GPU를 이용한 개선된 뷰포인트 벡터 렌더링 방식의 집적영상시스템 프레임워크에 관한 연구

이빛나라* · 박경신** · 조용주***

Research on the Development of an Integral Imaging System Framework and an Improved Viewpoint Vector Rendering Method Utilizing GPU

Bin-Na-Ra Lee* · Kyoung Shin Park** · Yongjoo Cho***

요 약

컴퓨터-생성 (Computer-generated, CG) 집적영상시스템은 사용자들이 컴퓨터 그래픽을 이용해서 미리 만들어진 기초영상들을 렌즈 어레이를 통해 보게 되면 3차원 입체 영상을 느낄 수 있도록 해주는 무안경식 양안시차 디스플레이 시스템이다. 이 때 CG 집적영상시스템에 컴퓨터를 이용하여 기초영상을 만드는 과정을 이미지 매핑 (Image Mapping)이라고 하는데, 뷰포인트 벡터 렌더링 (Viewpoint Vector Rendering, VVR) 이미지 매핑 방식은 표현하는 대상의 크기나 시스템에서 사용하는 렌즈 어레이 기초렌즈의 개수에 영향을 받지 않아 실시간 처리에 보다 유리하다. 본 논문에서는 실시간 3차원 그래픽 응용 프로그램에 보다 적합한 CG 집적영상시스템을 구축하기 위해 GPU (Graphics Processing Unit)를 이용하여 렌더링 성능을 향상시킨 VVR 집적영상시스템 프레임워크(Framework)를 소개한다. 그리고 일반적인 기존의 VVR 구현 방법과 GPU를 이용하는 새로운 방식의 성능을 비교 분석하며, 상당한 성능 향상이 이루어졌음을 보여준다.

ABSTRACT

Computer-generated integral imaging system is an auto-stereoscopic display system that users can see and feel the stereoscopic images when they see the pre-rendered elemental images through a lens array. The process of constructing elemental images using computer graphics is called image mapping. Viewpoint vector rendering (VVR) method is one of the image mapping algorithm specially designed for real-time graphics applications, which would not be affected by the size of the rendered objects or the number of elemental lenses used in the integral imaging system. This paper describes a new VVR framework, which improved its rendering performance considerably. It also compares the previous VVR implementation with the new VVR work utilizing GPU and shows that newer implementation shows pretty big improvements over the old method.

키워드

집적영상, GPU, 가상현실, 뷰포인트 벡터 렌더링, 이미지 매핑, 프레임워크, 오프스크린 렌더링

* 상명대학교 대학원 컴퓨터과학과
** 한국정보통신대학교 디지털미디어연구소
*** 상명대학교 디지털미디어학부

접수일자 : 2006. 8. 17

I. 서 론

양안시차 (Stereoscopic) 디스플레이는 현재까지 나와 있는 3차원 입체영상 디스플레이 중 쉽게 구현이 가능하다는 장점 때문에, IMAX 3차원 입체영상관이나 가상현실 시스템 구축 등에 가장 보편적으로 사용되고 있는 방식이다. 그러나 사용자들이 셔터 글래스 (Shutter glass)나 편광방식의 특수 안경을 착용해야 입체영상을 볼 수 있기 때문에 사용자가 언제 어디서나 즉각적으로 입체 영상을 볼 수 있는 즉시성이 떨어진다.

이러한 문제를 해결하기 위해, 특정 장치들을 디스플레이에 설치하거나 소프트웨어적으로 해결하여 관찰자의 육안만으로도 3차원 입체영상을 볼 수 있도록 해주는 무안경식 양안시차 (Auto-stereoscopic) 디스플레이 기술이 현재 선진 외국 연구소와 다국적 기업 또한 국내 대기업을 중심으로 많이 연구 및 개발되고 있다. 그러나 아직까지 상호작용적인 가상현실 환경에 적합한 무안경식 양안시차 디스플레이가 사용된 예를 찾아보기는 힘들다.

집적영상시스템(Integral Imaging System)은 무안경식 양안시차 방식으로 부피표현(Volumetric) 디스플레이 방식처럼 연속된 시차와 컬러 정보를 제공하고 관찰자가 시점을 비교적 자유롭게 움직일 수 있다는 장점을 가지고 있어 최근에 와서 주목을 받고 있다[1]. 집적영상시스템은 렌즈 어레이(Lens array)와 카메라 등과 같은 장치들을 이용하여 3차원 정보를 2차원 이미지인 기초영상(Elemental image)으로 변환해서 저장해주는 픽업(Pick-up) 부분과, 이렇게 만들어진 기초영상들을 이용해 다시 입체영상의 형태로 보여주는 디스플레이(Display) 부분으로 나누어진다.

이때 실제 장치들을 이용해 픽업 부분을 구성하지 않고 그림 1에서 보인 것처럼 컴퓨터 그래픽 기술을 이용해서 기초영상을 만들 수 있다. 이렇게 컴퓨터 그래픽을 활용해서 기초영상을 생성하는 시스템을 컴퓨터 생성(Computer Generated, CG) 집적영상시스템이라고 하고, 이런 시스템을 이용하게 되면 실제 카메라로 찍기 어려운 개체나 환경의 재현이 가능할 뿐만 아니라 실제 존재하지 않는 개체들도 만들어서 입체영상으로 볼 수 있다. 또한, 표현할 수 있는 범위의 제한이 줄어들어 기존의 집적영상 시스템에 비해서 사용자들에게 훨씬 다양한 경험을 줄 수 있다.

CG 집적영상시스템에서 기초영상을 생성하는 과정을

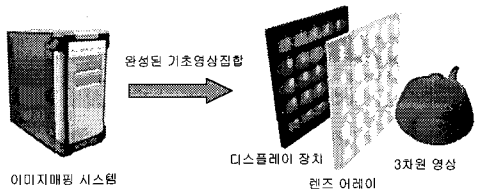

그림 1. CG 집적영상시스템의 개념도
Fig. 1 The conceptual diagram of an Computer-Generated Integral Imaging System

이미지 매핑 (Image Mapping)이라 부른다. 기존에 제안된 대표적인 이미지 매핑 기술들은 화면에 표시할 개체의 모든 점들을 일대일 대응시켜서 만드는 포인트 리트레이싱 렌더링 (Point Retracing Rendering, PRR)[2], 실제 픽업 과정을 컴퓨터 그래픽으로 유사하게 수행하는 다중 뷰포인트 렌더링 (Multiple Viewpoint Rendering, MVR)[3] 및 시스템 요소에 의해 정해진 방향에서 촬영된 장면들을 픽셀 단위로 처리하여 기초영상을 만드는 병렬 그룹 렌더링 (Parallel Group Rendering, PGR)[4], 기초 렌즈 배율에 의한 디렉셔널 장면을 이용하는 뷰포인트 벡터 렌더링 (Viewpoint Vector Rendering, VVR)[5] 등이 있다.

PRR방식은 속도가 매우 느리고, MVR은 기초렌즈의 개수, 디스플레이 장치의 해상도 및 개체의 폴리곤 수에 의해 기초영상의 생성 속도가 크게 영향을 받는다. 또한 PGR은 PRR이나 MVR에 비해 렌더링 속도는 빠른 편이지만, 기초 렌즈 하나가 한 개의 픽셀에 매핑이 되므로 해상도가 다른 방법에 비해서 월등히 떨어지게 되므로 가상현실이나 게임처럼 복잡한 3차원 물체를 실시간 그래픽 처리해야하는 분야에서 사용하기 힘들다.

VVR방식은 기초 렌즈의 배율에 의해 촬영한 디렉셔널 장면을 간단한 이미지 처리를 통해 기초 영상을 만들어내는 이미지 매핑 방법이다. 이 방법은 다른 이미지 매핑 방법에 비해 렌즈 개수나 화면 크기, 폴리곤 개수에 크게 영향을 받지 않아 가상현실 같은 실시간 그래픽에 보다 적합한 방식이다. 하지만 기존에 구현한 VVR방식은 일반 메모리에 렌더링 하는 오프스크린 렌더링 (Offscreen Rendering) 방법을 사용하였기에 속도 향상에 어려움이 많았다. 하지만 최근의 그래픽 카드들이 발전하면서 그래픽 처리 가속 능력을 지닌 GPU(Graphic Processing Unit)를 장착하고 있으면서 다양한 기능과 높은 성능을 제공하여 그래픽 처리 성능을 높일 수 있도록 해주고 있다. 따라서

본 논문에서는 이런 GPU의 그래픽 처리 능력을 활용하기 위해 GPU의 메모리에 렌더링하는 방법을 이용해 성능 향상을 이루었다.

본 논문에서는 먼저 VVR방식에 대해 간단하게 기술하고, GPU를 이용하여 그래픽 속도를 향상시킨 VVR 집적영상시스템을 설명한다. 그리고 VVR 집적 영상 시스템을 사용자들이 만드는 프로그램에 쉽게 적용시킬 수 있도록 디자인된 프레임워크(Framework)에 대해서 설명한다. 마지막으로, 기존의 방법과 GPU를 이용하는 현 시스템의 성능을 비교 분석한다.

II. 뷰포인트 벡터 렌더링 (VVR)

VVR은 가상 카메라를 이용하여 3차원 공간의 디렉셔널 장면을 촬영하고, 이를 미리 계산된 크기로 조각 처리한 후 디스플레이 장치의 윈도우 위치에 재배치하여 기초영상들을 생성함으로써, 실시간 영상처리에 보다 효과적으로 적용할 수 있도록 한 이미지 매핑 방식이다. VVR방식은 기초렌즈에 배율에 따라 촬영해야하는 장면의 횟수가 결정되므로 기초렌즈의 개수가 기초영상 생성속도에 크게 영향을 미치지 못한다. 또한 VVR을 이용하는 이미지 매핑 방법에서는 실제 사용자가 보는 입체 영상물이 OpenGL 같은 3차원 그래픽 라이브러리를 이용해서 구현되므로 3차원 객체를 바로 표현할 수 있다.

또한 VVR방식은 렌즈 어레이와 디스플레이 장치와의 간격이 같을 때 기초렌즈가 픽셀 하나처럼 보이는 초점모드는 물론이고, 기초렌즈의 초점거리가 디스플레이 장치와 렌즈 어레이와의 간격보다 짧아서 집적 영상이 렌즈 앞에 맺히는 리얼모드(Real mode)와 기초렌즈의 초점거리가 디스플레이 장치와 렌즈 어레이와의 간격보다 길어서 집적 영상이 렌즈 어레이 뒤에 맺히는 가상모드(Virtual mode)를 모두 보여줄 수 있다.

VVR에서는 한 장의 입체영상 정보를 담고 있는 완전한 기초영상집합을 얻기 위해서 디렉셔널 장면을 3D 그래픽 라이브러리를 이용해서 가상의 카메라로 촬영하는 과정과 촬영된 디렉셔널 장면을 기초 영상으로 만드는 영상처리 과정을 거친다. 그림 2는 디렉셔널 장면을 촬영하는 과정이다. 촬영할 가상의 공간 범위는 생성할 집적영상시스템의 윈도우 크기와 같다. 가상의 카메라 z 위치는 개발자가 결정한 디스플레이 장치와 렌즈 어레이와의 거리에 기초 렌즈의 배율을 곱해서 결정하게 된다.

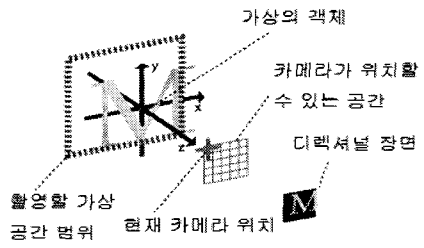

그림 2. 디렉셔널 장면 촬영
Fig. 2 Taking a Directional Scene

VVR방식은 그림 2에서 보인 것처럼 조금씩 다른 위치에서 디렉셔널 장면을 가상의 카메라를 이용해서 찍게 된다. 이 때 디렉셔널 장면의 크기는 촬영할 가상공간 범위를 기초렌즈의 배율로 나눈 것이 된다. VVR 방식을 이용해서 촬영될 가상공간의 크기, 카메라의 z축 위치와 디렉셔널 장면의 크기가 결정되었으면, 이 요소들을 이용하여 가상의 카메라의 FOV(Filed of View)를 계산하여 하나의 디렉셔널 장면을 촬영한다. 그런 후에, 이렇게 촬영된 디렉셔널 장면을 쪼개고 붙이는 이미지 영상 처리부분으로 넘긴다.

그림 3은 디렉셔널 장면을 기초영상으로 만들기 위해 영상처리 하는 과정을 보여주고 있다. 이 과정에서는 그림 2에서 촬영된 디렉셔널 장면과 디스플레이 윈도우를 기초렌즈의 개수만큼 조각을 내고, 디스플레이 윈도우 조각들을 다시 디렉셔널 장면의 개수만큼 조각을 낸다. 그리고 디렉셔널 장면의 조각들은 자신들이 디렉셔널 장면 안에서 몇 번째 조각이냐에 따라 배치될 윈도우 조각이 결정되고, 몇 번째 디렉셔널 장면이냐에 따라서 디스플레이 윈도우 조각 안에서의 위치가 결정되게 된다. 그림 2와 3의 과정을 디렉셔널 장면의 개수만큼 반복하게 되면 하나의 완전한 기초영상 집합이 만들어지게 된다.

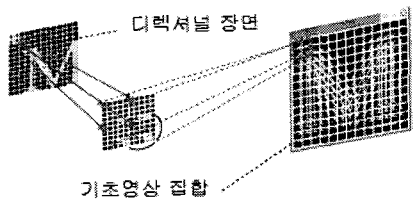

그림 3. 디렉셔널 장면의 영상처리
Fig. 3 Image processing of a directional scene

Ⅲ. GPU를 사용한 VVR 집적영상시스템 설계와 구현

앞에서 설명된 것처럼 VVR은 3차원 객체나 환경을 바로 화면에 렌더링하는 것이 아니라 메모리에 디렉셔널 장면으로 저장한다. 그리고 영상처리를 통해 디렉셔널 장면을 조각내고 다시 접합하는 과정을 거쳐서 최종 결과물인 기초영상들을 비로소 화면에 보여준다. 기존에 구현되었던 VVR은 C++와 OpenGL의 오프스크린 렌더링 기법을 사용하였는데, 생성한 기초영상들의 결과물의 질이 MVR방식과 차이가 나지 않으면서 렌즈 개수나 객체의 폴리곤수가 많아져도 속도에 크게 영향을 미치지 않는 점을 발견할 수 있었다[5]. 하지만 오프스크린 렌더링 방식 자체가 그래픽카드의 GPU를 전혀 사용하지 않는 방식이어서 기초영상을 생성하는 것이 너무 늦다는 단점이 있었다. 또한 VVR을 구현하는 방식 또한 가상현실이나 실시간 그래픽 개발자들이 본인들의 프로그램을 VVR방식을 통해서 보여주려면 그 프로그램들의 상당 부분을 고쳐야 했으므로 불편하였다.

그래서 기존의 VVR의 렌더링 방법을 GPU를 이용하도록 개선하여 생성 속도를 높이고, VVR을 사용하는 프로그램의 구현 및 개발을 쉽게 할 수 있도록 VVR 집적영상 프레임워크를 개발하였다. C++, OpenGL Extension, 그리고 GLUT라이브러리를 이용하여 개발된 VVR 집적영상 프레임워크는 그림 4에서 보인 것처럼 이미지 매핑모듈에서 GPU의 메모리 버퍼를 활성화하여 그 안에 디렉셔널 장면을 만들도록 하였다. 이렇게 만들어진 디렉셔널 장면들은 다시 디렉셔널 장면들을 조각내서 다시 접합하는 영상처리작업을 CPU에 의해 거치게 되며 결과물을 다시 화면에 보이기 위해 렌더링하는 작업을 거치게 된다 이렇게 GPU를 적극적으로 활용하도록 하여 전체적인 성능 향상을 이룰 수 있었다.

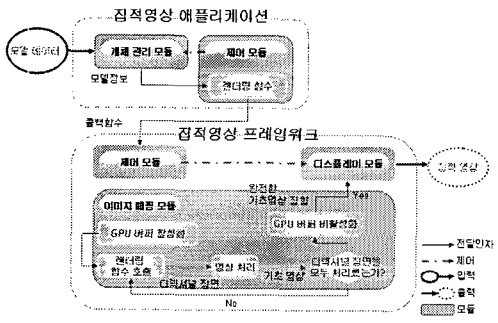

그림 4. GPU를 이용한 뷰포인트 벡터 렌더링을 이용한 집적영상 프레임워크
Fig. 4 The framework of integral imaging system with VVR using GPU

그림 4는 본 논문에서 구현한 VVR 집적영상 프레임워크를 사용해서 입체영상을 보여주는 응용프로그램의 구조를 보여주고 있다. 우선 응용프로그램 개발자는 그림 4에서 보인 객체관리 모듈과 제어 모듈을 구현해야한다. 개체관리 모듈에서는 각 프로그램에서 화면에 그릴 3차원 모델이나 개체들을 관리하는 부분이다. 그리고 제어모듈은 화면에 그릴 장면을 그리는 렌더링 함수와 응용프로그램의 논리적인 부분을 담당하는 각종 함수들, 그리고 집적영상 프레임워크를 초기화시키고 활성화시키는 부분을 포함한다.

응용프로그램에서는 프레임워크를 사용하는 것은 처음 초기화시키면서 화면에 실제로 원하는 내용을 그려주는 렌더링 함수, 사용자로부터 입력을 받았을 때 처리하는 함수 등과 같은 주요 함수들을 집적영상 프레임워크에 등록한다. 그럼 프레임워크에서는 다음과 같은 과정을 거쳐 화면에 기초영상의 집합을 만들어 입체 영상을 보여줄 수 있도록 된다. 우선 프레임워크에서는 그래픽 카드 메모리를 초기화시키고 등록되어 있는 렌더링 함수를 호출해서 GPU의 메모리에 디렉셔널 장면을 만들게 된다. 한 장의 디렉셔널 장면이 생성되면 그 데이터를 GPU의 메모리로부터 일반 메모리에 복사한 후에 그림 3의 영상처리 과정을 거친다. 모든 디렉셔널 장면에 대해 위의 작업을 거치게 되면 일반 메모리에 위치한 버퍼에는 집적영상을 보여주기 위한 기초영상들이 만들어지게 되고 이를 다시 디스플레이 모듈을 통해 실제 화면에 렌더링하게 된다.

그림 5. VVR방식을 이용해 만든 오토바이(왼쪽)와 강아지(오른쪽) 집적영상
Fig. 5 The integral images generated with VVR method. Motorcycle (left) and Dog (right)

이렇게 렌더링 된 기초영상들을 사용자들이 다시 렌즈를 통해서 보게 되면 입체 영상을 보게 되는 것이다. 그림 5는 본 논문에서 제안한 프레임워크로 구현한 집적영상이다. 이 내용에서는 5mm크기의 기초렌즈 1849개를 가지고 있는 렌즈어레이를 사용하는 집적영상 시스템을 보여주고 있다.

Ⅳ. 성능 비교분석 결과 및 토론

기존의 VVR 구현 방법과 새로운 방법을 비교하기 위해 성능 평가 실험을 하였다. 실험 환경은 Intel Pentium4 3.0Ghz, 1GB 메모리, Microsoft Windows XP 기반의 PC를 사용하였고 GPU의 활용도를 보기 위해 GPU를 바꾸어가며 테스트를 하였다. 그림 6은 두 방법의 평가 결과를 보여주고 있다. 그림에서 CPU_VVR1과 GPU_VVR1은 NVidia사의 Quadro FX1300 GPU를 이용하여 평가하였고, CPU_VVR2과 GPU_VVR2는 같은 회사에서 나온 보다 향상된 GPU인 FX1400를 사용하였다.

그림 6의 왼쪽 그래프는 폴리곤의 수가 달라질 때 렌더링 속도를 측정한 결과이다. 이때 사용한 집적영상 시스템은 기초렌즈 1849개, 기초렌즈의 크기 5mm, 픽셀 크기 0.25mm로 디스플레이 윈도우의 크기는 860x860이였다. 이 결과에서 볼 수 있는 것처럼 폴리곤의 개수가 증가될 때 CPU를 이용하는 방법과 GPU를 이용하는 방법 간에는 약 3-6배정도의 렌더링 속도 차이가 나타나는 것을 볼 수 있다.

오른쪽 그래프는 폴리곤의 개수를 고정시키고 렌즈의 개수를 증가시키면서 (즉 입체 영상 이미지의 해상도를 높이면서) 렌더링 속도를 측정한 결과이다. 이때 사용한 집적영상 시스템은 기초렌즈의 크기 10mm, 픽셀의 크기 0.25mm이고, 사용한 모델의 폴리곤 개수는 10000개이다. 이 때 렌즈의 개수를 5x5(해상도: 200x200) 에서부터 51x51(2040x2040)로 늘려가면서 측정했을 때 역시 폴리곤의 개수를 늘릴 때와 비슷한 결과를 보여주고 있다. 즉 FX1300을 사용한 경우에 약 3-4배 정도의 성능 향상을 보여주었고 더 빠른 FX1400을 사용했을 때에는 4-5배정도의 성능향상을 보여주는 것을 확인할 수 있었다.

이렇게 실험 결과에서 볼 수 있는 것처럼 기존의 방법에 비해 GPU를 활용했을 경우 3-6배 정도의 렌더링 속도 향상을 얻을 수 있었다. 그리고 기존의 방법에서는 GPU가 바뀔 때 큰 차이가 없는 것을 확인할 수 있는 반면에, 새로 개발된 방법에서는 GPU_VVR1과 GPU_VVR2 간에 확연한 성능 차이를 보여주고 있어 GPU의 성능이 나아질 때 더 좋은 결과를 얻을 수 있음을 보여주고 있다.

Ⅴ. 결론 및 향후 연구방향

본 논문에서는 GPU를 이용하여 VVR 이미지 매핑 방법의 성능 향상을 얻음으로써 CG 집적영상시스템도 가상현실같은 상호작용 환경에서 사용할 수 있다는 가능성을 보여주었다. 또한 VVR 집적영상 프레임워크를 개발하여 가상현실 응용프로그램 개발자들이 쉽게 사용할 수 있도록 하였다. 이는 CG 집적영상시스템이 무안경식 입체영상 디스플레이가 가상현실 등 다양한 실시간 3차원

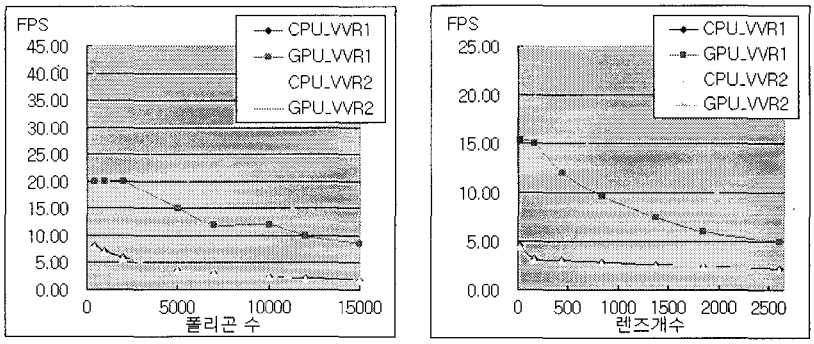

그림 6. 기존의 방법과 GPU를 이용한 렌더링 성능 비교 결과
Fig. 6 The comparison between the previous and the new image mapping method utilizing GPU

입체영상 분야에 널리 사용될 수 있는 계기를 마련한 것이라 할 수 있다.

그렇지만 집적영상 방식 자체가 가지고 있는 낮은 해상도와 입체로 표현할 수 있는 가상공간 두께의 협소함 같은 단점은 아직까지는 게임이나 가상현실 분야에 사용하기에는 걸림돌이 되고 있다. 이를 위해 향후 고해상도 타일 디스플레이(Tiled Display) 기법을 사용하여 낮은 해상도를 극복하고 입체로 표현할 수 있는 공간의 협소함을 극복할 수 있는 알고리즘 개발을 위해 연구할 것이다. 또한 프레임워크를 집적영상뿐만 아니라 현존하는 다양한 3D 입체영상 기법과 더 향상된 기법들을 편리하게 사용할 수 있도록 확장하여 3차원 입체영상 응용프로그램 개발을 더욱 편리하게 할 수 있도록 지원할 것이다.

참고문헌

[1] Lee,B ., Min, S., Jung, S., Park, J., "A Three-dimensional display system based on computer-generated integral photography", Journal of the Society for 3D Broadcasting and Imaging, 1, pp. 78-82, (2000)

[2] Yoshihide Igarashi, H. Murata and M. Ueda, "3D display system using a computer generated integral photography," Japan J. Appl. Phys. 17, pp. 1683 (1978)

[3] Myunghoon Suk "Enhanced image mapping algorithm for 3D integral imaging display system", Information and Communications University (2005)

[4] Ruigang Yang, Xinyu Huang, Shunnan Chen, "Efficient Rendering of Integral Images", SIGGRAPH2005 (2005)

[5] Sung-Wook Min, Kyoung Shin Park, Binara Lee, Yongjoo Cho, Minsoo Hahn, Enhanced Image Mapping Algorithm for Computer-Generated Integral Imaging System, Japanese Journal of Applied Physics, Vol. 45, No. 28, pp. L744-L747, 2006.

저자소개

이빛나라(Bin-Na-Ra Lee)

2004년 상명대학교 소프트웨어학부
2004년 ~ 현재 상명대학교 대학원 컴퓨터과학과 (석사과정 재학중)

※관심분야 : 가상현실, 게임, 게임엔진

박 경 신(Kyoung Shin Park)

1991년 덕성여자대학교 수학과
1997년 University of Illinois at Chicago 전기전자 컴퓨터과학과 공학석사
2003년 University of Illinois at Chicago 컴퓨터과학과 공학박사
2004년 ~ 현재 한국정보통신대학교 디지털미디어연구소 연구교수

※관심분야 : 가상현실, HCI, 감성공학, 인터랙티브미디어, Edutainment

조 용 주(Yongjoo Cho)

1993년 University of Illinois at Urbana-Champaign 컴퓨터과학과
1997년 University of Illinois at Chicago 전기전자컴퓨터과학과 공학석사
2003년 University of Illinois at Chicago 컴퓨터과학과 공학박사
2004년 ~ 현재 상명대학교 디지털 미디어학부 조교수

※관심분야 : 가상현실, 인터랙티브 컴퓨팅, Edutainment 환경