# 가상 데스크탑 환경 상 고해상도 원격 화면 전송 기술 연구

## (High-resolution Remote Presentation Research on Virtual Desktop Environment)

성 백 재, 박 찬 익
포항공과대학교 컴퓨터공학과
(Baegjae Sung, Chanik Park)
Department of Computer Science and Engineering, Pohang University of Science and Technology

Abstract : With the help of virtualization and high-bandwidth network, virtual desktop is becoming popular technique. Virtual desktop is launched on server using virtualization technique. Then client connect the virtual desktop over the network using remote desktop protocol. However, the virtual desktop technique has problems when it support high-resolution remote presentation. The problems are the screen delaying and long response time.
Firstly, we propose a prototype of high-performance virtual desktop environment using open-sources. Second, we analyse the screen delaying problem using profiling. In our experimental environment, remote presentation throughput is 12.6~13.1 Frame-per-second when we present Full HD movie. Most overhead of processing time per frame is encode and decode operations (63%~91%). To solve the overhead, we propose encode and decode technique using GPGPU.

Keywords : Virtual Desktop, Remote presentation, High resolution

## Ⅰ. 소 개

최근 가상화 환경과 고속 유무선 네트워크 환경의 대중화로 인해, 서버에 가상 머신을 실행키시고, 원격지에서 데스크탑 프로토콜을 통해 접속하여 사용하는 가상 데스크탑 기술 [1, 2] 이 주목받고 있다. 특히 가상 데스크탑을 언제 어디서나 사용할 수 있다는 장점과 함께, 데스크탑을 중앙에서 관리함으로써 얻을 수 있는 보안, 업데이트 관리 등의 구조적 장점을 가지고 있다.

이러한 장점을 기반으로 대중화가 진행 중이나, 현재 고해상도 화면을 지원함에 있어 사용자의 편의성이 떨어진다는 한계점을 가지고 있다. 예를 들어, HD급 해상도 (1280x720)은 30 Frame-per-second (FPS)를 지원하여 사용자가 응용프로그램을 사용하거나, 동영상 재생 시 끊김없는 화면을 전송받을 수 있으나, Full HD급 해상도 (1920x1080)은 15 FPS를 지원하여 전송화면이 끊기거나 사용자의 조작 응답속도가 떨어지는 문제점을 가진다.

본 논문에서는 고성능 가상 데스크탑을 지원하는 가상화 환경을 제안하고, 성능 프로파일링을 통해 Full HD급 고해상도 원격 화면 전송 시 발생하는 화면 끊김 문제점을 구체화 한다. 이후 해당 문제점을 해결하기 위한 방안을 제시한다.

* 교신저자(Corresponding Author)
성백재, 박찬익 : 포항공과대학교
※ 본 연구는 지식경제부 산업원천기술개발사업의 클라우드 DaaS 시스템 및 단말 기술 개발 및 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행되었음.
(No. 10035242) (No. 2011-0016972)

## Ⅱ. 본 론

### 1. 오픈소스 고성능 가상 데스크탑 환경 프로토타입

기존 서버 가상화에 사용하였던 가상화 환경은 가상 데스크탑 환경에 그대로 적용하기에 고해상도

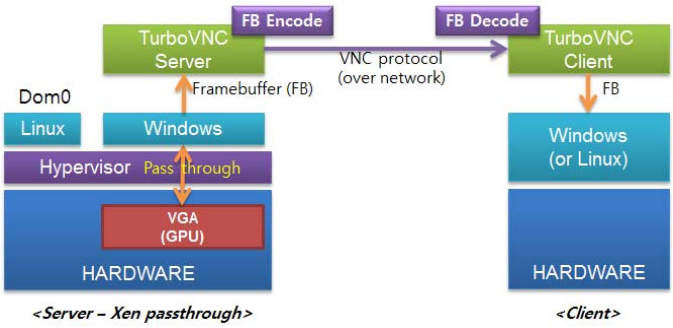


그림 1 오픈소스 고성능 가상 데스크탑 환경 프로토타입

미지원, 3D 미지원 등의 한계점을 가지고 있다. 예를 들어, Xen 의 기본 가상 그래픽 카드 Cirrus Logic은 1280x1024를 최대해상도로 지원하고 있으며 3D 가속을 미지원 한다. 이를 해결하기 위한 방법으로 데스크탑 가상화를 지원하는 업체 [1,2,3] 들의 솔루션을 사용하는 방법이 있으나, license 비용이 발생한다는 한계를 가지고 있다. 따라서 본 장에서는 오픈소스로 구성할 수 있는 고성능 가상 데스크탑 환경 프로토타입을 제안한다.

그림 1은 본 논문에서 실험환경으로 사용된 프로토타입이다. 하이퍼바이저 Xen 4.0.1을 사용하였으며, pass through기능 [4] 을 사용하여 그래픽 카드를 가상머신에 직접 할당하였다. 원격 화면 전송을 위한 기술로 TurboVNC [5] 를 채택하여, 각각 Server, Client 버전을 설치하였다. TurboVNC 는 TurboJpeg 라이브러리 [6] 를 사용하여, 화면 이미지 (Framebuffer) 를 압축(Encode)/복원(decode)하는 시간을 단축시킨 VNC 기술이다.

이후 성능 프로파일링에 사용된 환경은, 소프트웨어는 상위와 동일하며, 하드웨어는 Intel i7 950 (3.07 GHz), 4 GB 메모리, 기가비트 네트워크를 사용하였다.

## 2. 원격 화면 전송 성능 프로파일링

원격 화면 전송 성능을 도식화 하기 위해, 동영상 재생 워크로드를 구성하여 실험하였다. 즉 서버측의 가상 데스크탑은 동영상을 전체화면으로 재생하고, 클라이언트는 해당 데스크탑에서 접속하여 동영상 화면을 전송받는다. 이후 서버 측 또는 클라이언트 측에서 프레임 당 처리속도를 기반으로 FPS를 측정한다. 본 논문에서 목표로 하는 화면 전송 성능은 25 FPS 이며, 사용자가 느끼기에 부드러운 화면 갱신을 보여주는 수치이다.

그림 2는 서버 측에서 해상도와 할당된 VCPU 개수에 따른 프레임 당 처리 속도를 보여주고 있다. 괄호 내의 수치는 FPS를 나타내며 화면 전송 성능을 보이고 있다. 해상도가 작아짐에 따라 또한 VCPU 개수가 늘어남에 따라, FPS 가 증가하고 있음을 알 수 있다. HD 해상도의 경우 4 VCPU 할당 시 24.1 FPS 로 목표 화면 전송 성능에 근접하게 전송되고 있는 것을 알 수 있으나, Full HD 해상도의 경우 12.6 FPS 로 목표치의 절반 수준임을 알 수 있다. 흥미로운 사실은 VCPU 1개에서 2개로 설정 시, 연산 소요 시간이 급격히 줄어들고 있으며, 이는 VCPU 간 작업분배가 적절히 일어나고 있다고 판단된다.

그림 3은 서버 측에서 Full HD 동영상을 재생하고 VM 개수에 따른, 프레임 당 처리 속도를 나타내고 있다. 특히 프레임을 처리하기 위한 각 연산 과정을 나타내고 각 소요시간을 보여주고 있다. 각 연산 과정을 나열하면, *init*은 폴링 타이머 초기화, 해상도 변경확인 등 초기화 과정이며, *get_rects*는 프레임버퍼에서 운영체제가 알려주는 변경된 영역 중 보여지는 부분만 잘라내며, *check_rects*는 이전 프레임버퍼와 비교를 통해, 실제 변경된 영역만 잘라내는 연산이며, *encode*는 프레임버퍼를 BMP형식에서 JPG형식으로 압축하는 연산이며, *send*는 클라이언트 측으로 네트워크 전송하는 연산이다. 여러 연산 과정 중 88%~91%의 소요시간이 *encode* 연산에서 소요되는 것을 확인 할 수 있다. 또한 1 VM만 구동시킨 경우, 12.7 FPS를 보여주고 있으며, VM 개수가 늘어날수록 FPS는 더 낮아지는 것을 볼 수 있다.

그림 4는 클라이언트 측에서 Full HD 동영상을 재생하는 서버에 연결하여, 화면을 전송 받을 때 성능이 다른 여러 장치에서 프레임 당 처리 속도를 나타내고 있다. 역시 프레임을 처리하기 위한 각 연산 과정을 나타내고 각 소요시간을 보여준다. *init*은 이후 전송되는 프레임버퍼의 크기, 위치 정보를 받고 초기화하는 과정이며, *recv*는 프레임버퍼를 네트워크를 통해 받는 과정이며, *decode*는 JPG형식에서 BMP형식으로 변환하는

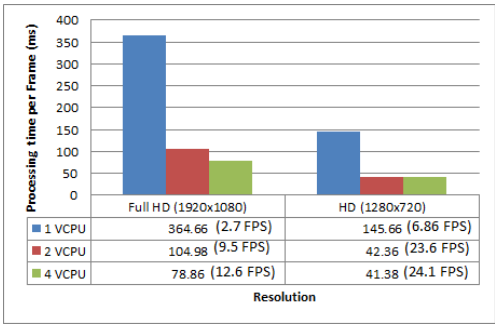


그림 2 해상도 및 VCPU 개수 변화에 따른 프레임 당 처리 속도

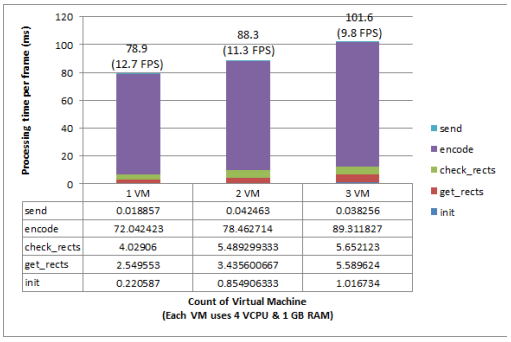


그림 3 Full HD 동영상 재생 시 서버 측 프레임 당 처리 속도

과정이며, *update*는 클라이언트에서 화면을 출력하는 과정이다. *decode*연산이 전체 과정 중 63%~83%의 소요시간을 차지하고 있다. 네트워크 오버헤드인 recv연산은 기가비트 네트워크 상황에서 6%~9%의 비중을 차지하고 있다. 각 장치 별 화면 전송 성능을 보면 Intel i7 950의 데스크탑 PC에서는 13.1 FPS를 보여주고 있으며, Intel Core2duo P8700 노트북에서는 11.1 FPS, Intel Atom D510의 ThinPC에서는 5.1 FPS를 보여주었다.

지금까지의 내용을 정리하면, Full HD 해상도의 원격 화면 전송 성능은, 서버 측에서 4 VCPU 할당 시에 12.6 FPS, 클라이언트 측은 데스크탑PC에서 13.1 FPS 정도를 보여 주었으며, 이는 목표치인 25 FPS의 절반 수준임을 알 수 있었다. 또한 프레임 당 처리시간에서 서버 측은 *encode* 연산과정이 88%~91%로, 클라이언트 측에서는 *decode* 연산과정이 63%~83%로 가장 많은 비중을 차지함을 알 수 있었다. 따라서 고해상도 원격 화면 전송을 위해서 encode, decode 연산 소요시간을 줄이는 것이 핵심적인 기술임을 알 수 있다.

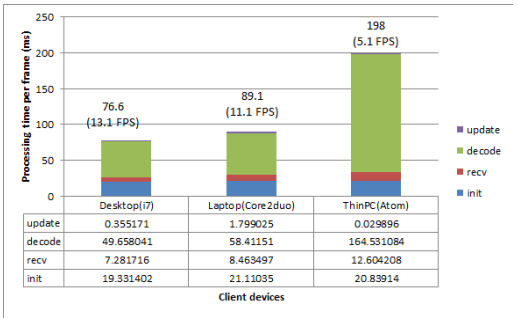


그림 4 Full HD 동영상 재생 시 클라이언트 측 프레임 당 처리 속도

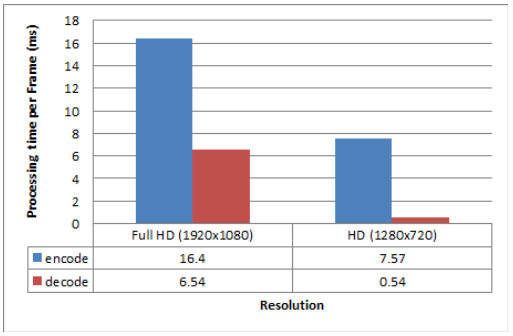


그림 5 프레임 당 H.264 encode, decode 처리 속도

## 3. GPGPU를 활용한 encode/decode 처리속도 향상

이전 장에서 정리한 바와 같이, 원격 화면 전송 성능을 높일 수 있는 핵심 기술은 encode, decode 처리 속도를 빠르게 하는 것이다. 이러한 시도는 VNC 성능을 높이기 위한 방법으로 연구가 진행되어 왔다. 특히, 본 실험에서 사용한 TurboJpeg 라이브러리 [6] 는 CPU의 MMX, SSE, SSE2 SIMD 연산을 통해 encode, decode 처리속도를 높인 솔루션이다. 하지만 본 논문에서는 CPU가 아닌 GPGPU 프로그래밍을 통한 방법을 제안한다.

그림 5는 CUDA 프로그래밍을 사용하여 H.264 encode, decode 하는 소요시간 및 FPS를 보여주고 있다. 사용 소스는 NVIDIA 개발자 사이트에서 배포된 코드 [7] 를 사용하였다. 사용된 실험환경은 Intel i7 950 (3.07 GHz), 4 GB 메모리, NVIDIA GeForce GTX 470을 사용하였다. Full HD 동영상의 경우 encode 소요시간은 16.4 ms 이며 decode 소요시간은 6.54 ms 정도를 보여주고 있다.

이 수준의 소요시간을 적용하는 경우 서버 측의 프

레임 당 처리 속도는 23.3 ms로 43 FPS를, 클라이언트 측은 33.5 ms로 29.8 FPS를 만족할 것으로 기대된다. 이는 목표치 25 FPS를 초과하는 것으로, Full HD 급의 고해상도 화면을 사용하는 사용자에게 끊김없이 부드러운 화면을 제공할 수 있다.

## III. 결 론

가상 데스크탑을 고해상도 화면으로 제공하는 경우, 화면끊김이 발생하여 사용자의 편의성이 떨어지는 문제점을 가지고 있었다. 본 논문은 이러한 문제점을 구체화하고 해결방안을 제시하고 하였다.

먼저 고성능 가상 데스크탑을 지원하는 오픈소스 기반의 가상화 환경 프로토타입을 제안하였다. 이후 해당 프로토타입을 기반으로 성능 프로파일링을 통해 Full HD급 고해상도 원격 화면 전송 시 발생하는 화면 끊김 문제점을 구체화 하였다. 프레임 당 처리시간에서 서버 측은 *encode* 연산과정이 88%~91%로, 클라이언트 측에서는 *decode* 연산과정이 63%~83%로 가장 많은 비중을 차지함을 알 수 있었다. 이러한 연산 오버헤드를 감소시키는 방법으로 본 논문에서는 GPGPU 프로그래밍을 통한 encode, decode 처리 기법을 제안하였다.

향후 GPGPU기반 encode, decode 모듈을 본 VNC 프로토타입에 적용시킬 예정이다. 본 모듈은 오픈소스기반의 다른 원격 화면 전송 기술에도 쉽게 적용 가능할 수 있을 것으로 예상된다.

## 참 고 문 헌

[1] Xen Desktop, http://www.citrix.com/virtualization/desktop/
[2] VMware View, http://www.vmware.com/products/view/overview.html
[3] Oracle Virtualbox, http://www.virtualbox.org/
[4] XenVGAPassthrough, http://wiki.xensource.com/xenwiki/XenVGAPassthrough
[5] TurboVNC, http://www.virtualgl.org/Downloads/TurboVNC
[6] libjpeg-turbo, http://sourceforge.net/projects/libjpeg-turbo/
[7] CUDA SDK Code Samples - NVIDIA Developer Zone, http://developer.nvidia.com/cuda-cc-sdk-code-samples