JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 18, No. 2, pp. 383-392, Apr. 2017

다시점 영상을 이용한 실시간 3D 모델 생성 시스템

박정선[1*] · 손형재[1] · 박정철[2] · 오일석[3]
[1*]코난테크놀로지 코난링크사업부
[2]한국전자통신연구원 기업현장지원실
[3]전북대학교 컴퓨터공학부

Real-time 3D model generation system using multi-view images

Jeong-Sun Park[1*] · Hyung-Jae Son[1] · Jeung-Chul Park[2] · Il-Seok Oh[3]
[1*]Department of KonanLink, KONAN Technology, Seoul 06627, Korea
[2]SMEs Support Section, Electronics and Telecommunications Research Institute (ETRI), Daejeon 34129, Korea
[3]Division of Computer Science and Engineering, Chonbuk National University, Jeonju 54896, Korea

[요 약]

본 논문은 다시점 영상 획득에서 영상 기반 3D 모델 생성까지 실시간으로 처리가 가능한 실시간 3D 모델 생성 시스템을 소개한다. 이 시스템은 18대의 카메라에서 입력되는 HD급 영상을 수집, 전송, 관리하는 방법을 소개하며, 전경과 배경의 분리와 부드러운 3D 볼륨 모델 생성 과정을 설명한다. 이 논문은 18대의 카메라에서 입력되는 HD급 영상을 실시간으로 처리하기 위한 새로운 분산 데이터 송수신 및 관리 방법을 제안한다. 또한, 다시점 영상으로부터 부드러운 3D 모델을 생성하기 위한 시각 차이를 이용한 코드북 기반 전경과 배경 분리 알고리즘, 원근 보정 보간법을 이용한 수정된 마칭 큐브 알고리즘을 기술한다. 이 시스템은 현재 초당 30프레임 처리 속도로 구축되어 있다.

[Abstract]

This paper introduces a real-time 3D model generation system that can process in real time from multi-view image acquisition to image-based 3D model generation. This system describes how to collect, transmit, and manage the HD images input from 18 cameras and explain the background separation and smooth 3D volume model generation process. This paper proposes a new distributed data transmission and reception method for real-time processing of HD images input from 18 cameras. In addition, we describe a codebook-based background separating algorithm and a modified marching cube algorithm using perspective difference interpolation to generate smooth 3D models from multi-view images. The system is currently being built with a throughput rate of 30 frames per second.

색인어 : 분산 데이터 수집, 전경과 배경 분리, 3D 모델링, 볼륨 모델

Key word : Distributed data collection, Background subtraction, 3D modeling, Volume model

http://dx.doi.org/10.9728/dcs.2017.18.2.383

Received 20 March 2017; **Revised** 10 April 2017
Accepted 25 April 2017

***Corresponding Author; Jeong-Sun Park**

Tel: +82-02-3469-8730
E-mail: jspark@konantech.com

Ⅰ. 서 론

3D 콘텐츠는 주로 게임에 적용되었지만, 지금은 영화뿐 아니라 방송 및 광고 분야에서도 널리 적용되고 있다. 2D 효과보다 3D 효과는 더욱더 사실적이며 다양한 효과를 낼 수 있으므로 여러 분야로 퍼지고 있다. 2009년 아바타라는 영화가 개봉된 이후로 일반 시청자도 관심을 많이 두게 되었다. 아바타는 안경을 쓰고 입체 효과를 느끼는 입체 영상을 보여주는 영화로써 양안에 시점이 다른 영상을 보여줌으로써 깊이감을 느끼게 하여 3차원적인 영상을 체험할 수 있게 하였다. 하지만 이런 응용에서는 3차원 그래픽 프로그램을 이용하여 애니메이션을 미리 제작해 두어야 하므로 정적인 콘텐츠에 머무는 한계를 안고 있다.

현실 세계에서 일어나는 동적인 장면을 3차원으로 모델링한 다음, 단말에서는 사용자가 지정한 시점에 따라 실시간으로 렌더링하는 작업은 많은 응용이 있다. 가령 축구장에 수십 대의 카메라를 설치하여 실시간으로 다중 영상을 취득하고, 이들 영상으로부터 3차원 장면을 재구성한 다음 사용자 단말로 보내 사용자의 지시에 따라 다양한 시점으로 디스플레이하는 응용을 생각해 볼 수 있다 [1, 2]. 하지만 실시간으로 이러한 시스템을 구축하는 일은 매우 어려운 일에 속하며, 다중 영상을 획득, 전송, 관리하는 과업, 3차원 재구성 과업, 실시간 렌더링 과업 등이 어우러져야만 가능하다.

이 논문은 최근 개발되어 운용이 가능한 시스템을 소개한다. 이 시스템은 초당 30프레임을 처리할 수 있는 수준으로 개발되어 있다. 단 3차원 장면 재구성 단계의 기술적인 한계로 인해 일정한 배경을 가진 환경으로 국한하여 개발되었다.

2차원으로 이러한 3차원적인 영상을 체험할 수 있으려면 양안 시점 영상을 생성하여야 하는데 이를 위해서 양안 시점 영상에 대한 깊이 영상을 생성하고 깊이 값에 따라서 시청자가 깊이감을 느끼게 하는 것이다. 일반적으로 깊이맵 이용하여 영상을 생성하고 3차원 비디오를 생성하면 3차원적인 느낌이 들지 않는 경우가 있는데 깊이맵 조절을 통한 편안한 3차원 비디오 생성 방법이 제시되어 있다 [3]. 사용자가 원하는 깊이감을 표현하기 위하여 깊이맵을 재생성하는데 일반적으로 선형 방법을 통해서 생성하지만 양안 시차의 불일치로 인한 어지러움이 발생한다. 이를 해결하기 위해서 비선형 방법을 적용한 논문도 있다 [4]. 3D 라이트 필드로부터 생성된 다중 원근 컷(multi-perspective cut)을 이용한 깊이 재생성(disparity remapping)을 통한 양안 영상 생성하는 방법도 있다 [5]. 양안 영상이 아닌 단일 영상으로부터 깊이 영상을 예측하는 방법도 제안되었다 [6]. 깊이맵을 통한 양안 시점을 생성하는 방법뿐 아니라 다중의 깊이맵을 이용하여 3D 모델링을 하는 방법도 제안되었다 [7].

이처럼 깊이맵을 통한 3D 모델링 방법은 깊이맵을 생성하는 중간 단계 과정을 거쳐야 하므로 많은 시간이 필요로 하는 문제가 있다. 본 논문은 깊이맵을 생성하지 않고 바로 3D 모델 생성 방법을 제안한다.

시스템의 앞 단에서는 18대의 카메라들에서 출력되는 HD급 영상을 수집, 병합하고 3D 모델 생성까지 처리할 수 있는 장비가 필요하다. 하지만 이들을 모두 처리할 수 있는 장비란 존재하지 않는다. 그 이유는 수십 개의 인코딩 보드 및 GPU 카드 등을 한 장비에 설치할 수 없기 때문이다. 따라서 다양한 기능들을 다수의 장비에서 분할되어 처리되어야 하는데, 이를 위해서는 장비 간에 데이터가 공유되어야 한다. 이때 장비 간의 가장 간단한 데이터 공유 방법으로는 하나의 장비에서는 파일을 쓰고, 다른 장비에서는 파일을 읽어서 처리하는 방법이다. 하지만, 이는 디스크 I/O에 상당한 시간이 소요되므로, 단순 공유가 아닌 실시간 처리에 적용하기에는 부적합하다. 이런 이유로 디스크 I/O에 따른 시간이 소요되지 않는 실시간 처리를 위한 새로운 전송 및 데이터 관리 방법을 제안한다.

2장에서는 다시점 영상 획득에서 3D 모델 생성을 위한 시스템 구성도 및 다시점 영상 획득 방법을 제시하고 기존의 연속적인 배경 모델 학습을 이용한 코드북 기반의 전경 추출 알고리즘 [8]과 달리 정밀한 객체 추출을 위해 시각 차이를 이용한 코드북 기반 전경과 배경 분리 알고리즘에 관해서 설명한다.

실시간 처리를 위한 전체 시스템에서 그 규모와 취급하는 데이터의 크기가 커질수록 콘텐츠 생성 중 일시적으로 발생할 수 있는 병목 현상이나 처리 지연에 강인하게 설계해야 한다. 이를 위해 장비 간 분산 처리된 데이터를 수집하고 운용하기 위한 전송 및 데이터 관리 방법은 3장에서 자세히 설명한다.

4장에서는 2장에서 설명한 시각 차이를 이용한 코드북 기반 전경과 배경 분리 데이터를 이용하여 볼륨 모델을 생성하고 원근 보정 보간법을 이용한 메쉬 모델을 생성하며 마지막으로 텍스쳐를 입히는 과정에 관하여 설명한다.

Ⅱ. 다시점 영상 획득 및 전처리

2-1 다시점 영상 획득 시스템 구성

그림 1은 3D 모델 생성을 위하여 다수의 카메라로부터 생성된 HD급 영상을 획득하여 3D 모델 생성 서버로 보내기 위한 구성도를 보여준다.

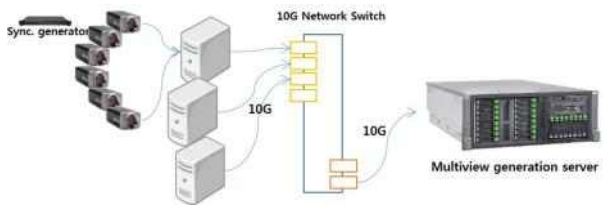


그림 1. 다시점 영상 획득 시스템 구성도
Fig. 1. A system design of multi-view image aquisition

이 시스템 구성은 다수의 카메라를 동기화시켜주는 장치 1대, 6대 카메라의 영상을 획득하는 워크스테이션(이하 노드) 3대와 18개 카메라 영상을 이용하여 3D 모델을 생성하는 서버로 구성되어 있다. 그리고 대용량 영상 데이터를 실시간 처리를 위한 전송을 위해 10G 네트워크를 사용하였다.

노드에서는 각 카메라에서 획득된 영상에 대해 시각 차이를 이용한 코드북 기반 전경과 배경 분리 알고리즘을 적용하여 객체를 분리한다. 이 알고리즘은 2-3절에서 자세히 설명한다.

하나의 노드는 6대의 카메라에서 생성된 영상과 전경과 배경 분리 영상을 3D 모델 생성 서버로 전송한다. 3D 모델 생성 서버에서는 3대의 노드에서 수신된 영상에 대해 프레임 단위로의 동기화 작업 및 실시간 처리를 위한 관리가 필요하며 이 방법은 3장에서 자세히 설명한다.

그림 2는 18대 카메라의 배치를 보여준다.

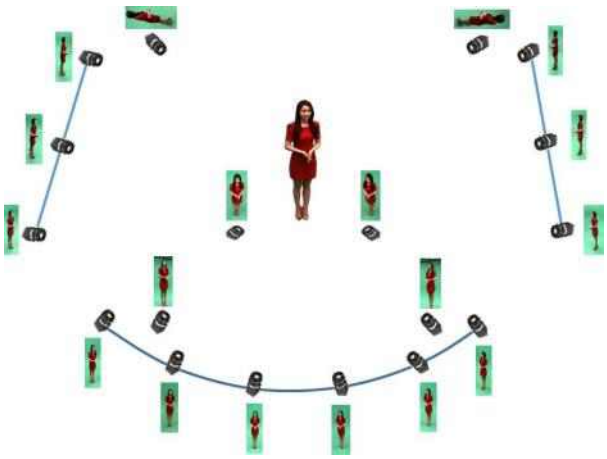


그림 2. 다시점 영상 카메라 배치
Fig. 2. Alignment of multi-view cameras

2-2 다시점 영상 획득

그림 3은 3D 모델 생성을 위한 영상 획득 장비로 사용한 산업용 카메라 Basler pilot piA1900-32gc 기종을 보여준다. 이 카메라의 최대 해상도는 Full HD(1926px × 1082px)이며 영상은 Bayer 포맷으로 생성된다.

그림 3. 영상 획득을 위한 산업용 카메라
Fig. 3. Industrial cameras for image aquisition

Bayer 포맷은 1채널 영상으로 각 픽셀에 1byte의 색상 정보를 가지고 있으며 그림 4에 보는 바와 같이 RGGB 형식의 픽셀 정보를 가지고 있다.

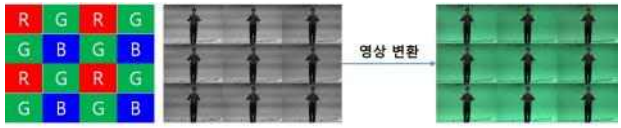


그림 4. Bayer에서 RGB 영상 변환
Fig. 4. Conversion of Bayer to RGB image

Bayer 영상에서 RGB 영상으로의 변환은 고품질 선형 보간 알고리즘[9]을 사용하였다. 표 1은 OpenCV의 cvtColor 함수와 고품질 선형 보간 알고리즘을 병렬처리 기법인 CUDA에 적용하여 변환한 속도를 보여준다.

실험 결과를 보면 OpenCV나 CUDA를 이용한 변환이나 전체 변환 속도는 큰 차이가 없다는 것을 알 수 있다. 하지만 CUDA를 이용한 변환 방법은 메인 메모리에서 GPU 메모리로의 데이터 복사에 많은 시간이 할애하는 것을 알 수 있었으며, 실제 변환 자체 속도만을 비교하면 OpenCV에 비해 몇 배 빨랐다. GPU 메모리에 올라온 데이터는 다음 전경과 배경 분리 작업에도 이용되므로 고품질 선형 보간 알고리즘 사용하였다.

표 1. Bayer 영상에서 RGB 영상 변환 속도 비교
Table. 1. Comparison of Bayer to RGB image conversion

<단위 : ms>

영상 개수		2	4	8
OpenCV	변환 속도	1.83775	4.46294	7.13746
CUDA 기반 알고리즘	변환 속도	0.20697	0.644181	1.40193
	메모리 복사	1.67034	3.487559	5.03526

2-3 시각 차이를 이용한 코드북 기반 전경과 배경 분리 알고리즘

그림 5는 RGB 공간에서 $\overline{p_1p_2}$와 $\overline{p_2p_3}$의 거리 값은 같지만 실제로 보이는 색상 차이는 다르다는 것을 보인다. 이렇게 시각적으로 차이를 보이는 것을 시각 차이(visual difference)라 칭한다.

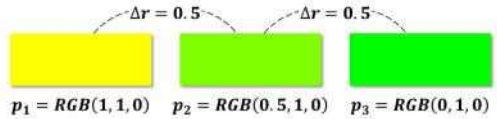


그림 5. RGB 공간에서 같은 거리의 색상 차이
Fig. 5. The visual differences between three RGB colors

전경과 배경을 구분하기 위하여 시각 차이를 이용하는데, 시각 차이는 세 가지 요소인 색상 차이(HD; hue difference), 채도(saturation), 명암(intensity)을 측정함으로써 시각 차이를 구한다.

그림 6은 HSI-RGB 스펙트럼(spectrum)이며 RGB 컬러 공간을 HSI 컬러 공간으로 변환시켜 주는 변환 모델을 보여준다. 이때 HSI 컬러 공간의 어떤 벡터 $HSI(h,s,i)$에 대응되는 RGB 컬러 공간의 벡터를 $RGB(f_R,f_G,f_B)$로 정의한다.

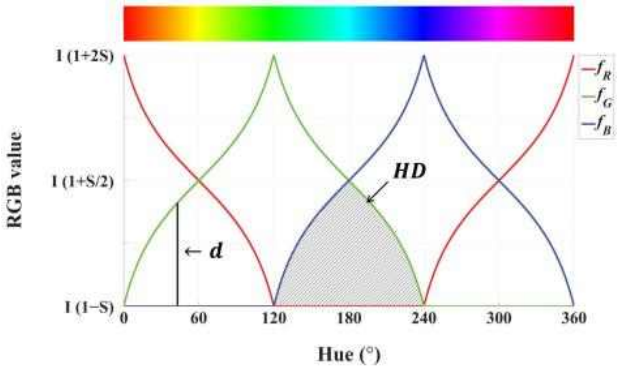


그림 6. HSI-RGB spectrum
Fig. 6. HSI-RGB spectrum

그림 6에서 d의 높이 값은 수식 (1)과 같이 정의되며

$$d(h,s,i) = mean(f_R, f_G, f_B) - \min(f_R, f_G, f_B) \quad (1)$$

색상 차이(HD; hue difference)값은 $HSI_1(h_1,s_1,i_1)$과 $HSI_2(h_2,s_2,i_2)$의 넓이로써 수식 (2)과 같이 정의된다.

$$HD_{(HSI_1, HSI_2)} = \frac{1}{2} \times \left(\int_{h_1}^{h_2} d(h, s_1, i_1) dh + \int_{h_1}^{h_2} d(h, s_2, i_2) dh \right) \quad (2)$$

표 2는 RGB 컬러 모델 내의 한 점을 R, G 축을 따라 거리 0.25 만큼씩 균일하게 이동시킬 때 나타나는 색상 차 *HD*와 Kim[10] 알고리즘에서 계산되는 컬러 왜곡(color distortion) δ를 비교한 것이다.

h=[60, 90] 구간에서의 Kim 알고리즘의 컬러 왜곡(color distortion) δ는 h = [60, 75]의 값보다 h = [75, 90]의 값이 작지만 색상 차이(HD; hue difference)의 값을 보게 되면 반대의 값을 갖는다.

이러한 결과는 RGB 공간에서 같은 거리를 가진 색을 구분하는 데 있어서 컬러 왜곡 값보다 색상 차이 값을 이용하면 시각적인 차이의 구분이 가능하다는 것을 보여준다.

표 2. 컬러 왜곡과 색상 차이의 값 비교
Table. 2. A comparison between color distortion and hue difference

		h = 45	h = 60	h = 75	h = 90	h = 105	h = 120	
δ		51.224	45.255	51.224	56.393	62.075		
HD		11.999	12.012	10.003	6.004	1.724		

본 논문은 시각 차이를 이용한 코드북 기반 전경과 배경 분리 알고리즘을 제안하며 코드워드의 요소로써 시각 차이의 값인 *HSI(h, s, i)*와 가장 작은 명암(I_{low})과 가장 큰 명암(I_{high})을 이용한다.

$$CW = \langle h, s, i, I_{low}, I_{high} \rangle$$

수식 (3)에 의해서는 τ_{HD}의 값에 의해서 색상 차이(HD; hue difference) 값이 결정되고 τ_S의 값에 의해서 채도(saturation) S의 값이 결정된다.

$$HD_{p,(HSI^j, HSI_f)} < \tau_{HD}, \ \left| S_p^j - S_{p,f} \right| < \tau_S \quad (3)$$

수식 (4)에 의해서 코드워드의 경계(I_B) 즉 가장 작은 명암(I_{low})과 가장 큰 명암(I_{high})이 α, β의 값에 의해 결정된다.

$$I_B = \left[\alpha I_{high}, \min\left\{ \beta I_{high}, \frac{I_{low}}{\alpha} \right\} \right] \quad (\alpha < 1 < \beta) \quad (4)$$

수식 (3)과 수식 (4)에 의해서 결정된 코드워드의 요소는 수식 (5)에서와 같이 프레임($j+1$)에 대한 모든 픽셀(p)의 코드워드를 업데이트함으로써 배경을 모델링한다.

최종적으로 객체와 배경을 분리하고자 하는 영상을 코드워드로 전환을 하고 각 픽셀에 대한 모델링된 코드워드와 비교함으로써 배경을 구분한다.

$$CW_p^{j+1} \leftarrow \left\langle HSI_p^j, \min\left\{ I_{low}{}_p^j, I_{low_{hf}} \right\}, \min\left\{ I_{high}{}_p^j, I_{high_{hf}} \right\} \right\rangle \quad (5)$$

그림 7은 코드북 기반 전경과 배경 분리 알고리즘이 적용된 결과를 보여준다.

그림 7. 전경과 배경 분리 결과
Fig. 7. Results of foreground and background subtraction

표 3은 nVidia Geforce GTX titan Z (2 GPU core) 1개를 이용하여 전경과 배경 분리 속도를 보여준다. 6개의 카메라에 대한 객체 추출 시간은 16ms 으로 실시간 처리가 확보됨을 알 수 있다.

표 3. 전/배경 분리 속도
Table. 3. Processing times of foreground and background subtraction

<단위 : ms>

	전경과 배경 분리		
	2	4	6
2 GPU Core	5.547	11.059	16.87

Ⅲ. 실시간 처리를 위한 전송 및 데이터 관리

3-1 송신 지연에 강인한 분산 처리 데이터 수집 방법

3D 모델을 생성하기 위해 3대의 노드에서 총 18개 카메라로부터 HD급 영상을 획득하고 전경과 배경 분리를 수행한 뒤 그 결과를 3D 모델 생성 서버로 전송한다.

이때 각 카메라가 초당 30장의 8 비트 Bayer 포맷 영상을 촬영한다고 가정할 경우 서버는 초당 약 950MByte의 데이터를 수신하게 된다.

본 논문에서는 RGB로 변환된 영상을 사용하였으며, 초당 데이터 크기는 HD 해상도 1280x720, RGB 영상 3채널, 전경과 배경 분리 영상 1채널, 18대 카메라, 실시간 처리를 위해 초당 30프레임을 이용한 총 1.85GByte를 사용하였다.

$$\text{초당 데이터 크기} = \text{픽셀수} \times \text{채널수} \times \text{카메라수} \times \text{촬영속도} \quad (6)$$

3D 모델 생성 서버에서 매초 수신하는 영상 데이터의 크기는 식 (6)을 이용하여 계산할 수 있다.

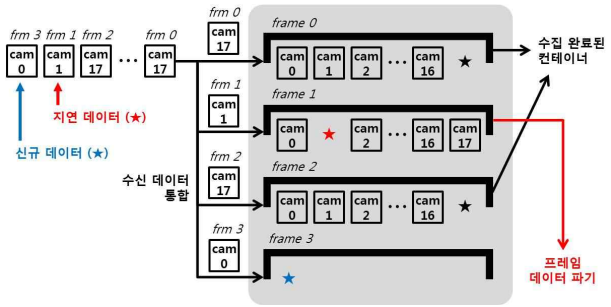


그림 8. 컨테이너 버퍼 기반 분산 데이터 동기화 및 관리 구조
Fig. 8. Designs of distributed data synchronization and management based on container

데이터를 수신하는 서버 측에서는 수집된 데이터를 다시 분석하여 3D 모델링에 사용하기 좋은 형태로 재배열하여야 한다. 그림 8은 컨테이너 버퍼 기반 분산 데이터 동기화 및 관리 구조 과정을 보여준다.

각 노드 단과 서버 단의 모든 프로세스가 항상 같은 속도로 동작하지는 않으며 분산 환경에서 처리된 영상 데이터가 서로 섞여 동시 다발적으로 입력되기 때문에 서버 측에서는 항상 영상 데이터를 프레임별로 분류하고 카메라 순서대로 정렬하는 작업이 필요하다.

이를 위해 컨테이너라는 이름의 버퍼 구조를 정의하고 노드로부터 수집된 분산 데이터를 각 컨테이너에 모아 담는 방식으로 분류 및 정렬을 수행하도록 한다. 하나의 컨테이너는 1개의 프레임에 대응되며, 각 컨테이너는 카메라 수 만큼의 데이터를 담을 수 있도록 설계하였다.

그리고 3D 모델링과 같은 그래픽 작업을 고속으로 처리하기 위해 GPU 연산을 이용할 때에는 동시에 처리하여야 하는 데이터들을 GPU 메모리로 한 번에 전달하는 편이 쉬우므로 최종적으로 18개 카메라 영상 수집이 완료된 컨테이너는 다시 GPU 연산을 위해 그래픽 카드 메모리로 전달된다.

이 과정에서 분산 노드 측의 처리 지연이 발생하여 데이터 입력이 늦어지는 경우 컨테이너가 다 찰 때까지 기다렸다가 처리되지만, 지연이 길어져 그다음 프레임 컨테이너가 먼저 다 차면 입력이 지연된 컨테이너의 프레임 데이터를 파기하고 그다음 프레임 데이터를 처리하도록 한다. 이를 통해 지연과 관계없이 버퍼를 순환시키며 항상 차례대로 데이터를 처리하도록 버퍼를 설계하여 관리하였다.

3-2 수신 지연에 강인한 분산 처리 데이터 관리 방법

일반적인 단일 영상 전송 시스템의 경우 수신 측에 원형 버퍼를 구축하여 네트워크 망을 통한 영상 데이터 입력과 이를 화면에 보여주는 출력 프로세스를 서로 비동기적으로 처리하지만, 사용 가능한 하드웨어 자원은 한정적이기 때문에 매초 수백 메가바이트에서 수 기가바이트 이상의 대용량 데이터가 생성되는 고품질 콘텐츠 생성 시스템에서는 데이터를 안정적으로 수집하기 위한 충분한 크기의 버퍼를 확보하기 어렵다.

수신 측의 처리가 조금만 지연되어도 쉽게 버퍼가 가득 찰 수 있고 이로 인해 송신 측에서도 전송이 가능해질 때까지 대기 상태에 빠질 수 있다. 일시적인 병목 현상이나 처리 지연과 같은 시스템 문제에 대처하기 위해서는 그만큼의 큰 버퍼가 필요하며 이에 따른 시스템 구축 비용도 같이 증가한다.

따라서 이러한 문제에 더욱 강인하고 유연하게 운용할 수 있는 시스템을 구축하기 위해서는 데이터 읽기 및 쓰기가 가능해질 때까지 기다리는 종래의 원형 버퍼와 달리, 쓰기 가능한 버퍼가 부족할 경우 사용 대기 중인 버퍼 일부를 반환하여 쓰기 가능한 상태로 만들 수 있는 순환 구조로 버퍼를 설계하였다.

그림 9는 버퍼를 순환 운용하기 위해 각 버퍼에 4가지 상태값을 부여하여 동적으로 관리하는 방법을 보여준다.

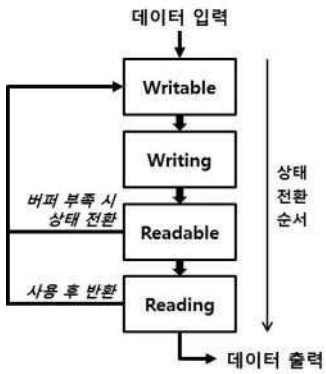


그림 9. 버퍼 상태를 이용한 동적 관리 방법
Fig. 9. A dynamic management method using states of buffer

모든 버퍼의 상태는 각각 쓰기 가능(writable), 쓰기 중(writing), 읽기 가능(readable), 읽기 중(reading)의 4가지로 구분되며 상시 동적으로 변화한다. 데이터의 입력을 기다리고 있는 상태를 writable, 현재 데이터를 기록 중인 상태를 writing, 기록이 완료되어 작업 대기 중인 상태를 readable, 특정 작업에서 데이터를 읽기 위해 버퍼를 점유하고 있는 상태를 reading이라고 한다. 작업 종료 후 사용이 완료된 버퍼는 다시 writable 상태로 전환되어 다음 데이터 입력을 기다린다. 만일 작업 시간이 길어져 writable 버퍼의 수가 0이 되었을 때 새로운 데이터가 입력되면, 가장 오래된 readable 버퍼를 반환하여 writable 상태로 전환한다.

제안된 버퍼 관리 방법은 영상처리 및 그래픽스 등의 분야에서 디스플레이를 위해 일반적으로 사용되는 이중 버퍼(double buffer) 방식과 유사하나, 버퍼의 상태가 상시 변화하며 동적으로 운용되기 때문에 작업 처리 지연으로 인해 버퍼가 가득 차더라도 입력 층에서 작업 종료를 기다리지 않고 곧바로 데이터를 기록할 수 있다는 점에서 차이가 있다.

그림 10은 이와 같은 버퍼 상태 기반의 동적 관리 방법을 이용하여 시스템을 운용하기 위해, 2개의 큐(queue)를 연결한 버퍼 순환 구조를 보여준다. 이는 최소 3개 단위의 버퍼만으로도 일시적인 병목 현상이나 처리 지연과 같은 문제에 강인한 시스템을 구축하는 방법을 제공한다.

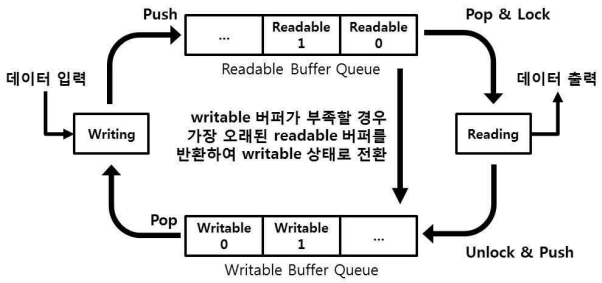


그림 10. 상태 기반 동적 관리 방법을 이용한 버퍼 순환 구조
Fig. 10. A design of buffer circulation mechanism using dynamic management method based on buffer states

그림 11은 네트워크 망을 통해 영상 데이터를 수신하는 서버 측에서 일시적인 병목 현상이나 처리 지연과 같은 시스템 문제가 발생하였을 때, 링 버퍼와 본 절에서 제안한 순환식 버퍼를 각각 이용하여 영상을 수신 및 출력한 결과를 비교한 것이다.

실험 결과 지연과 같은 시스템 문제가 발생하고 있는 상황에서 제안한 방법을 통해 얻은 결과는 실시간 재생 시와 유사한 프레임 처리가 가능하였으며 기존 링 버퍼를 사용했을 때는 실시간에서 처리해야 할 프레임보다 이전 프레임을 처리하여 시스템 전체적으로 지연 현상이 발생함을 확인하였다.

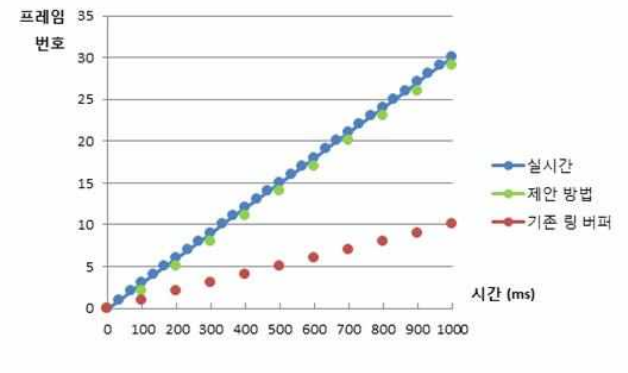


그림 11. 지연 발생 시의 버퍼 관리 방법 별 프레임 재생 간격
Fig. 11. Frame intervals according to buffer management method when delays are caused

Ⅳ. 영상 기반 3차원 모델링

본 논문에서 3D 모델링 방법은 삼차원 공간 정보를 이용한 영상 투영을 통해 3D 모델을 생성하는 방법인 visual hull 기법[11]을 이용하였다. 삼차원 공간 정보는 카메라 캘리브레이션을 통해서 얻을 수 있는 카메라의 외부인자와 내부인자로 구할 수 있다.

수식 (7)은 카메라의 내부인자(K)와 외부인자(E)를 보여준다.

$$K=\begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix},\ E=\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}\quad \begin{cases} fx, fy : focal\ length \\ u_0, v_0 : image\ center \end{cases} \tag{7}$$

카메라 캘리브레이션을 통해서 획득된 내·외부 인자를 이용해서 객체(object)의 삼차원 좌표(X_s, Y_s, Z_s)를 수식 (8)에서와 같이 외부 인자에 곱하게 되면 카메라를 원점으로 하는 카메라 좌표(x_s, y_s, z_s)가 된다.

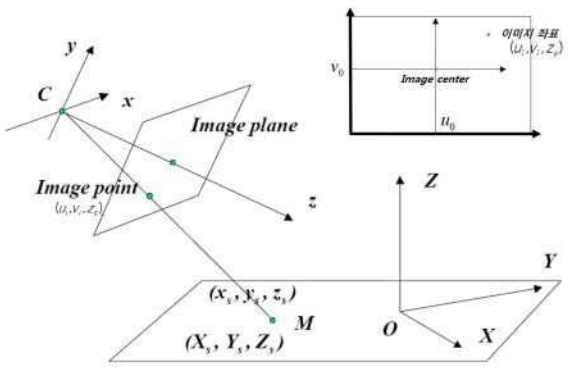


그림 12. 영상 투영
Fig. 12. An image projection

이에 내부인자를 곱하게 되면 그림 12에서 보이는 것처럼 영상으로 투영되는 이미지 좌표(u_i, v_i, z_s)를 얻을 수 있다.

$$\begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = E \begin{bmatrix} X_s \\ Y_s \\ Z_s \\ 1 \end{bmatrix}, \; I_c = \begin{bmatrix} u_i \\ v_i \\ w_i \end{bmatrix} = K \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} \begin{cases} u_i = f_x x_s + u_0 z_s \\ v_i = f_y y_s + v_0 z_s \\ w_i = z_s \end{cases} \quad (8)$$

삼차원 볼륨은 가장 작은 단위인 큐빅 형태의 복셀로 이루어진 그리드 형태이다. 이러한 볼륨 안에 실제 객체(object)가 위치하게 되며 실 객체는 경계 복셀(surface voxel)과 내부 복셀로 이루어져 있다. 삼차원 메쉬 모델을 만들기 위해서는 객체의 내부 영역은 필요하지 않기 때문에 그림 13과 같이 내부 복셀을 제외한 경계 복셀 만을 이용한다.

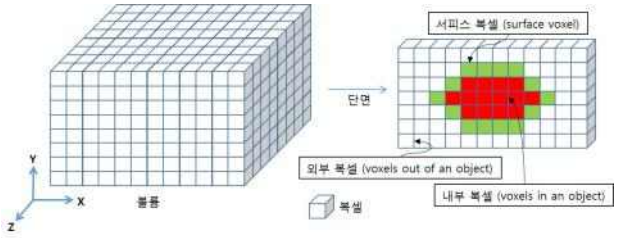


그림 13. 삼차원 볼륨 및 복셀
Fig. 13. 3D volume and voxels

경계 복셀은 수식 (8)에서와 같이 각 복셀을 전경과 배경 분리 영상에 투영하면 영상의 위치를 통해서 알 수 있다. 이렇게 해서 검색된 경계 복셀은 표 4와 같다. 표에서 보는 바와 같이 전체 복셀에 대한 경계 복셀의 비율은 약 1%의 점유율을 갖는다는 것을 알 수가 있다.

표 4. 전체 복셀 대비 경계 복셀 점유율
Table. 4. The occupancy rates of surface voxels relative to the total voxels

그리드 개수 / 복셀	64^3	128^3	256^3	512^3
전체 복셀	262,144	2,097,152	16,777,216	134,217,728
경계 복셀 (surface voxel)	2,134	8,944	76,222	158,727
점유률	0.814%	0.426483%	0.454319%	0.118261%

이런 결과는 모든 복셀을 검색하여 경계 복셀을 찾는 것은 많은 시간을 소비하는 것을 알 수 있다.

본 논문에서는 검색 시간을 단축하려는 방법으로 Octree 기반 볼륨 모델 생성 방법을 제안한다. 삼차원 공간을 x축, y축, z축을 기준으로 절반을 나누면 8조각이 된다. 8조각으로 나뉜 볼륨을 영상으로 투영하면 탐색이 필요치 않은 볼륨을 가릴 수 있고 탐색이 필요한 볼륨은 다시 세 축으로 나누어 재탐색을 수행한다.

그림 14에서는 경계 복셀을 탐색하기 위해서 Octree를 이용하여 탐색하는 과정을 보여준다.

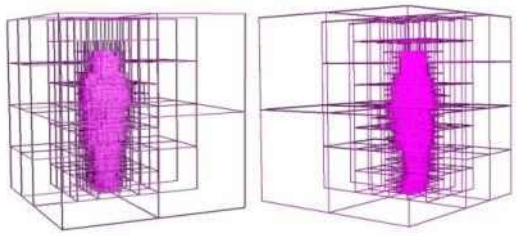


그림 14. Octree를 이용한 경계 복셀 탐색
Fig. 14. A search of surface voxels using octree

경계 복셀의 검색이 마친 후 일반적인 마칭 큐브(Marching Cube) 기법을 이용하여 메쉬 모델을 생성한다. 그림 15의 왼쪽 위에서 각 복셀의 녹색 점은 영상으로 투영 시 객체의 내부로 투영되는 점이며 이러한 점의 배치에 따라서 삼각형을 생성하는 것이다.

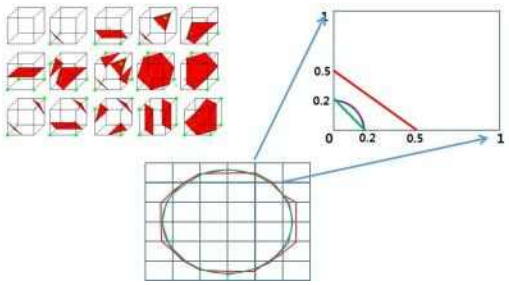


그림 15. 마칭 큐브를 이용한 메쉬 생성
Fig. 15. A mesh generation using marching cubes

기본적인 마칭 큐브 알고리즘은 그림 15의 오른쪽 위에서 0.5 위치 라인을 그리게 되며 그림 15의 중간에서 보는 바와 같이 찌그러진 원을 그리게 된다. 실제 원에 가깝게 하기 위해서는 0.2 위치에 라인을 그리면 된다.

표 5는 64^3, 128^3, 256^3, 512^3 그리드 형태의 Octree를 이용한 메쉬 복원 속도를 보여준다. 643 그리드 형태에서 실시간 복원 속도를 볼 수 있다.

표 5. 3D 메쉬 복원 속도
Table. 5. Processing times of 3D mesh restoration

<단위 : ms>

그리드 개수 / 속도	64^3	128^3	256^3	512^3
볼륨 생성 속도	16.732	81.7328	674.1898	2069.6228
마칭큐브 속도	10.927	47.8224	246.7236	1218.8320
복원 속도	27.659	129.5552	920.9134	3288.4548
삼각형(face) 개수	4,321	18,000	76,222	158,727

그림 16은 원근 보정 보간법을 이용하여 삼차원 점(x_1, z_1)과 점(x_2, z_2)사이에 있는 점(x_3, z_3)을 찾는 방법을 보여준다. 간략히 설명하면 삼차원 점(x_1, z_1)영상에서 투영된$(p_1, -e)$와 삼차원 점(x_2, z_3)영상에서 투영된 $(p_2, -e)$을 알고 있을 때, $(p_3, -e)$ 위치에서 삼차원 공간 상의 점 (x_3, z_3)을 구하는 것이다.

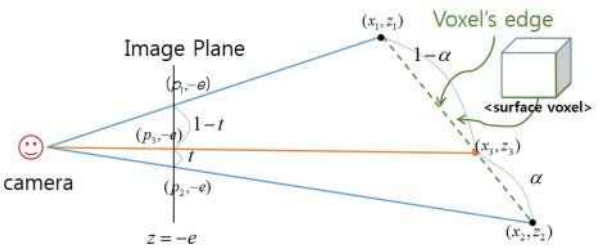


그림 16. 원근 보정 보간법
Fig. 16. An interpolation of perspective correction

$$ax_3 + bz_3 = c,\ \frac{p_3}{x_3} = \frac{-e}{z_3} \tag{9}$$

수식 (9)에 의해서 수식(10)가 도출이 된다.

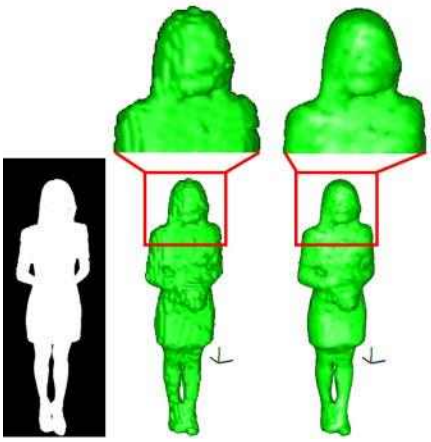
$$\frac{1}{z_3} = \frac{ap_3}{ce} + \frac{b}{c} \tag{10}$$

$$p_3 = (1-t)p_1 + tp_2 \tag{11}$$

도출된 수식(10)과 수식 (11)에 의해서 수식 (12)가 최종적으로 도출된다. 수식 (12)는 영상에서 투영된 위치의 비율을 이용하여 삼차원 공간의 z_3을 구할 수 있다는 것을 보여준다.

$$\frac{1}{z_3} = \frac{1}{z_1}(1-t) + \frac{1}{z_2}t \tag{12}$$

그림 17은 원근 보정 보간법을 이용한 재구성된 메쉬 모델을 보여준다.



(a) (b) (c)

그림 17. 메쉬 모델링 비교 (a) 전경과 배경 분리 영상 (b) 일반적인 마칭 큐브를 이용한 메쉬 모델링 (c) 원근 보정 보간법을 이용한 모델링
Fig. 17. Comparisons of mesh models (a) foreground and background subtraction image (b) the mesh model using original marching cube (c) the mesh model using perspective correction

생성된 메쉬에 텍스쳐를 입히기 위해서는 쉐이더(Cg Shader)를 이용한 시점 의존 텍스쳐 매핑(view-dependent texture mapping) 기법을 이용한다. 시점 의존 텍스쳐 매핑 기법은 각 카메라에서 획득된 영상을 텍스쳐 소스로 이용하는 것으로써 사용자가 바라보는 시점과 가장 가까운 카메라의 영상을 텍스쳐로 사용하는 방법이다.

삼차원 메쉬에 텍스쳐를 입히기 위해서 쉐이더를 사용하는데 이는 메쉬의 삼각형 정점을 그 시점에 맞는 영상에 투영하고 얻어진 세 점의 색깔에 대해서 보간법을 통해 삼각형의 면을 채우는 방식이다. 이 방식은 GPU에서 수행하기 때문에 빠른 시간에 면의 색깔을 채울 수 있게 된다.

그림 18은 시점 의존 텍스쳐 매핑 결과를 보여준다.

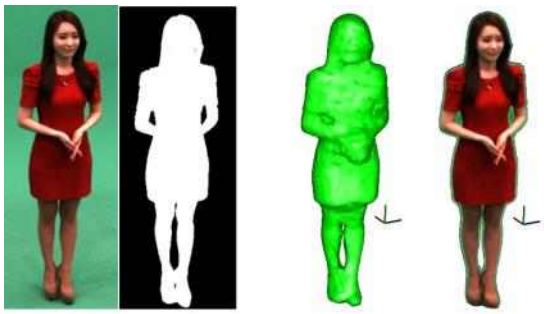


그림 18. 시점 의존 텍스쳐 매핑
Fig. 18. View dependent texture mapping

Ⅴ. 결 론

본 논문에서 다시점 영상 획득에서 영상 기반 3D 모델 생성까지의 과정을 분산으로 병렬처리 할 수 있으면서 지연 처리에 강건한 실시간 3D 생성 시스템을 소개하였다.

18대의 카메라에서 입력되는 HD급 대용량 영상을 3대의 노드로 병렬 수집하고 이렇게 획득한 영상 데이터에서 배경과 객체를 분리하기 위하여 시각 차이를 이용한 코드북 기반 전경과 배경 분리 알고리즘을 제안하였고 이 알고리즘을 통해 객체를 효율적으로 추출하였다.

그리고 제안한 전경과 배경 분리 알고리즘을 통해서 추출된 마스크 영상은 3D 모델 생성을 위한 서버로 전송되는데 이때 시스템 불안정으로 인한 송신 지연 또는 3D 복원 성능 저하로 인한 처리 지연에 강건하게 처리할 수 있는 데이터 메모리 관리 방법을 제시하여 실시간 처리가 가능한 시스템을 확보하였다.

마지막으로 3D 모델 생성은 카메라 캘리브레이션을 적용하여 3D 공간 정보를 얻고 캘리브레이션의 파라미터를 이용하여 그리드 형태의 볼륨을 영상에 투영함으로써 볼륨 모델을 생성하였다. 생성된 볼륨 모델에 일반적인 마칭 큐브를 개선한 원근 보정 보간법을 이용하여 정밀하고 부드러운 3D 메쉬 모델을 생성하고 쉐이더를 적용하여 텍스쳐를 생성하였다. 이 시스템은 현재 초당 30프레임 처리 속도로 구축되어 있다.

감사의 글

본 연구는 미래창조과학부 '범부처 Giga KOREA 사업'의 지원에 의하여 이루어진 연구로서, 관계부처에 감사드립니다.

[GK16C0100, 기가급 대용량 양방향 실감 콘텐츠 기술 개발]

참고문헌

[1] Y. Ohta, I. Kitahara, Y. Kameda, H. Ishikawa, and T. Koyama, "Live 3D Video in Soccer Stadium," International Journal of Computer Vision, Vol. 75, No. 1, pp. 173–187, 2007.

[2] A. Chapiro, S. Heinzle, et al., "Optimizing Stereo-to-Multiview Conversion for Autostereoscopic Displays," *Computer graphics forum*, Vol. 33, No. 2, pp. 63-72, May 2014.

[3] Cheon Lee, Yo-Sung Ho, "Depth-tuned Intermediate View Generation for Comfortable 3D Video," *2012 KICS*, Vol. 6, pp. 349-350, 2012.

[4] Lang, Manuel., et al., "Nonlinear Disparity Mapping for Stereoscopic 3D," *ACM Transactions on Graphics (TOG)*, Vol. 29, Issue 4, Article No. 75, July 2010

[5] C.Kim, A. Hornung, S. Heinzle, et al., "Multi-Perspective Stereoscopy from Light Fields," *Proceeding SA '11 Proceedings of the 2011 SIGGRAPH Asia Conference*, Vol. 30, Issue 6, Article No. 190, December 2011.

[6] D. E. C. P. R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems*, pp. 2366-2374, 2014.

[7] P. Gargallo, P. Sturm, "Bayesian 3D modeling from images using multiple depth maps," *CVPR '05 Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 885-891, 2005.

[8] Jae-young Jung, "Codebook-Based Foreground Extraction Algorithm with Continuous Learning of Background," *Journal of Digital Contents Society*, Vol. 15, No. 4, pp. 449-455, 2014.

[9] Malvar, Henrique S., Li-wei He, and Ross Cutler, "High-quality linear interpolation for demosaicing of Bayer-patterned color images," *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP'04). IEEE International Conference on.*, Vol. 3, pp. 482-485, 2004.

[10] K. Kim, et al., "Background Modeling and Subtraction by Codebook Construction," *Image Processing, 2004. ICIP'04. 2004 International Conference on.*, Vol. 5, pp. 3061-3064, 2004.

[11] W. Matusik, C. Buehler, R. Raskar, et al., "Image-based visual hulls," *SIGGRAPH '00 Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 369-374, 2000.

박정선(Jeong-Sun Park)

1998년 : 전북대학교 (학사-컴퓨터과학과)
2000년 : 전북대학교 대학원 (이학석사-전산통계학과)

2002년~2003년: ㈜엔비즈테크놀로지 연구원
2003년~현 재: ㈜코난테크놀로지 코난링크사업부 부장
※관심분야 : 패턴인식, 미디어 에셋 관리, 인공지능

손형재(Hyung-Jae Son)

2012년 : 충남대학교 (학사-컴퓨터공학과)
2015년 : 충남대학교 대학원 (공학석사-컴퓨터공학과)

2013년~2015년: 한국전자통신연구원 위촉연구원
2015년~현 재: ㈜코난테크놀로지 주임연구원
※관심분야 : 영상처리, 컴퓨터 비전, 실시간 3D/다시점 콘텐츠 생성 시스템

박정철(Jeung-Chul Park)

2004년 : 전북대학교 (학사-컴퓨터공학과)
2006년 : 광주과학기술원 대학원 (공학석사-기전공학과)

2006년~현 재: 한국전자통신연구원 기업현장지원실 선임연구원
※관심분야 : 컴퓨터 비전, 컴퓨터 그래픽스, 3D 모델링 및 렌더링, 다시점 콘텐츠

오일석(Il-Seok Oh)

1984년 : 서울대학교 (학사-컴퓨터공학과)
1992년 : 한국과학기술원 (공학박사-전산학과)

1992년~현 재: 전북대학교 전자정보공학부 교수
2003년~현 재: 전북대학교 영상정보신기술연구소 소장
※관심분야 : 영상처리, 패턴인식, 컴퓨터 비전, 인공지능