실시간 3D 브라우징 시스템을 위한 램 디스크 기반의 다시점 영상 합성 기법의 설계 및 구현

Design and Implementation of Multiple View Image Synthesis Scheme based on RAM Disk for Real-Time 3D Browsing System

심춘보, 임은천
순천대학교 정보통신공학부

Chun-Bo Sim(cbsim@sunchon.ac.kr), Eun-Cheon Lim(eclim@sunchon.ac.kr)

요약

다시점 영상 처리 기술은 다시점 디스플레이 장치와 압축된 데이터 복원 장치를 통해 장치 사용자의 시각에 3차원의 입체 영상을 제공하는데 목적이 있다. 본 논문은 4시점의 병렬 카메라를 통해 실시간으로 입력되는 스테레오 이미지들에 대해서 효율적인 영상 합성을 통해 3차원 입체 영상을 브라우징할 수 있는 램 디스크 기반의 다시점 영상 합성 기법을 제안한다. 제안하는 기법은 입력 영상들을 이진화 영상으로 변환한 후, Sobel 및 Prewitt 에지 발견 알고리즘을 적용시키고 이를 토대로 4개 영상들의 시차를 구한다. 아울러 기존의 알고리즘에서 모호하게 언급되었던 동기화 문제를 해결하기 위해 하드웨어 트리거와 소프트웨어 트리거를 위한 시간 간격을 적용한다. 제안하는 기법을 분산 환경에서도 적용할 수 있도록 영상의 스냅샷에 대한 유일한 식별자를 이용한다. 성능 분석 결과, 전체 영상(왼쪽, 오른쪽) 및 시차정보를 모두 전송하여 고정밀의 3차원 입체 영상을 출력하는 데 소요되는 전체 시간은 각 이진 배열 당 약 0.67초로 실시간으로 적용하는 데 적합하다고 볼 수 있다.

■ 중심어 : | 3차원 브라우징 | 영상 합성 | 다시점 | 램 디스크 |

Abstract

One of the main purpose of multiple-view image processing technology is support realistic 3D image to device user by using multiple viewpoint display devices and compressed data restoration devices. This paper proposes a multiple view image synthesis scheme based on RAM disk which makes possible to browse 3D images generated by applying effective composing method to real time input stereo images. The proposed scheme first converts input images to binary image. We applies edge detection algorithm such as Sobel algorithm and Prewiit algorithm to find edges used to evaluate disparities from images of 4 multi-cameras. In addition, we make use of time interval between hardware trigger and software trigger to solve the synchronization problem which has stated ambiguously in related studies. We use a unique identifier on each snapshot of images for distributed environment. With respect of performance results, the proposed scheme takes 0.67 sec in each binary array. to transfer entire images which contains left and right side with disparity information for high quality 3D image browsing. We conclude that the proposed scheme is suitable for real time 3D applications.

■ keyword : | 3D Browsing | Image Synthesis | Multiple View | RAM Disk |

* 본 연구는 지식경제부의 대학 IT연구센터 지원사업의 연구결과로 수행되었음. (IITA-2009-(C1090-0902-0047))

접수번호 : #090216-002
접수일자 : 2009년 02월 16일
심사완료일 : 2009년 04월 06일
교신저자 : 심춘보, e-mail : cbsim@sunchon.ac.kr

I. 서 론

3차원 입체 영상 기술은 조금 더 현실적인 화면을 보기 원하는 사용자의 욕구에 의해서 발전해왔으며, 3차원 영상 기술은 1940년에서 1960년까지의 홀로그래피 기술에 그 기원을 두고 있다. 1980년대에 편광안경, 색차 안경을 이용한 입체 영화가 등장하게 되었고, 1980년대에 최초로 개인형 입체 모니터가 등장하게 되었다. 그 이후로는 다인용 입체 TV, 다인용 입체 프로젝터(다중 뷰 방식) 제품 등장, 가상 현실 시스템 구현[1] 및 양안식 3D TV 방송 중계 서비스를 위한 연구를 시작으로 본격적인 3차원 영상 출력 기술에 대한 다양한 연구가 진행되었다. 본 연구에서 3차원 영상은 2차원 영상을 통해 추출한 데이터를 알고리즘을 통해 가공하여 얻은 입체적인 영상을 뜻한다. 또한 3차원 영상 브라우징은 임의의 시점의 영상을 미리 생성하여 보여주는 것이 아니라, 변화하는 2차원 영상을 실시간적으로 복원하여 3D 영상을 보인다.

일반적으로 3차원 영상 출력을 위한 과정은 크게 영상 획득, 획득 영상의 모델링, 인코딩 방식, 전송, 3차원 영상 전환, 3차원 표시 장치의 6가지 과정으로 나뉘며[2][3]. 획득하는 카메라의 방식에 따라서 크게 병렬(parallel) 방식, 컨버전스(convergence) 방식, 짝 컨버전스 방식 등이 있다. 3차원 영상을 생성하는 카메라는 사람의 눈의 역할을 하며 좌 영상과 우 영상의 시차(disparity)를 통해 새로운 3차원 영상을 생성한다. 이 영상을 획득하는 과정에서 높은 대역폭이 필요하기 때문에 대부분의 카메라는 IEEE 1394와 같은 높은 대역폭을 가지는 인터페이스와 연결하여 구현되게 된다[5]. 조금 더 현실적인 화면을 얻기 위해서 오차를 보정한 알고리즘이 사용되거나, 대역폭을 줄이기 위해서 시차 예측을 수행하여 중간 영상을 생성한다[4]. 대량의 대역폭을 차지하는 영상을 바로 사용하는 것은 비효율적이기 때문에 데이터를 압축해서 사용하게 된다. 압축된 데이터를 수신한 후에 중간 영상(IVR, Intermediate View Reconstruction)을 생성한다. 이 때 영상 처리 기술이 사용되며 이 과정을 통해서 시차 정보를 얻을 수 있다. 큰 규모의 영상을 위해서는 서버/클라이언트 구축, 압축 코덱의 구현이 요구된다. 비디오 스트리밍 시스템은 보통 UDP 기반의 RTP 프로토콜을 사용하며, DSS, GPAC, VideoLAN, Apple QuickTime 등이 알려져 있다.

본 논문은 4시점의 병렬 카메라를 통해 실시간으로 입력되는 스테레오 이미지들에 대해서 효율적인 영상 합성을 통해 3차원 입체 영상을 브라우징 할 수 있는 램 디스크 기반의 다시점 영상 합성 기법을 제안한다. 제안하는 기법은 입력 영상들을 이진화 영상으로 변환한 후, Sobel 및 Prewitt 에지 발견 알고리즘을 적용시키고 이를 토대로 4개 클러스터의 영상들의 시차를 구한다. 아울러 기존의 알고리즘에서 모호하게 언급되었던 동기화 문제를 해결하기 위해 하드웨어 트리거와 소프트웨어 트리거를 위한 시간 간격을 적용한다. 또한 제안하는 기법을 분산 환경에서도 적용할 수 있도록 영상의 스냅샷에 대한 유일한 식별자를 이용한다. 성능 분석 결과, 전체 영상(왼쪽, 오른쪽) 및 시차정보를 모두 전송하여 고정밀의 3차원 입체 영상을 출력하는 데 소요되는 전체 시간은 클러스터의 한 영상당 약 0.67초이다. 단일 클러스터가 4개 영상으로 이뤄지는 경우 4채널에서 23.88 FPS를 얻을 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 연구의 배경 및 관련 연구를 살펴보고 3장에서는 제안하는 실시간 다시점 영상 합성 기법을 기술한다. 4장에서는 구현 결과 및 고찰에 대해서 소개하고 마지막으로 5장에서 결론 및 향후 연구를 제시한다.

II. 배경 연구

본 논문과 관련된 배경 연구로 3차원 브라우징 시스템의 구성요소인 카메라 구성 방식과 영상 보정 방법에 대해서 소개한다.

1. 카메라 구성 방식

3차원 브라우징 시스템의 입력 장치에 해당하는 다시점 카메라 구성 방식은 양안 카메라(Stereo Camera) 방식을 사용하며, 이 방식은 크게 교차축 방식과 수평식

방식으로 나뉜다. 먼저, 교차축 방식은 관심물체에 대해 모든 카메라의 광축을 회전시켜 한 점에서 수렴하도록 하고, 각 카메라가 물체로부터 동일한 거리만큼 떨어지도록 설계하며, 카메라를 평행하게 설치하고 관심물체에 수렴점을 형성하기 위해서 카메라의 렌즈의 위치를 수평이동 시키는 방식이다[4]. 교차축 방식의 경우에 영상들 사이에서 오차가 발생하게 되어 왜곡된 입체 영상을 만들기 때문에 반드시 이를 보정해야하기 때문에 일반적인 3차원 콘텐츠 제작에서는 전혀 사용되지 않는다.

이와 달리 수평식 방식은 카메라의 렌즈 위치가 수평이동되어 오차를 수정하기 때문에 영상 왜곡이 적기 때문에 대부분의 3차원 콘텐츠 제작에 사용된다. 그러나 카메라 수가 늘어날수록 넓은 범위를 볼 수 있는 렌즈를 사용해야 한다는 단점이 존재한다. [3]에서는 다중뷰 방식의 3차원 비디오 영상을 표현하기 위해서 주시각 제어를 통한 3차원 합성 알고리즘을 사용한다. 이 방식은 다수의 카메라를 통해 얻은 시차를 적용하여 입체감을 얻을 수 있는 방식이나 전체적으로 시스템 구성 및 촬영, 전송 등에서 효율적이지 못하다.

2. 영상 보정

카메라가 위치한 장소에서 바라보는 시점에 따라서 물체 자체의 색상, 밝기, 균일도가 달라지며, 카메라와의 거리 차이로 인한 왜곡 현상, 카메라 오차, 미세한 크기 차이, 카메라 자체의 밝기, 색상 차이 문제로 인해 일반적으로 카메라로부터 수신한 데이터는 직접 사용할 수 없다. 따라서 이러한 문제를 해결하기 위해 두 가지 방법을 이용한다.

첫 번째 방법은 차이가 발생할 수 있는 단계별로 보정을 적용하고, 최종 영상을 합성하는 방법이다. 이 방법은 정확하게 원하는 데이터를 얻을 수 있는 장점이 있지만, 많은 양의 데이터를 위한 큰 대역폭이 요구되며, 현재 시점이 아닌 이전 시점의 데이터는 필요가 없는 실시간적인 응용에서는 적용하기 어렵다. 그래서 각 단계 중에 최종 영상에 영향을 크게 줄 수 있고, 연산량이 적은 연산만 수행하여 최적화를 수행해야 한다.

두 번째 방법은 중간에 보정 작업을 수행하지 않고, 영상에 대해서 이진화를 수행한 후에 차이값을 계산하고, 이를 각 채널에서 얻은 영상에 적용하는 방법이다. 이는 첫 번째 방법에 비해서 연산량이 거의 없지만, 정확도와 화질 면에서 문제가 생긴다.

III. 제안하는 기법

1. 설계 시 고려사항

본 논문에서는 실시간으로 입력되는 스테레오 이미지들에 대해서 효율적인 영상 합성을 통해 3차원 입체 영상을 브라우징 할 수 있는 램 디스크 기반의 다시점 영상 합성 기법을 위해 다음과 같은 사항을 고려한다.

수평식 방식의 다안 카메라 방식을 사용하나, 카메라 렌즈를 움직이는 방식이 아닌 영상 오프셋 수정 방식을 이용한다.

포인터 연산에 의한 메모리 접근 방식이 아닌 램 디스크(RAM Disk) 기술을 적용시켜 압축된 JPEG 파일을 명시적으로 저장하여 시스템의 안정성을 향상시킨다.

실시간 처리를 위해 최소의 연산으로 빠른 프레임율을 얻고, 오차의 확률을 낮추기 위해 불필요한 왜곡 연산이나, 색상, 밝기 보정 연산을 수행하지 않는다.

2진 2차원 배열을 처리하여 얻은 차이 값을 기준으로 혼합 알고리즘을 적용하여 깊이 정보를 제외한 3D 영상을 복원한다.

출력 장치 외에 별도의 장비가 필요하지 않는 다시점 3D 영상 출력 장치에 획득한 영상을 출력한다.

2. 시스템 구성도

본 논문에서 사용하는 시스템의 구성은 [그림 1]과 같다. 카메라가 지원하는 IEEE 1394 DCAM 1.31 호환 프로토콜을 사용하기 위해 운영체제에서는 IEEE 1394 허브에 접근할 수 있도록 설정한다. 카메라는 수평식으로 지지대에 고정이 되고, 카메라 사이에 약간씩의 차이를 가진 영상을 생성하게 된다. 아울러 허브(Hub)로

부터의 데이터를 임시적으로 보관하기 위해 포인터 연산에 의한 메모리 접근 방식이 아닌 램 디스크(RAM Disk) 기술을 적용시켜 카메라로부터 수신된 압축된 JPEG 파일을 명시적으로 저장하고 사용하도록 함으로써 시스템의 안정성을 향상시킨다. 데이터는 메모리에서 처리된 후에 전체 화면에 사상되어 표준 3D 디스플레이 장치에 브라우징된다.

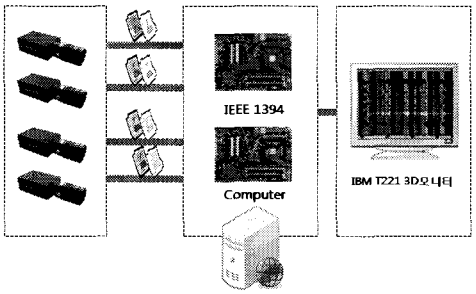

그림 1. 전체 시스템 구성도

3. 카메라 상호작용

IEEE 1394 인터페이스를 지원하는 마이크로프로세서 내장형 카메라를 사용하기 때문에 별도의 조작 없이 명령을 전달할 수 있다. 동시 촬영을 위해서 외부 트리거(External Trigger)를 작동(enable) 시키고, 소프트웨어에서 이를 제어하기 위해서 주기적인 소프트웨어 트리거(Software Trigger)를 사용한다. 만약 트리거를 주지 않고 사용하게 되면, 각 카메라마다 다른 동기화 클럭을 가지게 되기 때문에 영상은 현실적으로 보이더라도 영상 처리를 위한 동기화가 되지 않아 영상 처리 후 영상이 손실되거나 많은 노이즈가 발생한다. 카메라는 IEEE 1394 DCAM 1.31 호환 프로토콜을 사용하여 구동하며, 이중 Format7 Mode0 모드에서 최대 대역폭으로 하여 데이터 수신을 하게 된다. 그러나 시스템 구현 시 하드웨어 동기와 소프트웨어 동기간 속도 차이가 매우 심하기 때문에 적절한 영상 처리 시간을 감안하여 동작 모드(Mode)는 동일하되, 소프트웨어 트리거에서 일정 시간 간격을 벗어나지 못한 신호에 대해서는 폐기(discard) 연산을 한다.

카메라는 외부 PC에 허브를 추가하고 케이블을 연결을 함으로써 시스템에 추가될 수 있도록 하였다. 사용하는 IEEE 1394 인터페이스의 대역폭이 400 Mbps 이므로, 시스템 순수 대역폭은 640*480 크기의 RGB 데이터에 대해서 초당 프레임 수가 30 FPS일 경우 식(1)에 의해 843.75 Mbps가 된다.

최대 대역폭 = 픽셀수 * 픽셀당 비트수 * 채널수 * 초당 프레임 수 ... (식 1)

한 채널당 211 Mbps 정도의 대역폭이 필요한 것을 알 수 있다. 그러나 최대 대역폭으로 시스템을 그대로 사용할 경우 처리해야할 데이터가 너무 많고, 카메라도 겨우 2대 밖에 운용할 수가 없게 된다.

4. 3D 영상 생성 기법

제안하는 3D 영상 생성 과정은 [그림 2]와 같다. 먼저 4채널 다중 카메라로부터 얻은 영상 데이터에 대해서 전송량을 줄이기 위해 JPEG으로 압축한 후, 수신 측에서는 다시 비트맵 형태의 이미지로 복원을 한다. 채널별 영상을 합성하기 위해서 채널별 영상에 제안하는 영상 합성 알고리즘을 적용한 뒤 최종적으로 3D 출력 장치로 전송하여 실시간으로 브라우징한다.

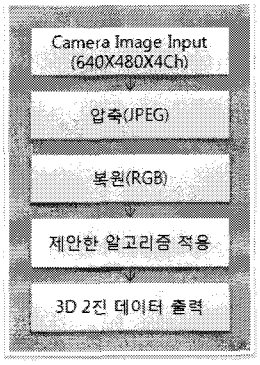

그림 2. 3D 영상 생성 과정

전체적으로 3D 영상 합성 알고리즘은 [그림 3]과 같은 과정을 거친다. 영상 보정 두 번째 방식에서 언급했던 이진화를 수행한 뒤에 Sobel 또는 Prewitt 등의 에지

발견 알고리즘을 적용한다. 그 후, 발견된 에지를 통해서 시차(disparity)를 계산한다. 시차는 역 핀홀(pin-hole) 카메라 모델에서 가상의 3차원 점을 핀홀로 하고, 이를 현실 세계의 카메라 상의 점 사이에서 가상의 투영면을 생성하여 이 투영면에 생긴 스테레오 쌍의 오른쪽과 왼쪽 영상의 일치점들의 위치 차이를 의미한다. 수평식 배열을 통해서 카메라는 동일한 일치점에 대한 시차를 가진다. 시차를 얻은 후 영상 합성을 위해, 가상 세계 좌표에 맞게 영상을 지리적으로 이동시킨다. 지리적으로 이동을 시킨 영상은 특정 채널의 가시 영역의 차이로 인해 생기게 되는 폐쇄 영역(occlusion)을 영상으로부터 제거한다. 마지막으로 4채널 다시점 카메라로부터 얻은 영상을 합성하여 3차원 공간의 백 버퍼(back buffer)로 보낸다.

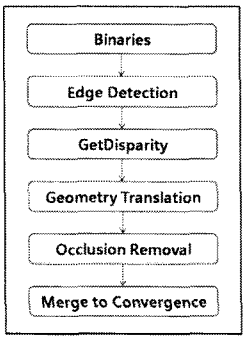

그림 3. 3D 영상 합성 과정

1.1 전처리(Preprocess) 단계

카메라로부터의 데이터는 IEEE 1394 허브를 통해서 400 Mbps의 속도로 램 디스크로 전송이 되고 CPU에서 처리되게 된다. 그러나 CPU에 연산 데이터를 넘기는 과정에서 소모되는 비용은 0이라고 가정하면, 메모리 전송량은 식(2)와 같이 계산할 수 있다.

초당 전송량 = Frequency * Data Rate * Bus Size ... (식 2)

본 논문에서 사용한 DDR 333 메모리의 경우 Frequency가 166 Mhz이며, Data Rate 값은 2이며, 64 bit 버스를 사용하므로 21,248 Mbps의 전송량을 가진다. 최대 대역폭이 843.75 Mbps이며, 제한 프레임 비율인 14.2 FPS를 적용할 경우에는 4 채널 데이터에서 400 Mbps 대역폭을 사용하기 때문에 메모리로 데이터 전송 시에 발생하는 지연문제는 없다. 단지 시간이 많이 소요되는 부분은 다시점 카메라로부터 입력되는 영상을 3D 영상으로 합성하는 부분이다.

1.2 이진화(Binarize) 단계

이진화 단계에서는 카메라로부터 받은 영상에서 경계 검출 알고리즘을 적용하기 위해서 이를 0과 1로 이뤄진 이진 배열로 변환한다. 램 디스크(RAM Disk)에 저장된 원 영상을 기준으로 원본, R, G, B의 4개의 배열을 생성할 수 있지만 보통 RGB 중 한 가지 값을 선택하여 배열을 생성한다. 이 단계에 생성된 2진 2차원 배열은 알고리즘의 다른 단계에서 기본 데이터로 사용된다.

1.3 경계 검출(Edge Detection) 단계

경계 검출 단계에서는 영상 안에 있는 객체의 모서리를 조금 더 두드러지게 하여 영상 안의 좌표를 더욱 명확하게 구분할 수 있게 한다. 이 단계에서는 영상의 크기 자체는 변화되지 않기 때문에 성능 최적화는 어떤 알고리즘을 적용하느냐에 따라 달라질 수 있다. 본 논문에서는 Sobel 알고리즘과 Prewitt 알고리즘 2가지를 적용한다.

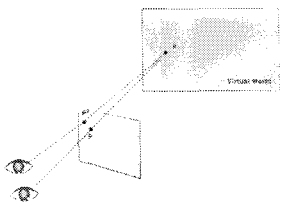

그림 4. 시차(Disparity) 개념

1.4 시차(Disparity) 계산 단계

역 pin-hole 모델에서 보이듯이 최종 3D 화면은 가상 세계에 존재하는 영상이다. 가상 세계의 특정 좌표는

현실 세계에서 물리적인 시차(Disparity)만큼 떨어진 위치의 좌표를 통해 표시된다. 따라서 가상 세계의 3D 화면을 완성하기 위해서는 시차값을 계산해야 한다. 시차를 얻기 위해서 이진화된 영상에서 유사 영역을 찾은 후, 유사 영역에서 좌표를 가상 세계 좌표와 비교를 통해 상대값을 가지고 있어야 한다. 그러나 이런 경우에 채널수가 4개라 하더라도 첫 채널과 마지막 채널 사이의 시차값의 차이가 많이 나서 알고리즘 적용의 오차가 커질 수 있다. 이러한 오차를 줄이기 위해 본 논문에서는 현재 영상과 직전 영상에 누적된 시차를 계산한다. 시차를 얻는 것은 가상 세계 좌표의 동일한 영역을 나타내는 지점의 투사 영상의 좌표가 각 채널 영상에서 어떤 좌표를 가지고 있는가를 구하는 것이다. 이는 [그림 5]와 같은 세부적인 단계를 통해서 이루어진다.

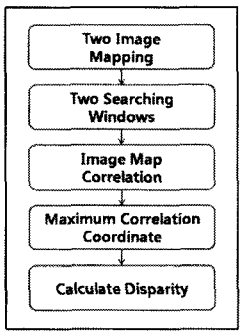

그림 5. 시차 계산 처리 과정

먼저 원본 영상과 목적 영상의 이진화된 영상을 순회할 수 있는 마스크(최대 64*64)를 생성하고 이를 통해 무한정한 순회를 막기 위해 특정 크기의 검색 윈도우를 2개 생성한다. 하나는 원본 영상에 대한 윈도우이며, 다른 것은 목적 영상에 대한 윈도우이다. 각 윈도우의 크기는 동일하다. 이 후에 각 윈도우는 목적영상과 원본 영상의 동일한 지점을 이동하면서 Correlation이라는 값을 영상 맵으로부터 얻어내게 된다. 이 때 검색 범위는 다음과 같다.

(0, 0) ~ (Image Map의 넓이 - 검색 윈도우의 넓이, Image Map의 높이 - 검색 윈도우의 높이)

Correlation 값은 윈도우 영상의 현재 포인터 값이 유클리디언 거리 차가 얼마인지 계산하며 식(3)과 같으며, SRC(x, y)와 DST(x, y)는 각각 원본 영상과 목적 영상의 x축, y축 좌표이다.

$$\text{Correlation} = S_1/(\text{SQRT}(S_2) * \text{SQRT}(S_3)) \quad \text{... (식 3)}$$

$$S_1 = \text{SRC}(x, y) * \text{DST}(x, y)$$
$$S_2 = \text{SRC}(x, y) * \text{SRC}(x, y)$$
$$S_3 = \text{DST}(x, y) * \text{DST}(x, y)$$

위의 과정을 거쳐 가장 큰 값을 가지는 Correlation 값이 가상 세계가 아닌 각 채널의 물리 세계에서 투영된 영상의 절대 좌표를 가지게 된다. 가상 세계의 좌표를 얻기 위해서 최종적으로 역으로 뒤집힌 영상의 좌표를 고려하여 MINUS 연산을 수행한다. 이를 통해 최종적인 시차값을 구한다. 누적 시차값 계산은 식(4)와 같다.

$$d(x,y) = window(x,y) - \max\left(\sum_{n=0}^{N} correlation(n, n-1)\right) \quad \text{... (식 4)}$$

window는 검색 윈도우 시작점이며, 윈도우 시작점이 영상이 뒤집힌 형태로 되어 있기 때문에 이를 계산하며, Correlation은 식(3)에 의해 계산되고, n은 채널 번호, N은 전체 채널 수 이다.

1.5 기하 변환(Geometry Translation) 단계

이 단계는 각 채널의 영상을 가상 세계 상의 수렴하는 좌표 위치에 매핑하기 위해 이전 단계에서 구한 시차값을 이용하여 영상을 옮기는 단계이다. 각 영상은 이전 영상의 시차값을 누적하여 가지고 있기 때문에 가상 세계의 좌표계는 0번 영상을 기준으로 생성하게 된다. 현재 영상의 특정 점의 가상 세계 좌표는 식(5)와 같이 계산된다.

$$p(x,y,ch) = d(ch, ch-1) + w \cdot y + mc \cdot x + c \quad \text{... (식 5)}$$

여기서 x, y는 각각 x좌표와 y좌표를 나타내며, ch는 현재 채널, d()는 누적 시차값, w는 영상의 너비, mc는 전체 색상 수, 그리고 c는 영상의 한 색상을 나타낸다. 즉 식(5)를 통해 영상은 가상 세계의 좌표를 기준으로 정렬되게 되며, 원본 영상은 변하지 않고 복사본만이 이동하게 된다.

1.6 폐쇄 영역 제거(Occlusion Removal) 단계

이전 단계에서 가상 세계 좌표를 기준으로 영상을 이동 시켰기 때문에 영상의 상하 좌우에 각 채널 별로 보이지 않는 영역에 대해서 폐쇄(Occlusion) 현상이 나타나게 된다. 즉, 단순히 시작 좌표부터 합성을 시작하게 되면 최종 영상의 상하 좌우 양 끝에는 영상 데이터가 없는 형태로 합성이 된다. 이러한 폐쇄 영역을 제거하기 위해 가상 세계 좌표를 기준으로 영상을 잘라내야 한다. 또한 잘라진 영상의 크기는 각 채널 당 초기 예상 크기인 640*480에 크게 못 미치기 때문에 이를 720*800의 크기로 Resample 한다. 잘라낼 영역을 계산하기 위해 식(6)을 사용한다.

$$c(t,l) = \max\left(\sum_{n=0}^{N} d(x,y)\right) \qquad \text{... (식 6)}$$

여기서 t는 위쪽 좌표, l은 왼쪽 좌표, n은 채널, N은 전체 채널수이며, d는 누적 시차값이다.

1.7 병합(Merge to Convergence) 단계

폐쇄 영역을 제거한 영상은 가상 세계의 3D 영상으로 최종적으로 병합되어야 한다. 이미 Resampling이 된 후의 영상이기 때문에 별도의 처리 없이 바로 영상을 사용할 수 있지만, 최종 3D 영상은 105.47 MB의 크기를 가지고, 병합 시점에 한 채널의 데이터가 26.38 MB정도 되기 때문에 총 131.84 MB를 사용하고 대역폭은 633 Mbps이다. 이는 식(2)에서와 같이 메모리의 전송량의 절반에 해당하는 데이터로 이 부분에서 가장 큰 병목 현상이 발생하게 된다. [그림 6]은 3차원 표시장치에서 영상을 생성하는 Spatial Multiplexing 알고리즘을 나타낸다.

```
Algorithm Merge

for (h=0; h < H; ++h){
    S.y = h/colors;
    if(0 = (h % channels)){
        matchCount =
        (++matchCount)% colors;
    }
    for(w=0; w < W; ++w){
        pixelPos =
        (w*colors) + shift;
        channel =
        pixelPos % channels;
        S.xB =
        pixelPos/channels+1;
        S.xG =
        (pixelPos+1)/channels+1;
        S.xR =
            (pixelPos+2)/channels+1;
        if(matchLine[matchCount] >
        shift){
            --S.xB; --S.xG; --S.xR;
        }
        shift = (++shift)%channels;
        Invert(w, sourceImage(S));
    }
}
```

그림 6. Spatial Multiplexing 알고리즘

메모리상에서 고정된 공간에 있는 목적 영상의 위치로 데이터를 복사할 때 영상이 뒤집어진 것처럼 복사가 되기 때문에 이를 위해서 별도의 상하/좌우 반전을 수행해야 한다. 위의 알고리즘은 채널 개수와 색상 수에 기반해서 영상을 병합하게 된다. 최종적으로 표준 디스플레이 장치로 데이터를 전송하게 됨녀 모든 알고리즘이 완료된다.

Ⅳ. 구현 결과

1. 구현 환경

제안하는 시스템은 Windows XP에서 .NET Visual Studio 2005를 사용하여 C++ 및 Windows 32 API와

MFC를 이용하여 구현하였다. 아울러 3D 출력 장치는 720*800*4의 해상도와 13Hz의 주파수를 가지고 있는 IBM T221를 사용했으며, 4개의 IMITech의 IMC-40FT 카메라를 사용했다. 시스템에서 사용하는 데이터는 하드 디스크가 아닌 램 디스크(RAM Disk)에 저장되며, RAM은 DDR 333 메모리를 사용했으며, Frequency가 166Mhz, 2의 Data Rate, 64비트의 버스를 가지고 있다. [그림 7]은 IEEE 1394 호스트 카드를 이용하여 한 PC에 4개의 카메라를 구성한 것이다.

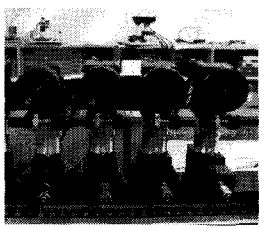

그림 7. 영상 카메라(IMC-40FT)

[그림 8]은 4개의 카메라로부터 동시에 획득한 샘플 영상이다.

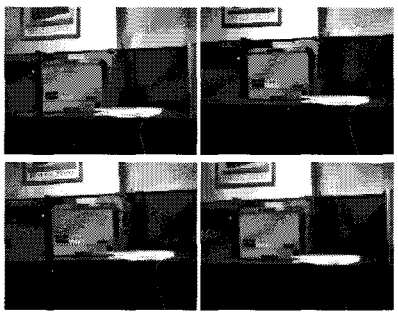

그림 8. 4개의 카메라에서 획득한 샘플 영상

본 논문에서는 [그림 7]의 4채널 영상을 아래와 같은 형태의 파일명으로 저장한다.

SEQ(기계번호-채널번호)[Disparity].jpg

여기서 Disparity는 x, y 순이다. 위의 영상 자체는 변경되지 않으며 램 디스크 상에 상주하게 된다. SEQ가 있는 이유는 FPS가 높아질수록 동일한 영역에 접근할 확률이 높아지기 때문에 동기화 문제를 해결하고 한 영상의 클러스터 버전을 일치하여 원자성을 보장하기 위함이다. 즉, SEQ를 사용해서 이미지를 구분하게 된다. 기계 번호는 단일 컴퓨터에서 처리할 수 있는 영상의 한계가 있기 때문에 다중의 컴퓨터를 연결할 때 사용하기 위해서 존재한다.

2. 구현 결과 및 고찰

[그림 9]는 본 시스템에서 구현한 사용자 인터페이스(GUI)이다. 먼저 [그림 9]의 ①부분은 IOCP(IO Completion Port)를 통해서 획득한 영상 중 가장 SEQ의 값이 큰 베이어(Bayer) 형태의 파일을 클러스터 수만큼 반복하여 램 디스크로부터 읽어 들여 화면에 출력한 것이다. ②부분은 ①의 베이어 형태의 영상을 24비트 RGB 형태로 변환하여 화면에 출력한 것이다. ③부분은 ②의 비트맵 영상을 이진화 영상으로 변환하여 출력한 것이다. ④부분은 ③의 이진화 영상에서 시차를 계산하기 위해 경계(Edge) 부분을 강조하고 ⑧부분에서 선택한 경계 검출 알고리즘을 사용하여 출력한 것이다. ⑤부분은 ④의 경계 검출 영상으로부터 일정 크기의 마스크를 사용하여 정해진 영역의 윈도우 상에서 이동하면서 ⑧부분에서 선택한 시차 계산 옵션에 따라 시차를 추출하여 계산하고 계산된 시차만큼 영상을 이동시킨 것이다. ⑥부분은 ⑤에서 생성된 영상에서 보이지 않는 영역을 제거한 후에 잘려진 크기를 보정하기 위해 영상을 리샘플링(Resampling)하여 출력한 것이다. ⑦부분은 영상을 요청하고 영상 요청을 중지 및 혼합된 영상을 전체화면으로 보기 위한 명령 버튼들이다. ⑧부분은 경계 검출을 위한 경계 검출 알고리즘들과 시차 계산을 위해 사용하는 영상의 종류를 선택하는 부분이다. Bayer의 경우는 ①번, Edge의 경우는 ④번, 그리고 Binary의 경우는 ③의 영상들을 기준으로 시차가 계산된다. 마지막으로 ⑨부분은 현재 프레임 속도 및 처리된 영상의 수에 대한 정보와 현재 클러스터 내의 각 영상의 시차를 나타낸 것이다.

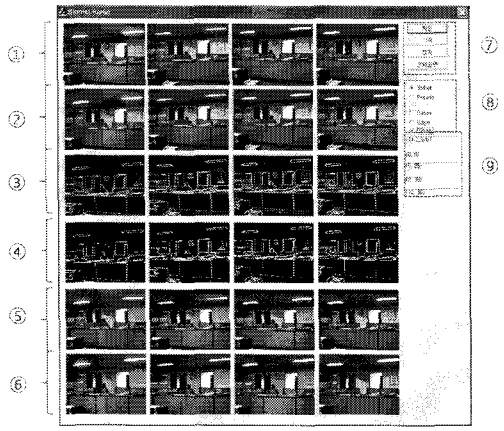

그림 9. 사용자 인터페이스(GUI)

[그림 10]은 각각 3차원 영상 데이터를 일반 2D 모니터와 3D 모니터인 IBM T221 모니터를 출력한 결과를 보인다. 3D 모니터로 출력했을 경우 일반 2D 모니터에 비해 훨씬 더 입체적인 영상을 볼 수 있다.

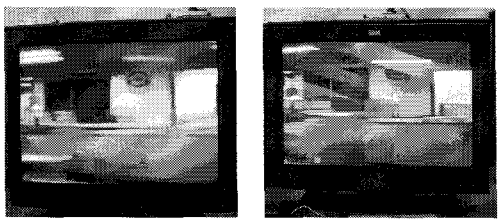

그림 10. 2D 모니터(왼쪽)와 3D 모니터(오른쪽) 출력 결과

[그림 11]은 3차원 입체 영상을 생성하기 위해 각 알고리즘 단계에서 소요되는 시간을 측정한 것이다. 전체 알고리즘 중에서 가장 많은 시간을 차지하는 것은 시차를 계산하는 단계이다. 평균적으로 전체 시간의 약 70%를 차지한다. 시차를 계산하는 시간은 Sobel의 경우 0.55초, Prewitt의 경우는 0.54초가 소요된다. 따라서 시차를 계산하는 시간을 포함한 전체 소요 시간은 Sobel의 경우 0.68초, Prewitt의 경우 0.67초 소요됨을 알 수 있다. 본 논문이 높은 성능을 얻을 수 있는 이유는 다음과 같다. IOCP를 이용한 고성능의 네트워크 기반의 처리는 영상 처리가 분산될 수 있도록 한다. 즉 단일 클러스터의 크기가 4인 경우 4개의 클러스터를 운용할 경우 프레임율은 영상 전송 속도가 최소화 되었을 경우 16 / 0.67 = 23.88 FPS이다. 프레임율은 클러스터에 포함되는 영상의 수가 증가할수록 커지게 되며 더욱 현실적인 영상을 얻을 수 있다. 다만 현재의 설정에서는 5 프레임 정도의 손실을 얻었으며 18.3 FPS가 최대 프레임 율이었다.

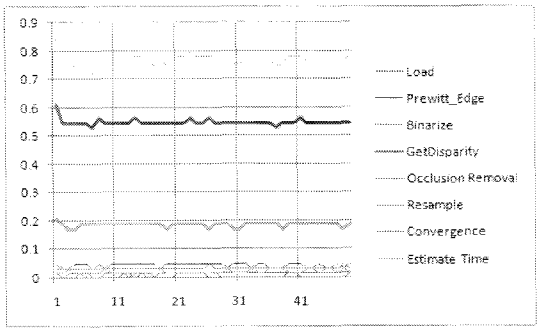

그림 11. 각 알고리즘 단계별 측정 시간(sec.)

[표 1]은 기존 연구와 비교 결과이다. 전체적으로 카메라 구성 방식, 채널 수, 해상도, 실시간 등의 여부 측면에서 비교한 것이다. 여기서 가장 중요한 요소인 영상의 프레임 처리 수(FPS)의 경우, 기존 알고리즘에서 영상의 프레임 수를 계산 표시 할 때, 정지 영상을 이용하는 경우가 있기 때문에 [표 1]의 FPS는 정확하다고 할 수 없다. 특히 시차를 계산할 때, 정지 영상의 경우 시차값을 빠르게 계산할 수 있어 프레임 비율이 높을 수 있다.

표 1. 기존 연구와 비교 결과

	[3]의 연구	[8]의 연구	[9]의 연구	본 연구
카메라방식	Multi view + Convergence	binocular parallax	Multi view + Convergence	Multi view Parallel + Convergence
FPS	12.5	1이하	2.5	18.3
채널 수	2	2	6	4
3D 표시 장치	×	○	×	○
해상도	352X288	Unknown	640X480	640X480
깊이 테스트	×	○	○	×
실시간여부	○	×	×	○

V. 결론 및 향후 연구

본 논문은 3차원 입체 브라우징 시스템을 위해 IEEE 1394 인터페이스를 이용하여 4개의 병렬 카메라를 통해 램 디스크 기반의 실시간 다시점 영상 합성 기법을 제안하였다. 제안하는 기법은 전처리 단계, 이진화 단계, 경계 검출 단계, 시차 계산 단계, 기하 변환 단계, 폐쇄 영역 제거 단계, 마지막으로 병합 단계를 거쳐 IBM T221 3D 모니터를 통해 3차원 입체 영상을 출력한다. 기존의 알고리즘에서 모호하게 언급되었던 동기화 문제를 해결하기 위해 하드웨어 트리거와 소프트웨어 트리거를 위한 시간 간격을 적용하고 분산 환경에서도 적용할 수 있도록 영상의 스냅샷에 대한 유일한 식별자를 이용하였다. 또한 사용자의 편의성을 도모하기 위한 사용자 인터페이스를 구현하였다. 성능 분석 결과, 각 알고리즘 단계 중에서도 특히 시차를 계산하는 부분이 가장 많은 시간이 소요되고 있으며 전체 영상 및 시차정보를 모두 전송하여 3차원 입체 영상을 출력하는 데 소요되는 시간은 단일 배열 당 약 0.67초 정도로 실시간 측면을 고려할 때 매우 좋은 성능을 보임을 확인했다. 또한 시스템의 구현 시작 때의 예상과 달리 Sobel과 Prewitt 사이에서 큰 시간 차이가 발생하지 않았기 때문에 차후 연구에서는 에지 처리 알고리즘에 큰 비중을 두지 않을 것이다.

향후 연구로는 보다 더 현실적인 3D 입체 영상을 얻기 위해서는 영상 내에 있는 객체들의 깊이 정보를 계산하여 관찰자로부터의 거리를 고려하여 이미지 확대/축소 및 음영 부분까지도 세심하게 조절할 수 있는 연구가 요구된다.

참 고 문 헌

[1] 오길록, *3차원 입체영상(3DTV) 방송중계 시범서비스*, 정보통신부, 2002.

[2] 조숙희, 윤국진, 서정일, 안충현, 이수인, "지상파 DMB 기반 3차원 AV 서비스 시스템", Telecommunication Review, 제14권, 제4호, pp.652-665, 2004.

[3] J.-R. Ohm and K. Müller, "Incomplete 3D - Multiview representation of video objects," IEEE Transaction on Circuits and Systems for Video Technology, special issue on SNHC, pp.389-400, 1999.

[4] 이광순, 김형남, 허남호, 안충현, "평행축 스테레오 카메라에서의 주시각 제어 방법", 한국방송공학회 학술대회 논문집, pp.297-301, 2002.

[5] O. Schreer, N. Brandenburg, and P. Kauff, "Real-Time Disparity Analysis for Applications in Immersive Tele-Conference Scenarios - A Comparative Study," Proc. of Int'l Conf. on ICIAP 01, pp.346-351, 2001.

[6] S. Laveau and O. D. Faugeras, "3-D Scene Representation as a Collection of Images," Proc. of Int'l. Conf. on Pattern Recognition, pp.689-691, 1994(revised on 2002).

[7] H. Klaus, "An Autostereoscopic Display Providing Comfortable Viewing Conditions and a High Degree of Telepresence," IEEE Ttrans. on Circuits and Systems for Video Technology, Vol.10, No.3, pp.359-365, 2000.

[8] S. Kiyohide, K. Itaru, and O. Yuichi, "3D Image Display with Motion Parallax by Camera Matrix Stereo," IEEE Proc. of MULTIMEDIA '96, pp.349-357, 1996.

[9] 허경무, 박영빈, "A Viewpint-Dependent Autostereoscopic 3D Display Method," Proc. of ISIE 2001, 2001.

[10] Ioannis Pitas, *Digital Image Processing Algorithms and applications*, Wiley-IEEE, 2000.

저 자 소 개

심 춘 보(Chun-Bo Sim) 정회원

- 1996년 2월 : 전북대학교 컴퓨터공학과(공학사)
- 1998년 2월 : 전북대학교 컴퓨터공학과(공학석사)
- 2003년 2월 : 전북대학교 컴퓨터공학과(공학박사)
- 2005년 2월 ~ 현재 : 순천대학교 정보통신공학부 조교수

<관심분야> : 멀티미디어 데이터베이스 & 정보검색, 유비쿼터스 컴퓨팅, 멀티미디어시스템

임 은 천(Eun-Cheon Lim) 준회원

- 2007년 2월 : 순천대학교 정보통신공학부(공학사)
- 2009년 2월 : 순천대학교 멀티미디어공학과(공학석사)

<관심분야> : 멀티미디어 데이터베이스 & 정보검색, RFID/USN, 멀티미디어시스템