제4호(2008.4)

1. 서 론

타일드-디스플레이(Tiled-Display)는 다수의 디스플레이 디바이스를 그리드형태로 연결하여 높은 해상도를 제공함으로써 VR 기술을 이용한 가상공간에서의 공동 협업 환경 등에 사용된다. 이런 타일드-디스플레이 환경에서 원격지에 있는 설계자에 대한 화상 회의용 동영상을 실시간으로 타일드-디스플레이 상에 상영할 필요가 있으며, 또한 공동 작업을 위하여 지역 저장 장치에 저장된 참고 동영상을 실시간으로 타일드-디스플레이 상에 상영하여야 한다.

본 논문에서는 원격지의 화상 회의용 동영상 및 저장

되어 있는 일반 동영상을 타일드-디스플레이상에서 실시간으로 상영할 수 있는 상영기 구조를 설계하였으며, 실제 6개의 PC로 이루어진 타일드-디스플레이 상에서 시스템을 구축하였다. 타일드-디스플레이상에서 동영상을 재생하기 위해서는 동영상소스를 전송하는 송신기와 네트웍 및 전송된 동영상을 자신이 맡은 디스플레이 영역에 재생하는 상영기가 필요하다. 이런 시스템을 구축하기 위하여 해결하여야 하는 문제 중 하나는 동영상 화면을 여러 PC에 분할하여 시간적인 동기화를 맞추어서 상영하는 것이다. 우선 동영상을 분할하는 방법으로는 비디오 데이타의 압축을 푼 후 타일드-디스플레이상의 각 영역에 맞게 이미지를 나누어서 해당 PC의 상영기에 전송하는 방법이 있는데, 이 경우 압축되지 않은 데이타의 전송으로 인한 네트웍 자원 소비가 문제가 되며, 각각의 나누어진 이미지를 다시 압축하여 보내는 방법을 사용하더라도 송신기 측의 압축 작업으로 인한 시스템 부하라는 문제가 남는다. 따라서 본 논문에서는 동영상의 압축된 데이타를 그대로 방송하고 상영기가 디코딩 후 자신의 영역만 클리핑하여 디스플레이하는 방법을 제안한다. 이러한 방법은 네트웍 부하와 송신측 PC의 시스템 부하가 적고 모든 상영기에 동일한 스트림을 전송하기 때문에 UDP 멀티캐스트를 사용하여 효과적인 방송을 할 수 있다는 장점이 있다. 또한, 이러한 방법으로 분할된 동영상을 동기화 하여 재생하는 방법으로는 어느 한 PC가 기준이 되어 나머지 PC에게 동기화 정보를 주기적으로 전송하는 방법이 있는데, 이러한 방법은 동기화 정보 전송에 대한 오버헤드 및 각 PC마다의 전송 지연시간의 차이로 인한 오차가 문제가 된다. 본 논문에서 제안한 동기화 방법은 각 상영기의 재생 시작 시간을 동기화한 후 각 상영기의 리퍼런스 클럭의 속도가 동일하도록 하드웨어클럭을 사용하고 이에 맞추어서 동영상 프레임들을 상영하도록 하는 것이다. 추가적으로, 동영상 방송 시 각 PC의 상영기에서 발생하는 언더플로우 및 오버플로우를 방지하기 위하여 압축된 동영상의 비트레이트에 맞추어서 방송하는 방법을 설계하고 구현하였다. 본 시스템은 여러 압축 형식 및 전송 형식에 대한 종속성을 탈피하기 위하여 Microsoft DirectShow[1] 필터 구조를 이용하여 구현되었으며, 실제로 6대의 PC로 이루어진 타일드-디스플레이상에서 실험하여 수신율 및 동기화 상태를 측정하였다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 타일드-디스플레이 상영기의 동기화 방법에 대하여 살펴본 후, 3장에서는 본 시스템의 흐름 제어 방법과 동기화 방법 및 시스템의 구조에 대하여 기술한다. 4장에서는 본 시스템의 수신율과 동기화에 대하여 실험을 통하여 측정한 후 이를 분석하였고, 마지막으로 5장에서는 결론을 맺는다.

2. 기존의 타일드-디스플레이 상에서의 동기화 방법

타일드-디스플레이 상에서 동영상을 상영하기 위하여 필요한 기술은 동영상을 기가 비트로 연결된 PC들에게 실시간으로 방송 및 수신하는 기술과 동기를 맞추어서 상영하는 기술로 나눌 수 있다. 이런 기술들에 대하여 기존의 시스템에서 개발된 방법들을 정리하면 표 1과 같다. 표 1에서 나타낸 것과 같이 대부분의 시스템에서는 비디오(혹은 그래픽 오브젝트)를 방송하는 통신망과 동기화 정보를 보내는 통신망을 따로 가지고 있으며, 기가 비트 이더넷상에서의 실시간 방송은 대부분 UDP 프로토콜의 방송 기능을 이용하여 구현하였다. 반면에, 동기를 위한 정보는 신뢰도 있는 TCP 프로토콜을 이용하여 구현하였으며, 동기화 정보 전송의 낮은 지연 시간 및 신뢰도를 위하여 패킷의 우선 순위를 높여서 주기적으로 전송을 한다. 그러나 이와 같이 주기적으로 전송되는 정보를 기준으로 하는 동기화 방법은 각 PC간의 전송에 대한 지연시간의 차이로 인한 오차가 발생될 가능

표 1 기존 시스템의 실시간 방송 및 동기화 방법들

	실시간 동영상 방송 및 수신 방법	PC들 사이의 동기화 방법
SAGE (Scalable Adaptive Graphics Environment) by U. Of Illinois at Chicago, 2006 [2]	-UDP 상에서 각 비디오 프레임들의 번호를 표시하는 프로토콜 사용 -Video Stream : Gbps interface -Sync Message : FastEthernet interface	-TCP channel작은 지연시간과 높은 우선순위의 TCP 사용, 별도의 동기화를 위한 쓰레드 사용
TeraVision at U. Of Illinois by Chicago, 2004 [3]		-TCP 상에서 Two way handshake 방식 사용 -TCP 소켓의 TCP_NO_DELAY 옵션을 사용하여 지연시간을 최소화
AG (Access Grid) Media Architecture by GIS [4]	-UMTP (UDP Multicast Tunneling protocol)을 사용	-TCP를 이용하여 자체 구현
S/W Environments for Cluster-based Display System by Princeton Univ. [4]	-Myrinet에서의 graphic primitive 방송	-System-level sync. (SSE)와 Application-level sync (ASE) 제안 100 Mbps Ethernet상에서의 Winsock을 이용한 구현(no detailed description)

성이 있으며 동기화 정보 전송에 대한 오버헤드가 발생한다. 본 연구에서는 별도의 주기적인 동기화 정보 전송을 사용하지 않는 방식으로 기가 비트 폐쇄 이더넷상에서 실시간 스트리밍을 통해 동영상을 재생한다.

3. 실시간 동영상 상영기의 설계

본 연구에서는 동영상의 압축 포맷 및 전송 포맷에 독립적인 시스템을 구축하기 위해서 Microsoft의 DirectShow 구조를 이용하여 설계 및 구현하였으며 전체 시스템 구조는 그림 1과 같다. 본 장에서는 제안한 시스템의 동영상 분할 재생 방법과 흐름 제어 방법, 동기화 방법, DirectShow를 이용한 구현에 대하여 설명한다.

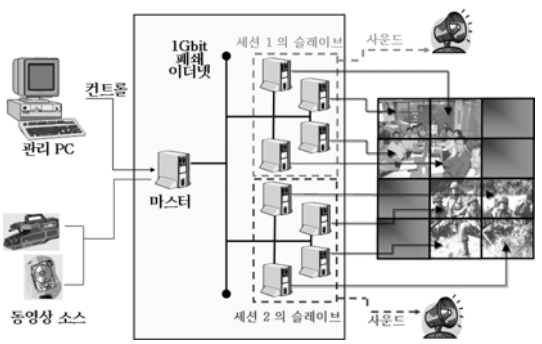


그림 1 전체 시스템 구조

3.1 동영상 분할 상영 방법

타일드-디스플레이상에 하나의 동영상을 여러 부분으로 분할하여 재생하는 방법으로 크게 세 가지가 가능하다. 첫 번째는 송신기가 비디오 데이타의 압축을 푼 후 타일드-디스플레이상의 각 영역에 맞게 이미지를 나누고 각 상영기의 디스플레이상에 재생될 좌표 및 비율을 계산하여 나누어진 비디오 데이타와 함께 해당 PC의 상영기에 전송하는 방법이다. 이러한 방법은 상영기에서 디코딩할 필요가 없으므로 상영기의 구조가 간단하다는 장점이 있지만 압축되지 않은 데이타를 전송해야 하므로 네트워 부하가 심하다. 예를 들어, 1280×720 크기의 30fps 비디오를 4개의 디스플레이 영역에서 재생할 경우 약 5.3Mbps×4의 전송이 필요하다. 두 번째 방법은 압축을 풀어서 이미지를 나눈 후 이를 다시 압축하여 각 상영기에 전송하는 방법으로, 이 경우는 송신기에서 여러개로 나누어진 이미지에 대한 압축 작업으로 인하여 시스템 부하가 문제가 되므로 실시간 전송에 적합하지 않다. 세 번째 방법은 송신기가 압축된 동영상 데이타를 그대로 방송하고 각 PC의 상영기에게 자신이 디스플레이할 이미지 좌표 정보를 전송하여 상영기가 동영상 데이타의 압축을 푼 후 이미지 좌표 정보에 맞게 이미지를 클리핑하여 디스플레이하는 방법이다. 이 방법은 압축을 푼 이미지에서 클리핑된 부분의 나머지를 사용하지 않기 때문에 실제로 사용되지 않는 연산을 한다는 단점이 있지만, 네트워과 송신기 PC의 부하가 적고 모든 상영기에게 동일한 스트림을 전송하므로 UDP 멀티캐스트를 사용하여 효과적인 방송을 할 수 있다. 특히 이미지를 송신기에서 분할할 경우 각 PC에 유니캐스트 방식으로 방송해야 하며 송신기는 각각의 유니캐스트 전송에 대하여 흐름 제어를 따로 해야 한다. 이에 비하여 멀티캐스트 방식은 하나의 흐름 제어로 방송하는 것이 가능하다. 따라서 본 논문에서는 세 번째 방법의 분할 재생 방법을 사용하였으며 이러한 방법을 통해 네트워 및 시스템 자원 절약과 각 상영기에 대한 개별적인 제어의 최소화라는 이점을 얻을 수 있다.

3.2 동기화 방법

서로 다른 PC에서 하나의 동영상을 재생하기 위해 동기화를 맞추는 방법으로는 동기화 정보를 주기적으로 전송하는 방법[2-5]이 있다. 그러나 그러한 방식은 별도의 동기화 정보를 전송하는데 대한 오버헤드를 발생시키며, 특히 동기화 정보 전송의 딜레이차이로 인하여 정확한 동기화를 맞추기 어렵다.

본 논문에서는 기가 비트 폐쇄 이더넷 상에서 각 PC의 상영기들의 동기화를 유지하며 실시간 스트리밍을 구현하기 위해 별도의 동기화 정보를 전송 하지 않으며, 이에 대한 설명은 다음과 같다. 여러 PC에서 동시에 재생하는 각 상영기들의 동기화는 스트림 타임(stream time)과 타임스탬프, 리퍼런스 클럭을 통해서 이루어진다. 타임스탬프는 미디어 데이타가 재생될 시간으로써 전송전에 계산된 후 미디어 데이타와 함께 페킷화 되어 전송되며, 리퍼런스 클럭은 한 상영기내에서 서로 다른 미디어간의 동기화를 위하여 시간 축으로 참조되는 클럭이고, 스트림 타임은 각 미디어의 논리적 재생 시간으로써 재생이 시작될 때 0으로 세팅되고 그 이후에는 리퍼런스 클럭의 증가량에 대한 값을 가진다. 비디오의 재생은 스트림 타임과 동영상 프레임의 타임스탬프의 비교를 통해서 스케줄링 되며, 마스터로부터 전송되는 타임스탬프는 모든 상영기에서 동일하다. 따라서 각 상영기를 동기화하기 위해서는 스트림 타임을 동일하도록 하여야 하며, 이를 위해서는 각 상영기에서의 재생 시작 시간과 리퍼런스 클럭의 속도를 동일하게 해주어야 한다. 리퍼런스 클럭의 속도를 동일하게 하기 위해서 각 PC의 상영기들은 하드웨어 클럭(RTC - Real Time Clock)을 리퍼런스 클럭으로 사용하도록 한다. 물론 RTC 역시 각 PC마다 속도가 차이날 수 있지만, 일반적으로 그 차이가 수십 일 동안에 수 초정도이므로 각 PC의 상영기들의 재생 동기화를 위한 싱크 허용치(±

120 밀리세컨드[7])보다 크지 않다. 그리고 각 PC의 상영기들의 재생 시작 시간을 맞추기 위해서 송신측 PC의 리퍼런스 클럭을 글로벌 클럭으로 사용하였다. 재생 전 각 송신기는 글로벌 클럭과 각 상영기의 로컬 클럭을 Cristian's Algorithm[9] 방식에 의하여 동기화 한다. 송신기와 모든 상영기간의 클럭 동기화를 맞춘 후 방송을 시작하기 전 각 상영기가 재생을 시작해야 할 시간에 대한 글로벌 클럭의 값을 각 상영기에게 알려준다. 이와 같은 방식으로 각 상영기는 스트림 타임을 동일하게 유지함으로써 동기화를 맞추게 된다. 이러한 방법은 주기적으로 동기화 정보를 전송하는 방법에 비하여 오버헤드가 적으며 시작점에서만 동기화를 수행하므로 오차가 발생할 확률이 적다.

3.3 흐름 제어 방법

미디어 데이타 전송 시 수신 버퍼의 언더플로우 및 오버플로우를 방지하기 위해서는 전송 중 흐름 제어를 통해 데이타량을 조절하여야 한다. 이러한 흐름 제어 방법으로는 전송기가 상영기의 버퍼량을 모니터링 하면서 전송률을 조절하는 방법[8]이 있는데, 모니터링을 위한 프로토콜 구축 및 비트레이트 조절이 가능한 인코더 또는 압축 포맷의 지원이 필요하기 때문에 본 연구에서는 사용하지 않는다. 또한, 미디어 데이타의 전송방법을 크게 특정 시점에 전송률이 집중되도록 보내는 버스트(burst) 모드와 항상 일정한 전송률로 보내는 일정 속도 모드로 나눌 수 있다[6]. 버스트 모드는 송신 어플리케이션의 CPU 활용 면에서는 적합하지만 버스트 시점에서 네트워상에서의 충돌로 인한 지터(jitter) 및 버퍼 오버플로우등을 발생시키는 단점이 있다. 반면에 일정 속도 모드는 네트웍에 주는 부하가 적지만 전송률을 일정하게 유지하기 위해서는 CPU를 지속적으로 독점해야 한다는 단점이 있다.

본 연구의 시스템에서는 미디어 데이타를 전송하는 PC에서 전송기와 상영기가 함께 동작할 수 있으며 타일드-디스플레이 시스템상의 다른 응용프로그램도 같이 동작할 수 있으므로, CPU사용량이 많은 일정 속도 모드의 전송 방식은 사용 하지 않고 대신 압축된 미디어 데이타의 저장 단위인 미디어 샘플(media sample)[1]의 타임스탬프에 맞추어서 버스트 모드로 전송한다. 그리고 버스트 모드 전송 시 발생할 수 있는 버스트 구간에서의 소켓 버퍼 오버플로우 현상을 해결하기 위하여 최대 버스트 크기를 기준으로 소켓 버퍼를 세팅한다. 이러한 방식의 전송 시 최대 버스트 크기는 최대 미디어 샘플 사이즈와 동일하며, 대부분의 미디어 포맷에서 최대 미디어 샘플 사이즈는 스트림을 끝까지 읽어보기 전에는 알 수 없는 경우가 많으므로 *초당 최대 비트레이트 / 평균 프레임 레이트*를 통해 근사값을 사용한다. 그리고 소켓으로부터 데이타를 읽는 일을 담당하는 쓰레드를 두어 소켓 인터페이스에서 발생하는 이벤트에 빠르게 반응할 수 있도록 하여서 디코딩 및 렌더링 작업에 블러킹되지 않고 안정적으로 소켓 버퍼의 데이타를 소비할 수 있도록 한다. 또한 버스트 모드의 전송으로 인한 지터 발생과 미디어 스트림의 비트레이트 변동으로 인한 수신율의 불규칙성은 상영기의 일시적인 언더플로우 및 오버플로우를 발생시킬 수 있다. 이 문제를 해결하기 위해 버퍼를 두어 디코딩 및 렌더링작업이 언더플로우 시 블러킹 되는 현상을 최소화할 수 있도록 한다. 버퍼링 양은 본 시스템상에 적합한 초기 버퍼링시간을 실험을 통하여 구한 후 *초기 버퍼링 시간 × 평균 비트레이트* 만큼의 크기로 초기화 하며, 재생 중 버퍼량을 모니터링하여 크기를 조절한다.

4. DirectShow 구조를 이용한 구현

본 논문에서 구현한 타일드-디스플레이에서 동영상 상영기의 전체 시스템 구조는 그림 2에서 나타낸 바와 같이 마스터(Master), 슬레이브(Slave), 관리 PC로 구성된다. 관리 PC는 인터페이스를 통해 동영상 세션 컨트롤을 본 시스템에서 설계한 컨트롤 프로토콜 형식으로 마스터에게 요청한다. 동영상 세션의 소스로는 캠코더, 동영상파일, 스트리밍 URL, 브로드캐스팅 등이 될 수 있다. 마스터는 관리 PC의 요청에 따라 하나의 동영상 소스에 해당하는 세션을 생성하고, 동영상 소스로부터 스트림을 읽어서 필요시 역 다중화 또는 파일 포맷 파싱 등을 거친 후 오디오와 비디오 스트림을 UDP로 전송할 수 있도록 패킷화 하여 멀티캐스트로 전송한다. 슬레이브는 마스터가 전송하는 스트림을 받아서 필요시 디코딩을 거친 후 이미지를 자신이 담당하는 영역으로 클리핑하거나 사이즈를 조절하여 재생한다. 마스터와 슬레이브는 데몬에 의하여 동적으로 생성되고 여러 전송 포맷 및 압축 포맷을 지원하기 위하여 Microsoft DirectShow 필터 구조로 구성된다. 세부 구조는 그림 2에 나타낸 바와 같으며, 본 논문에서는 멀티캐스트 필터(Multicast Filter)와 리시버 필터(Receiver Filter)를 직접 구현하고 전송/컨트롤/모니터링 프로토콜을 설계하였다.

마스터는 크게 소스 필터(Source Filter)와 스플리터 필터(Splitter Filter), 멀티캐스트 필터, 슬레이브 메니저(Slave Manager)로 구성된다. 소스 필터와 스플리터 필터는 동영상 소스를 읽어서 오디오와 비디오 스트림을 분리하는 역할을 한다. 또한, 본 연구에서 구현한 멀티캐스트 필터는 스플리터 필터로부터 분리된 오디오와 비디오 스트림을 UDP 상에서 멀티캐스트하는데 필요한 프레그멘테이션, 타임 스템프, 시퀀스등의 헤더정보와 함께 본 연구에서 정의한 전송프로토콜인 패킷화된 샘

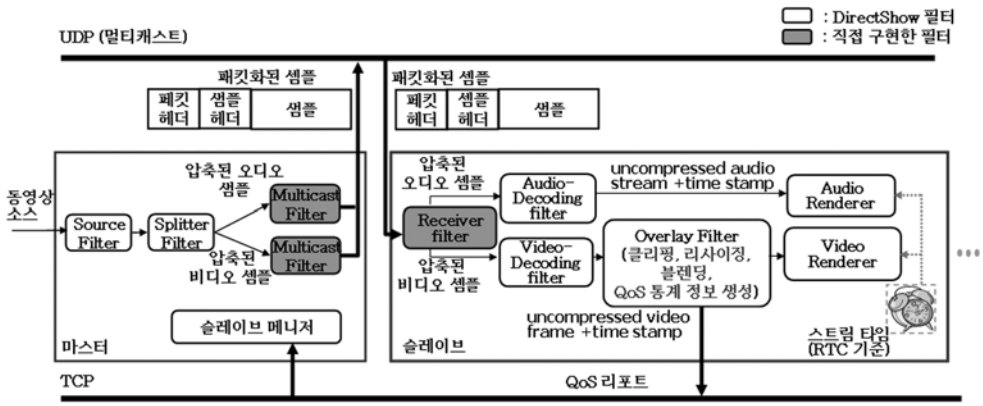


그림 2 마스터 프로세스 및 슬레이브 프로세스의 내부 구조

플 형식에 따라 패킷화하여 전송한다. 멀티캐스트 필터는 본 논문의 3.1장에서 기술한 흐름 제어를 위하여 DirectShow에서 제공하는 기본 렌더러(Base Renderer)필터의 스케줄링기능을 이용한다. 즉, 마스터의 멀티캐스트 필터는 기본 렌더러 필터로부터 상속받아 구현하므로써 입력받은 데이타의 타임 스템프까지 블러킹되는 방식으로 스케줄링 된다. 따라서 미디어 데이타의 타임 스템프에 맞추어서 전송을 하게 되므로 미디어의 비트

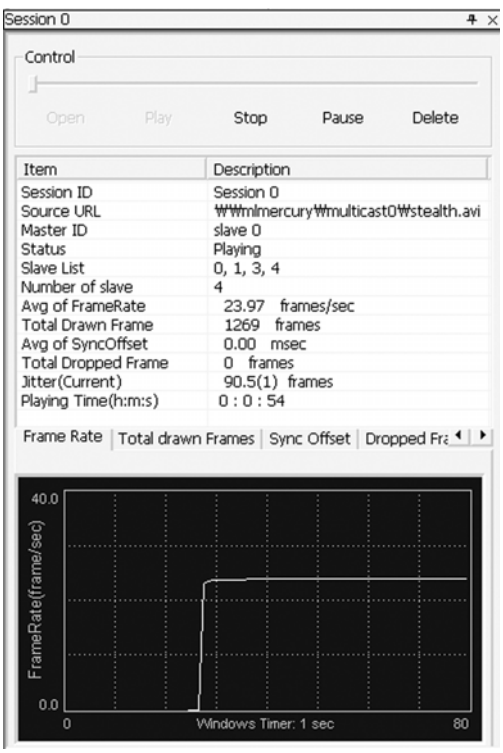


그림 3 UI를 통한 실시간 상영기 성능 모니터링 예

레이트와 동일한 속도로 전송할 수 있게 된다. 슬레이브 메니저는 마스터가 전송할 동영상 세션에 참가하는 슬레이브들에 대한 생성을 해당 데몬에게 요청하고 생성된 슬레이브가 타일드-디스플레이상에서 자신이 맡은 영역을 디스플레이하는데 필요한 이미지 좌표 정보를 계산하여 전송하는 역할을 한다.

슬레이브는 크게 리시버 필터, 디코더 필터(Decoder Filter), 오버레이 필터(Overlay Filter)와 렌더러 필터(Renderer Filter)로 구성된다. 리시버 필터는 버퍼링을 담당하는 쓰레드가 UDP 소켓 인터페이스를 통해 데이타를 읽어서 역 패킷화 한 후 버퍼링을 한다. 디코딩과 렌더링을 담당하는 쓰레드는 리시버 필터로부터 비동기적으로 미디어 데이타를 읽어간다. 각 미디어의 디코딩 및 렌더링을 담당하는 쓰레드들은 렌더러 필터에서 리퍼런스 클럭(reference clock)을 기준으로 하여 미디어 데이타의 타임 스템프까지 블러킹되는 방식으로 스케줄링 된다. DirectShow는 기본적으로 리퍼런스 클럭을 오디오 렌더러 필터가 제공해 준다. 오디오 렌더러 필터가 제공해 주는 클럭은 오디오 스트리밍 속도를 기반으로 생성된다. 이 경우, 한 상영기에서 오디오와 비디오에 대한 동기화에는 적합하다. 그러나 타일드-디스플레이상에서처럼 여러 시스템에서 모든 미디어가 동기화 되기 위해서는 본 논문의 3.2장에서 기술한 바와 같이 각 상영기의 리퍼런스 클럭의 속도가 동일해야 하는데, 오디오 스트리밍 속도는 각 상영기간의 차이가 존재하므로 리퍼런스 클럭의 어긋남이 발생하게 된다. 따라서 오디오 렌더러 필터에서 제공한 클럭보다 더 정확한 RTC를 기준으로 하는 클럭을 사용하도록 설정한다. 그밖에, 디코더 필터는 압축된 데이타의 디코딩 작업을 하며, 오버레이 필터는 마스터로부터 세팅된 이미지 좌표 정보를 참조하여 디스플레이 할 이미지에 대한 클리핑, 리사이징 및 블렌딩을 한다.

관리 PC는 타일드-디스플레이 상에서 동영상의 이미지 배치 및 비율 설정과 각 PC사이에서 마스터와 슬레이브들의 지정을 위한 인터페이스를 GUI를 통해 제공한다. 또한 동영상 세션에 참가하는 모든 슬레이브들은 재생 중 QoS 통계 데이타를 생성하여 마스터에게 주기적으로 보고하며, 마스터는 보고 된 데이타들을 멀티플랙싱 하여 관리 PC로 전송하고, 관리 PC는 이를 그래프 형태로 사용자가 모니터링 할 수 있도록 한다. 그림 3은 관리PC에서 받은 모니터링 정보를 GUI를 통해 디스플레이 하는 예이다.

5. 실험 및 분석

본 논문에서는 구축한 3×2 타일드-디스플레이 시스템에 자체 제작한 UI를 이용해서 표 3과 같은 동영상 데이타들에 대해 크게 두 가지 실험을 수행하였다. 첫 번째 실험은 본 논문의 3.2절에서 서술한 동기화 방법에 대한 실험으로써 재생이 시작될 때 스트림 타임을 0으로 세팅하고 각 상영기의 리퍼런스 클럭의 속도를 동일하게 세팅하는 작업을 진행한 후 상영결과에 대한 각 슬레이브의 모니터링 정보를 비교 및 분석하는 실험을 수행하였으며, 두 번째 실험은 본 논문의 3.1절에서 서술한 흐름제어 방법과 관련해서 버스트모드로 동영상 소스를 전송할 때 슬레이브에 버퍼가 없는 경우와 있는 경우의 모니터링 정보를 비교 및 분석하고 가장 적절한 버퍼링의 크기를 결정하기 위한 실험을 수행하였다.

표 3 동영상 소스의 종류와 정보

	해상도	Codec	재생시간
샘플 1	800 × 448	StandardMPEG	00:12:39
샘플 2	640 × 352	MPEG 2	00:35:45
샘플 3	1280 × 720	XviD MPEG 4	00:45:29
샘플 4	1280 × 720	WMV	00:05:14

5.1 실험환경

본 타일드-디스플레이 상영기 구축에 사용된 컴퓨터는 Windows XP Home Edition이 탑재된 LG전자 XPION(X270) Tower 6대로써 인텔 듀얼코어 3.4GHz CPU, 1GB DDR2 메모리, 160GB IDE-HDD, GeForce 6200 128MB, 기가비트 랜카드로 구성되어 있으며, 모니터는 20.1인치 LG전자 LCD 모니터로, 해상도는 1600*1200으로 설정하였다. 그 외의 기타 장비로는 3COM 1 기가비트 허브(8 port)와 JELEEI 마이크로 사운드 믹서(8 channel) 등이 있으며, DirectX는 9.0C 버전을 사용하였다.

5.2 흐름 제어에 대한 실험

첫 번째 실험은 본 논문의 3.1절에서 서술한 흐름제어 방법과 관련하여 하나의 슬레이브에 샘플 1을 버스트 모드로 전송하면서 버퍼를 추가했을 때와 추가하지 않았을 때의 수신율(receive rate)과 소비율(playout rate) 결과를 비교 및 분석하였다. 버퍼를 추가하지 않았을 때 수신율와 소비율에 대한 그래프는 그림 4와 같으며, 충분한 크기의 버퍼를 추가했을 때의 수신율과 소비율에 대한 그래프는 그림 5와 같다.

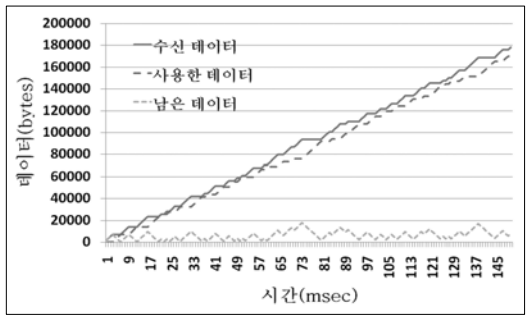


그림 4 버퍼가 없을 때 데이타 수신율 및 소비율

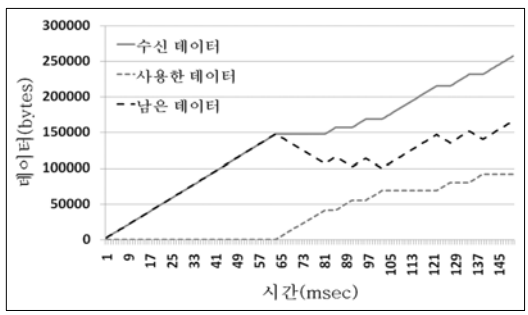
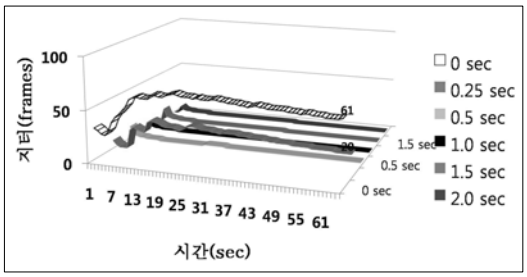


그림 5 버퍼가 있을 때 수신율 및 소비율

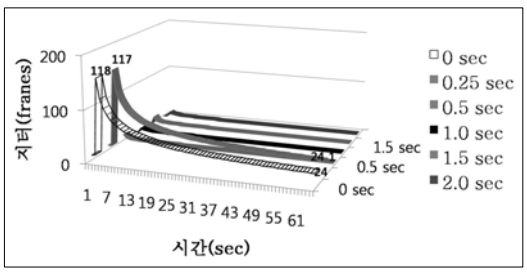
그림 4에서 보는 바와 같이, 버퍼가 없는 경우에는 재생곡선이 수신곡선과 붙는 부분이 자주 보이는데 이는 재생을 담당하는 쓰레드가 가져갈 데이타가 없음으로 인해 블러킹되어 있는 상황을 나타낸다. 반면 그림 5에서처럼 흐름제어를 위한 버퍼를 추가하고 버퍼링을 한 뒤 상영할 경우이며 재생을 담당하는 쓰레드는 블러킹되지 않고 버퍼로부터 자신이 원하는 소비율로 재생할 수 있게 된다. 초기 버퍼링 시간을 설정할 때, 안정적인 동영상 재생과 버퍼링 딜레이 최소화를 동시에 고려하여 적절한 버퍼링 시간을 설정해야 한다.

두 번째 실험은 본 절의 실험에서 사용한 버퍼의 적절한 크기를 유추해내기 위한 실험이다. 각각 다른 초기 버퍼 크기에서 샘플 3, 샘플 4를 테스트 했으며, 재생품질을 측정하기 위한 기준으로 오버레이 필터에서 얻는 플레이 지터를 사용했다.

그림 6을 보면, 버퍼가 없거나 0.25 초 이하의 경우는



(a) 샘플 3



(b) 샘플 4

그림 6 버퍼링 시간에 따른 동영상 재생 품질의 변화

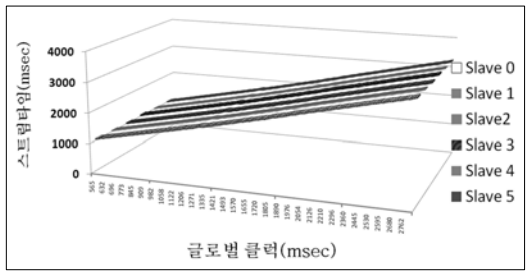


그림 7 각 슬레이브의 스트림타임 변화

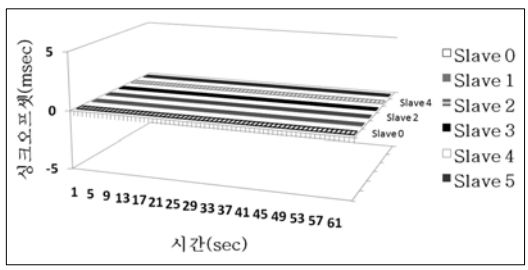


그림 8 각 슬레이브의 싱크오프셋 변화

재생품질이 눈에 띄게 좋지 않다는 것을 알 수 있다. 이 결과는 네트워크 혼잡도가 거의 없는 이상적인 환경에서 나온 것다. 재생 미디어의 특성은 끝까지 전부 재생하기 전까지는 완전히 알 수 없으며, 미래의 네트워크 혼잡도는 정확히 예측할 수 없다. 이러한 예측할 수 없는 상황을 고려하면, 안정적인 재생 품질을 보장하려면 적어도 0.25초를 초과하는 버퍼링이 필요하다는 것을 알 수 있다.

본 논문에서는 안정적인 재생 품질과 사용자가 느끼는 버퍼링 딜레이를 고려하여 1초의 버퍼링 초기값을 선택하였다.

5.3 동기화에 대한 실험

이 실험은 본 논문의 3.2절에서 제안한 동기화 방법에 대하여 표 3에 나와 있는 여러 종류의 동영상 소스에 대해 반복적인 실험을 수행하면서 각 슬레이브의 모니터링 정보를 비교 및 분석함으로써 동기화 여부를 판단하였다. 아래의 실험 결과는 샘플 3을 이용한 실험 결과로써, 그림 7은 각 슬레이브의 글로벌 클럭에 대한 스트림타임의 변화를 측정한 것이다.

그림 7을 통해 3.2절에서 제안한 방법을 이용하여 각 상영기의 동기화를 맞출 경우 각 슬레이브의 스트림타임이 동일하게 변화함을 알 수 있다. 또한 그림 8에서 보는 바와 같이 모든 슬레이브가 동일하게 싱크오프셋의 값이 '0'을 유지하면서 상영하고 있음을 알 수 있다. 이와 같은 결과를 바탕으로 각 상영기는 스트림 타임을 동일하게 유지함으로써 동기화를 맞춘다는 사실을 알 수 있으며, 따라서 본 논문에서 제안한 동기화 방법인, 시작점에서만 동기화를 수행하고 각 상영기는 스트림타임을 동일하게 유지함으로써 동기화를 맞추는 방식이 적절함을 알 수 있다.

그림 7 및 그림 8을 통해 3.2절에서 제안한 방법을 이용하여 각 상영기의 동기화를 맞출 경우 시간의 흐름에 상관없이 일치함을 알 수 있다. 또한 그림 9의 상영결과와 그림 10의 시간에 따른 각 슬레이브의 프레임레이트 변화를 통해서 모든 슬레이브가 안정적으로 상영하고 있음을 확인할 수 있다.

이외에도 표 3의 모든 동영상 소스 및 그 밖의 다른 코덱으로 압축된 동영상 소스에 대해서도 안정적인 상영이 가능하도록 3×2타일드-디스플레이를 구현하였다.

6. 결 론

본 논문에서는 타일드-디스플레이에서 원격지에 있는 설계자에 대한 화상 회의용 동영상 또는 공동 작업 중 사용될 참조 동영상 등을 재생하는데 이용될 수 있는 실시간 동영상 상영기를 구현하였다. 본 논문의 동영상 상영기는 Microsoft DirectShow 구조상에서 개발되었으며, 제안한 흐름 제어 방법과 동기화 방법에 대하여 기가 비트 폐쇄 이더넷 환경에서의 실험을 통하여 확인하였다.

향후에 연구해야 할 과제로는, 타일드-디스플레이에서 상영되는 모든 이미지 및 소리를 동영상 파일로 저장 및 전송이 가능하도록 확장을 할 계획이다.

(a) 한 개의 세션을 상영한 결과 화면

(b) 동시에 다수의 세션을 상영한 결과 화면

그림 9 상영 결과화면

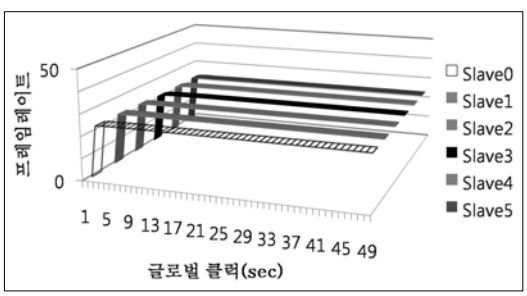


그림 10 각 슬레이브의 프레임레이트 변화

참 고 문 헌

[1] Microsoft MSDN, *DirectShow*, http://msdn2. microsoft.com/en-us/library/ms783323.aspx, 2007.

[2] University Illinois, *Scalable Adaptive Graphics Environment*, http://www.evl.uic.edu/cavern/sage/description.php, 2007.

[3] R. Singh, B. Jeong, L. Renambot, A. Johnson and J. Leigh, "TeraVision: a Distributed, Scalable, High Resolution Graphics Streaming System," *in proceedings of the 2004 IEEE International Conference on Cluster Computing*, pp. 391-400, 2004.

[4] Y. Zhao and X. Zhang, "Design a secure and scalabel CVE: The Access Grid," *in Proceedings of the 2001 ACM/IEEE conference on Supercomputing*, 59-59, 2001.

[5] H. Chen, D. Clark, Z. Liu, G. Wallace and K. Che, "Software Environments For Cluster-Based Display Systems," *in proceedings of the 1st International Symposium on Cluster Computing and the Grid*, pp. 202, 2001.

[6] E. Magaña, J. Aracil and J. Villadangos, "Packet Video Broadcasting with General-Purpose Operating Systems in an Ethernet," *in proceedings of Multimedia Tools and Applications*, pp. 5-28. Sep. 2004.

[7] Dennis R. Allison, David J. Farber, and Bruce D. Shriver, *Multimedia: Computing, Communications and Applicationstation, Compression, Synchronization and Programming.*

[8] ETSI, *TS 126 234 v6.4.0 Release 6 Protocol and codecs*, 2005.

[9] A. Tanenbaum and M. Steen, 2002, *Distributed Systems Principles and Paradigms*, Prentice Hall.

최 기 석

정보과학회논문지 : 시스템 및 이론
제 35 권 제 3 호 참조

유 정 수

2005년 서강대학교 컴퓨터공학과 학사
2007년~서강대학교 컴퓨터공학과 석사 과정. 관심분야는 멀티미디어 시스템, 이미지 검색, 동영상 분석

최 정 훈

2007년 서강대학교 컴퓨터공학과 학사.
2007년~서강대학교 컴퓨터공학과 석사 과정. 관심분야는 멀티미디어 시스템, 이미지 검색, 동영상 분석

낭 종 호

정보과학회논문지 : 시스템 및 이론
제 35 권 제 3 호 참조