순차적 이중 전방 사상의 병렬 처리를 통한 다중 시점 고속 영상 합성

정회원 최 지 윤*, 유 세 운*, 신 홍 창**, 박 종 일*

Fast Multi-View Synthesis Using Duplex Foward Mapping and Parallel Processing

Ji-Youn Choi*, Sae-Woon Ryu*, Hong-Chang Shin**, Jong-Il Park* ***Regular Members***

요 약

3차원 입체 영상을 디스플레이에 출력하려면, 여러 시점에서의 영상 정보가 필요하다. 여러 시점의 영상을 얻을 수 있는 가장 기본적인 방법은, 필요로 하는 시점의 개수와 동일 한 수의 카메라를 사용하는 것이다. 하지만 이를 위해서는 카메라간의 동기화 와 방대한 데이터 처리 및 전송 등의 현실적인 문제가 해결되어야 한다. 이러한 현실적인 문제를 해결하기 위해서 연구되고 있는 방법이 한정된 시점 영상을 이용하여 여러 중간 시점 영상을 생성하는 영상 기반의 임의 시점 합성 방법이다. 본 논문에서는 두 개의 기준 시점 영상과 각각의 깊이 정보가 주어졌음을 가정 하고 주어진 정보를 바탕으로 이중의 순차적인 전방 사상을 통하여 목표로 하는 여러 다중 시점의 영상을 동시에 합성하는 방법을 제시한다. 제안된 방법은 좌우 기준 시점 영상의 평행 이동으로 가상 시점 영상을 생성 할 수 있으며, 평행 이동은 시점의 거리에 비례한 행렬간의 관계로 나타난다. 따라서 이중의 순차적인 전방 사상이라 함은 좌우 시점에서 가상 시점 거리에 따른 관계식을 통한 순차적인 양안 시점의 평행 이동을 의미한다. 이 때 전방 사상을 통해 생성되는 가상 시점 영상과 기준 시점 영상간의 기하관계가 시점간 거리에 비례하여 반복적이므로 이를 GPU 프로그래밍을 통해 병렬 처리를 통해 고속화 하는데 초점을 맞추었다.

Key Words : 3D Display, View Synthesis, Parallel Processing, GPU(Graphic Processing Unit) Processing, OpenGL Shading Language

ABSTRACT

Glassless 3D display requires multiple images taken from different viewpoints to show a scene. The simplest way to get multi-view image is using multiple camera that as number of views are requires. To do that, synchronize between cameras or compute and transmit lots of data comes critical problem. Thus, generating such a large number of viewpoint images effectively is emerging as a key technique in 3D video technology. Image-based view synthesis is an algorithm for generating various virtual viewpoint images using a limited number of views and depth maps. In this paper, because the virtual view image can be express as a transformed image from real view with some depth condition, we propose an algorithm to compute multi-view synthesis from two reference view images and their own depth-map by stepwise duplex forward mapping. And also, because the geometrical relationship between real view and virtual view is repetitively, we apply our algorithm into OpenGL Shading Language which is a programmable Graphic Process Unit that allow parallel processing to improve computation time. We demonstrate the effectiveness of our algorithm for fast view synthesis through a variety of experiments with real data.

※ 본 연구는 방송통신위원회, 지식경제부 및 한국산업기술평가관리원의 IT 원천기술개발사업의 일환으로 수행한 연구로부터 도출된 것이다. [과제관리번호: 2008-F-011, 과제명: 차세대 DTV 핵심기술 개발].

* 한양대학교 컴퓨터공학과 가상현실 연구실(jeromechoi@mr.hanyang.ac.kr), ** 한국전자통신연구소 방송시스템연구부(hcshin@etri.re.kr)

논문번호 : #KICS2009-08-353, 접수일자 : 2009년 8월 8일, 최종논문접수일자 : 2009년 11월 10일

Ⅰ. 서 론

3차원의 입체감이 있는 이미지를 출력 하는 방법은 스테레오 비전, 홀로그램 등이 있으며 각 분야에서 활발하게 연구개발이 진행되고 있다. 이 중 다중 시점 스테레오 비전 기술을 이용한 3차원 디스플레이는 2차원의 색상 정보에 깊이 정보를 추가 하여 다중 시점을 생성하고 이를 디스플레이에 출력하여 사용자에게 입체감 있는 멀티미디어 서비스를 제공한다.

무안경식 입체 디스플레이어는 여러 시점의 영상을 동시에 투영해 줌으로서 사용자로 하여금 입체감을 느끼게 한다[1], [2]. 다중 시점 영상은 디스플레이어가 필요로 하는 시점 수와 동일한 수의 카메라를 사용하여 얻을 수 있다. 하지만 디스플레이어마다 필요한 영상의 시점 수가 동일하지 않고 카메라 간의 동기화와 방대한 데이트 처리 및 전송 등의 현실적인 문제가 해결되어야 한다.

이러한 현실적인 문제를 해결하기 위해서 연구되고 있는 방법이 소수의 한정된 시점 영상을 이용하여 다수의 가상 시점 영상을 생성하는 영상 기반의 임의 시점 합성 방법이다. 그 동안 다양한 형태의 영상 기반의 임의 시점 합성 방법이 제안되어 왔으며, 최근에는 생성된 합성 영상의 화질을 개선하기 위한 방법들이 많이 소개되고 있다. 예로, 경계 영역을 추출하고 이를 보정하는 방법[3]이나 색상 정보와 깊이 정보를 보간 하여 가상 시점의 영상 화질을 향상 시키는 방법[4][5]등을 들 수 있다.

영상 기반의 임의 시점 합성 방법이 3차원 디스플레이에 적용되기 위해서는 필요로 하는 시점을 실시간으로 합성할 수 있어야 한다. 하지만, 알고리즘의 개선으로는 한계가 있으며, 시점이 개수에 비례하여 계산 시간은 증가할 수밖에 없다. 한 가지 방안은 영상 합성 과정이 동일한 연산의 반복 수행으로 이루어진다는 것에 착안하는 것이다. 즉 반복적인 연산을 병렬 처리한다면, 이상적으로는 시점의 개수에 상관없이 실시간으로 가상 시점 영상을 합성할 수 있다. 이와 관련하여 병렬 처리에 적합한 GPU(Graphic Processing Unit)을 이용하여 가상 시점 영상을 고속으로 합성한 사례를 들 수 있다[1].

본 논문은 SHIN[1]의 방법을 개선한 것으로, 여기에서와 달리 가상 영상 합성을 위한 기준 영상으로, 양안 시점의 영상과 각 양안 시점 영상의 깊이 지도가 모두 주어진다고 가정한다. 두 기준 영상의 깊이 정보를 모두 알기 때문에 기본적으로 양쪽 시점으로부터의 전방 사상을 통해 중간 시점의 깊이 지도를 생성할 수 있다. 이를 통해 중간 시점 영상의 화질 및 합성 속도를 크게 개선할 수 있다. 또한 기존에는 CUDA(Compute Unified Device Architecture)를 사용하였으나, 본 논문에서는 GLSL (OpenGL Shading Language)을 사용함으로써 운영체제 및 그래픽 카드에 대한 호환성을 크게 개선할 수 있다.

본 논문은 다음과 같이 구성된다. 먼저 2장에서는 3차원 디스플레이 시스템에 대하여 간략하게 설명하고, 3장에서는 기준 영상 데이터를 GLSL를 이용하여 처리하는 방법을 설명한다. 4장에서는 제안한 가상 시점 영상 생성 알고리즘을 설명하고, 이를 구현한 결과 및 실험 결과를 5장에서 보인다. 마지막으로 6장에서 결론을 맺는다.

Ⅱ. 3차원 디스플레이 시스템

그림 1은 본 논문이 가정하는 3차원 디스플레이 시스템의 개요도이다. 스테레오 카메라로 촬영한 양안 영상과 그들 각각의 깊이 지도가 전송되면 장치 내의 영상 합성 모듈에서 다중 시점 영상을 합성하

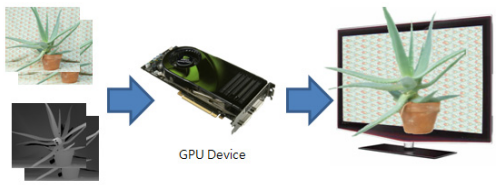


그림 1. 3차원 디스플레이 시스템의 개요도
Fig. 1. Concept of 3D Display System

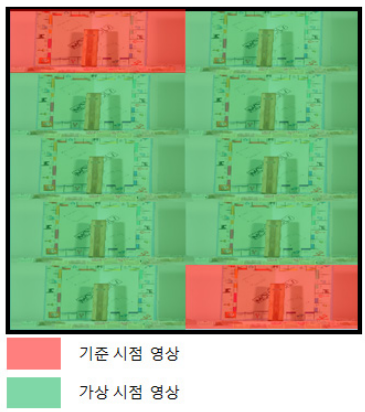


그림 2. 10개의 시점으로 구성된 단일 영상
Fig. 2. Ten views on one a tile image

고 3차원 디스플레이 장치에 넘겨준다.

3차원 디스플레이 장치에서 영상을 출력하기 위해서는 다시점 영상이 필요하다. 예를 들어 LG 3D TV는 그림 2와 같이 2개의 가로 시점과 5개의 세로 시점, 총 10개 시점인 단일 영상을 입력 받아 출력한다.

이 때, 10개의 시점 영상은 2개의 기준 영상과 8개의 가상 시점 영상이 된다. 따라서 8개의 가상 시점 영상을 실시간으로 생성하기 위한 기술이 필요하게 된다.

Ⅲ. 영상 데이터의 처리

본 장에서는 GPU와 GLSL에 대해 설명하고, 색상 정보와 깊이 정보를 입력 받아 데이터를 처리해 나가는 기본적인 방식에 대하여 설명한다.

3.1 GPU

중간 시점의 영상이 기준 시점 영상에 대해 수평으로 평행하다고 가정하였을 때 기준 시점 영상의 기하학적 변화는 세로 방향의 평행 연산과 같아진다. 이러한 연산을 가상 시점 수만큼 반복하는 구조는 CPU로 처리 하는 것 보다 GPU의 스트림 프로세서를 활용한 연산을 통해 고속화함으로서 큰 속도 향상을 기대 할 수 있다. 현재 시판중인 GPU를 비교하면 표 1과 같다.

표 1. 두 제조사의 GPU의 세부 정보 비교
Table 1. Comparison of GPUs of two major manufacturers

항목	Radeon HD 4890	Nvdia GTX 280
메모리	GDDR5 1GB	GDDR3 1GB
프로세서 개수	800개	240개
프로세서 클럭	850Mhz	600Mhz
버스	256bit	512bit

3.2 GLSL pipeline

GPU를 이용한 연산 방법은 CUDA, GLSL 등 여러 가지가 있다. 이 중 OpenGL기반의 GLSL은 매킨토시나 윈도우, 리눅스 등 여러 운영 체제 간 호환이 되며, 여러 제조사의 그래픽 카드에서 동작하는 등의 장점이 있기 때문에 고속 프로세서의 선택 폭이 넓다. GLSL의 파이프라인은 그림 3과 같이 버텍스 셰이더(vertex shader)와 프래그먼트 셰이더(fragment shader)로 나뉜다.

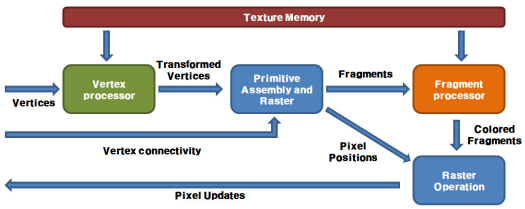


그림 3. GLSL 파이프라인
Fig. 3. GLSL pipeline

버텍스 셰이더는 각 폴리곤의 버텍스에 관련된 값, 법선, 좌표 등을 바꾸는 방법을 제공한다. 이러한 기능은 DirectX 8에서 처음으로 소개되었고 OpenGL에서도 사용 가능하다.

버텍스 셰이더를 사용할 때는 기존의 고정 함수 파이프라인(fixed function pipeline)을 사용 할 수 없다. 법선 벡터가 회전과 이동으로 된 행렬에 의해 변환 될 때, 계산된 법선 벡터는 다시 정규화 될 필요가 없지만 고정 함수 파이프라인에서는 이 과정을 모두 수행하게 된다. 이러한 파이프라인을 제어하여 파이프라인을 최적화하는 것이 버텍스 셰이더이다[8].

프래그먼트 셰이더의 입력 값은 이전 단계에서 계산되어 보간 된 값들로써, 정점의 위치, 색상, 법선 벡터 등이다. 버텍스 셰이더에서 이러한 값들은 각각의 정점에 대해서 계산된다. 여기서는 프래그먼트 내부를 다룸으로 보간 된 값을 필요로 한다. 프래그먼트 셰이더는 하나의 프래그먼트 에 대해서 작동을 한다. GLSL의 세부적인 동작은 그림 4와 같다[8].

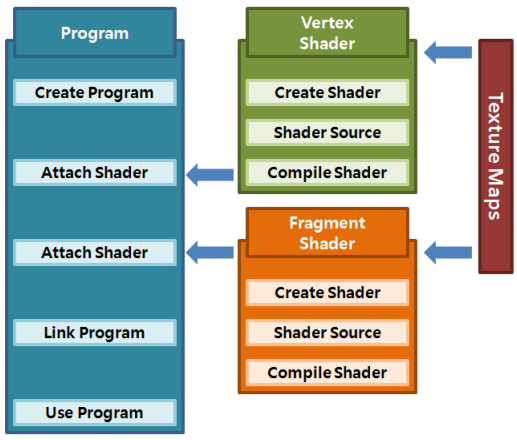


그림 4. GLSL의 세부적인 동작
Fig. 4. GLSL steps to create the shaders

3.3 OpenGL기반 입력 데이터의 처리

영상 기반 가상 시점 합성을 위해서는 영상의 칼라 정보와 깊이 정보를 동시에 활용하여 처리해야 한다. OpenGL은 기본적으로 BGR의 칼라 정보와 함께 투명도를 결정하는 알파 정보를 함께 구성 하는 데이터 구조를 가지고 있다. 이 알파 정보에 깊이 정보를 삽입 하여 하나의 텍스쳐 이미지로 만들 수 있다. 따라서 하나의 기준 시점에서 가상 시점을 만들기 위해서는 하나의 텍스쳐에 담겨져 있는 정보를 활용 하면 된다. 그림 5는 OpenGL기반 입력 데이터의 처리 방법이다.

두 시점의 색상 정보와 깊이 정보를 입력 받아 이를 각각의 텍스쳐로 저장 후 GPU로 보내 가상 시점을 생성하게 되고, 결과를 하나로 묶어서 출력 한다.

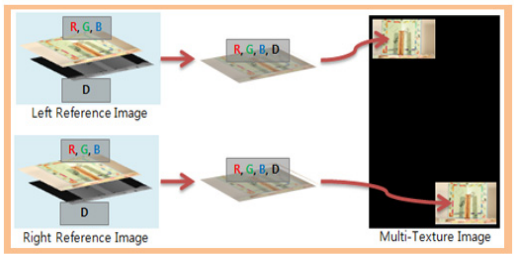


그림 5. OpenGL기반 입력 데이터 구조
Fig. 5. Input data structure based on OpenGL

Ⅳ. 가상 시점 영상 합성

본 장에서는 가상 시점 영상 합성의 기본적인 원리를 설명하고, 전방 사상을 통하여 가상 시점 영상을 합성하여 다시점의 단일 영상을 획득하는 방법에 대해서 설명한다.

4.1 가상 시점 생성 원리

임의의 가상 시점에 대한 영상을 합성하기 위해서는 기본적으로 두 대의 카메라 시점의 위치와 촬영된 영상의 기하학적 관계를 활용 한다[2].

시점의 위치와 영상간의 기하학적 관계는 행렬의 형태로 나타낼 수 있으며, 시점의 변화는 시점과 영상간의 관계를 표현하는 행렬간의 연산으로 나타낼 수 있다. 따라서 카메라와 영상의 기하 관계를 이용하여 중간 시점 영상을 구할 수 있다. 식 1은 기준 카메라의 시점 X_o에서 가상 시점 X_n으로의 평행 이동과 회전을 나타낸다.

$$X_n = R^* X_o + T \tag{1}$$

식 1을 기본으로 시점의 좌우 이동과 시점의 상하 이동시의 기하학적 변화는 식 2로 나타 낼 수 있다.

$$x_n = F_n \frac{X_n}{Z_n},\ y_n = F_n \frac{Y_n}{Z_n} \tag{2}$$

식 1과 식 2를 이용하여 전개 하면 시점의 이동에 따른 영상의 변화를 식 3과 식 4를 통하여 정의 할 수 있다.

$$x_n = F_n \frac{r_{11}x_o + r_{12}y_o + r_{13}F_o + \frac{F_o}{Z_o}T_x}{r_{31}x_o + r_{32}y_o + r_{33}F_o + \frac{F_o}{Z_o}T_z} \tag{3}$$

$$y_n = F_n \frac{r_{21}x_o + r_{22}y_o + r_{23}F_o + \frac{F_o}{Z_o}T_y}{r_{31}x_o + r_{32}y_o + r_{33}F_o + \frac{F_o}{Z_o}T_z} \tag{4}$$

이 때 좌우 카메라의 시점이 수직 방향으로 변화가 없음을 가정하면 세로 방향 성분에 대한 연산이 불필요하다. 따라서 식 4는 식 5로 단순화 된다.

$$y_n = y_o \tag{5}$$

따라서 기준 시점의 영상으로부터 가상 시점 영상 생성을 위한 기하학적 변화는 x_n의 선형적 변환과 같게 된다.

4.2 가상 시점 영상 합성

그림 6은 좌우 기준 시점과 그 중간의 가상 시점간의 관계를 나타내고 있다.

이 때 f_r은 좌측 기준 시점부터 가상 시점까지 전방 사상이 됨을 의미하고, f_l은 우측 기준 시점부

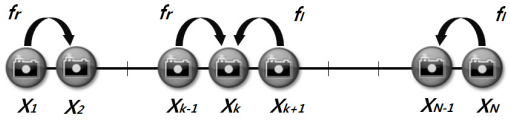


그림 6. 양안 기준 시점과 중간 가상 시점간의 관계
Fig. 6. Geometrical relationship between reference view and virtual view

터 가상 시점 까기 전방 사상이 됨을 의미한다.

$I(X_k)$는 임의의 시점 X_k에서 바라본 영상 이미지이며, $D(X_k)$는 해당 시점에서의 깊이 지도를 말한다. 따라서 좌측의 기준 시점을 기준으로 k번째 시점에서의 가상 영상은 다음과 같이 전개 된다.

$$I(X_k) = f_r(I(X_{k-1}), D(X_{k-1})) \quad (6)$$

3장 3절에서 영상 이미지와 깊이 지도가 하나의 데이터 집합임을 가정하였기 때문에 식 6은 아래와 같이 전개 되어 진다.

$$f_r(I(x_{k-1}), D(X_{k-1})) \rightarrow f_r(I(X_{k-1})) \quad (7)$$
$$I = (r, g, b, d)$$

이 때 r, g, b는 영상 이미지의 색상 정보이고, d는 깊이 지도의 깊이 정보이다. 식 7를 기본으로 좌측 기준 영상 X_1부터 k만큼 떨어진 시점의 영상은 식 8과 같이 전개된다.

$$I(X_k) = f_r(I(X_{k-1})) = f_r^{k-1}(I(X_1)) \quad (8)$$

같은 방법으로 우측 기준 영상 X_N으로부터 k만큼 떨어진 시점의 영상은 식 9와 같이 전개된다.

$$I(X_k) = f_l(I(X_{k+1})) = f_r^{N-k+1}(I(X_N)) \quad (9)$$

식 8과 식 9를 토대로 k만큼 떨어진 위치에서의 영상은 식 10과 같다고 가정 할 수 있다.

$$I(X_k) = f_r(I(X_{k-1})) = f_r^{k-1}(I(X_1)) = f_l(I(X_{k+1})) = f_r^{N-k+1}(I(X_N)) \quad (10)$$

이 때 각 픽셀의 깊이 정보에 따라 출력 될 색상 정보를 선택하기 위하여 다음과 같은 조건을 사용한다.

$$\begin{aligned} &\text{if}\,(d_L \geq d_R)\; I_{u,v}(X_k) = f_r^{k-1}(I_{u,v}(X_1)) \\ &else \qquad\qquad I_{u,v}(X_k) = f_r^{N-k+1}(I_{u,v}(X_n)) \end{aligned} \quad (11)$$

위 식을 토대로 전방 사상은 그림 7과 같이 진행 된다.

붉은 영역은 해당 시점에서 가까운 기준 시점을 기준으로 전방 사상을 하게 되고 채워지지 않은 영역은 반대쪽 기준 시점의 전방 사상을 통해 채워진다.

기준 시점에서 목표 시점까지의 기하학적 변화는 목표 시점 바로 전의 가상 시점의 관계에 의해 덧셈 연산으로 진행되기 때문에 연산 속도가 줄어든다.

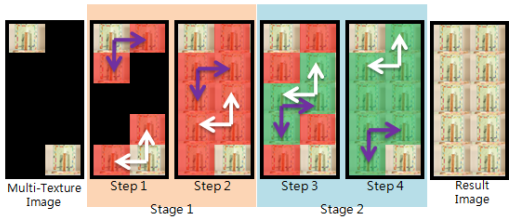


그림 7. 10개 시점의 단일 영상 생성 과정
Fig. 7. Processing pipe-line of 2 by 5 tile image

V. 구현 결과

본 장에서는 구현 결과에 대해서 설명한다. 설계된 알고리즘은 GLSL을 이용하여 구현되었고, 샘플 영상은 Middelbury 대학의 스테레오 연구실에서 제공하는 기본 샘플 영상 데이터를 사용하였으며 표 2에 나타난 3가지 영상을 사용하였다[11].

구현한 시스템의 전체적인 하드웨어의 세부 사양은 표 3과 같다.

표 2. 실험에 사용한 기본 샘플 영상
Table 2. Sample images used for experiment

		
Monopoly	Plastic	Moebius

표 3. 하드웨어 세부 정보
Table 3. Experimental Environment

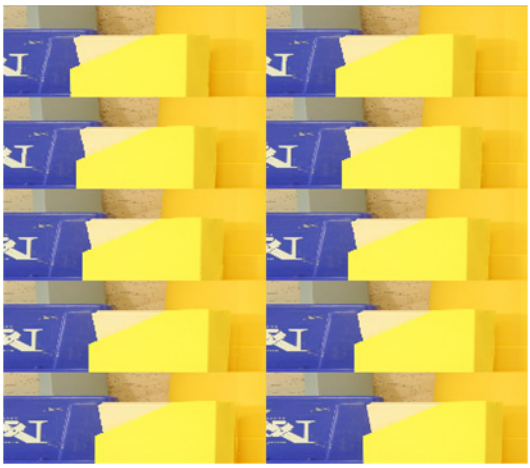
항목	세부 사항
CPU	Intel Core2 Quad CPU Q9400 2.66GHz
MEM	3071MB
GPU	Nvidia Geforce GTX285
OS	Window Vista UL. 64bit

5.1 속도 측정

최종적으로 출력된 영상은 그림 8과 같이 총 10개 시점의 단일 영상을 출력하였으며 각 샘플 영상에 대한 속도는 표 5와 같다.

10개 시점의 단일 영상 한 장을 생성하는데 필요

Monopoly



Plastic

Moebius

그림 8. Monopoly, Moebius 그리고 Plastic의 10개 시점 단일 영상 생성 결과
Fig. 8. Results of generating a tiled image (2 by 5) with Monopoly, Moebius and Plastic

한 단계별 수행 시간은 3개의 실험 영상을 그림 7

표 5. 다시점 단일 영상 생성 계산 시간 [ms]
Table 5. Computation Time [ms]

영상	ms
Monopoly	67.72
Moebius	62.15
Plastic	66.10

표 6. 단계별 수행 시간 [ms]
Table 6. Step wise computation time [ms]

단계	ms
Input/Output	2
Step 1.	14
Step 2.	15
Step 3.	17
Step 4.	17
Total	65

에 언급 한 각각의 단계별로 측정하여 평균을 구하였으며 그 결과는 표 6과 같다.

5.2 화질 평가

화질 평가는 좌우 기준 영상을 가지고 중간 시점의 영상을 목표로 가상적인 영상을 생성하여 중간 시점의 ground truth 영상과의 차이를 구함으로서 측정하였다.

이 때 중간 시점의 ground truth 영상은 Middlebury 대학의 스테레오 연구실의 데이터를 활용하였다[11]. PSNR로 화질을 측정하였으며 측정된 화질의 결과는 표 7에 나타내었고 생성된 영상은 그림 9와 같다.

표 7. 합성된 영상과 ground truth 영상의 차이 비교 (PSNR)
Table 7. Difference between synthesized images and ground truth images (PSNR)

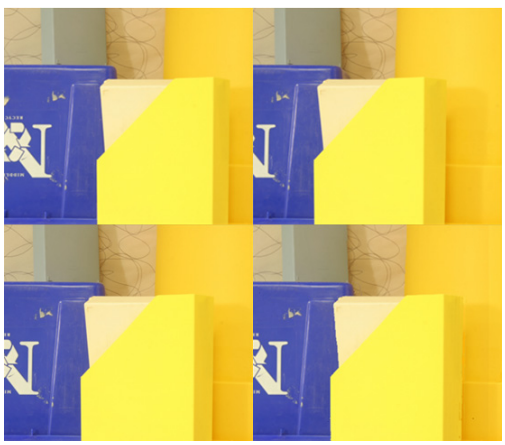
영상	PSNR
Monopoly	21.96 dB
Plastic	28.06 dB
Moebius	22.29 dB

Ⅵ. 결 론

본 논문에서는 2차원 영상의 3차원 디스플레이 시스템으로의 적용을 위해 두 개의 기준 시점 영상과 각각의 깊이 정보를 토대로 순차적 이중 전방 사상 및 병렬 처리 함으로서 가상의 다중 시점 영

Monopoly

Moebius



Plastic

a	b
c	d

그림 9. Monopoly, Moebius 그리고 Plastic의 화질 측정. a) 좌측 시점 영상, b) 우측 시점 영상, c) 중간 시점의 ground truth 영상, d) 가상 시점 영상
Fig. 9. Measurement of image quality of Monopoly, Moebius, and Plastic. a) left view image, b) right view image, c) gorund truth image of center view, d) Synthesized image of center view

상 합성을 고속화하는 방법을 제안하고 구현 하였다.

앞으로 고속화 기법의 개선을 통하여 실시간 영상 합성이 가능하도록 알고리즘을 최적화 하고, 제안한 방법에 후처리 과정을 적용함으로서 속도와 화질 모두 향상 시키는 연구가 필요하다.

참 고 문 헌

〔1〕 H.-C. Shin, Y.-J. Kim, H. Park, J.-I. Park, “Fast view synthesis using GPU for 3D display,” *IEEE Trans. Consumer Electronics*, Vol.54, No.4, pp. 2068-2076, Nov., 2008.

〔2〕 D, Scharstein, “View synthesis using stereo vision,” *Lecture Notes in Computer Science (LNCS)*, Vol.1583, 1999

〔3〕 M. Bleyer, M. Gelautz, and C. Rother, C. Rhemann, “A stereo approach that handles the matting problem via image warping,” *Computer Vision and Pattern Recognition*, 2009.

〔4〕 K. Wegner and O. Stankiewicz, “Similarity measures for depth estimation,” *Proc. of 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, 2009.

〔5〕 K.-J. Oh, S. Yea, and Y.-S. Ho, “Hole-filling method using depth based in-painting for view synthesis in free viewpoint television (FTV) and 3D video,” *Proc. of Picture Coding Symposium (PCS)*, 2009.

〔6〕 P. Kauffm, N. Atzpadin, C. Fehn, M. Muller, O. Schreer, A. Smolic and R. Tanger, “Depth map creation and image-based rendering for advanced 3DTV services providing inter-operability and scalability,” *Signal Processing: Image communication*, Vol.22, No.2, pp. 217-234, 2007

〔7〕 C. Jin and H. Jeong, “Intermediate view synthesis for multi-view 3D displays using belief propagation-based stereo matching,” *Proc. of International Conference on Con-vergence and Hybrid Information Technology*, Vol.1, pp.919-924, 2008

〔8〕 R. J. Rost, *OpenGL Shading Language*, second edition, Addison Wesley Professional, 2006.

〔9〕 R.S. Wright, Jr. and B. Lipchak, *OpenGL Superbible*, 3rd edition, Sams Publishing, 2005.

〔10〕 M. Pharr and R. Fernando, *GPU Gems 2: Programming Techniques for High Performance Graphics and General-Purpose Computation*, 2005.
〔11〕 Middlebury Stereo Vision Page, http://vision.middlebury.edu/stereo
〔12〕 GPGPU, http://www.gpgpu.org

최 지 윤 (Ji-Youn Choi) 정회원

2009년 2월 한양대학교 전자통신컴퓨터공학부 학사
2009년 3월~현재 한양대학교 전자컴퓨터통신공학부 석사과정
<관심분야> 가상현실, HCI, 3차원 영상처리, 컴퓨터그래픽스/비전, GPGPU

유 세 운 (Sae-Woon Ryu) 정회원

2002년 2월 동국대학교 전자전기공학부 학사
2005년 3월 한양대학교 전자컴퓨터공학부 석사
2005년~현재 한양대학교 전자컴퓨터공학부 박사 과정
<관심분야> 3D 컴퓨터 비전, 모델 렌더링, GPGPU

신 홍 창 (Hong-Chang Shin) 정회원

2005년 2월 세종대학교 컴퓨터공학과 학사
2008년 8월 한양대학교 전자통신컴퓨터공학과 석사
2009년 3월~현재 한국전자통신연구소 방송시스템연구부 연구원
<관심분야> 3차원 영상처리, 3차원 디스플레이, 영상 합성, GPGPU

박 종 일 (Jong-Il Park) 정회원

1987년 서울대학교 전자공학과 학사
1989년 서울대학교 전자공학과 석사
1995년 서울대학교 전자공학과 박사
1992년~1994년 일본 NHK방송기술연구소 객원연구원
1995년~1996년 한국방송개발원 선임연구원
1996년~1999년 일본 ATR지능영상통신연구소 연구원
1999년~현재 한양대학교 전자통신컴퓨터공학과 교수
<관심분야> 가상현실, 증강현실, HCI, 3차원 영상처리, 컴퓨터그래픽스/비전