

볼륨프로 하드웨어를 이용한 효율적인 투시투영 방법

(An Efficient Perspective Projection using VolumePro™ Hardware)

임석현* 신병석**
(Sukhyun Lim) (Byeong-Seok Shin)

요약 볼륨프로는 일반 PC상에서도 실시간 볼륨렌더링이 가능하도록 고안된 하드웨어이다. 그러나 볼륨프로는 직교투영(orthographic projection) 기능만 제공하기 때문에 가상 내시경과 같이 투시투영(perspective projection)이 필요한 분야에는 이용할 수 없다. 이런 문제를 해결하기 위해 볼륨 데이터를 여러 개의 슬랩(slab)으로 분할하여 투시투영을 근사하는 방법이 소개되었으나 이 방법은 시각 질두체에 포함되지 않은 부분까지 렌더링 하기 때문에 처리시간이 길어지는 문제가 있다. 본 논문에서는 볼륨프로의 크로핑(cropping) 기능을 이용하여 볼륨 데이터를 여러 개의 서브볼륨으로 분할함으로써 효율적인 투시투영이 가능하도록 하는 방법을 제안한다. 이 방법은 시각 질두체(view frustum)에 포함되는 부분만 렌더링 하므로 슬랩을 이용하는 기존 방법과 동일한 화질을 가지면서 처리시간을 단축시킨다.

키워드 : 볼륨프로, 볼륨렌더링, 슬랩, 서브볼륨, 직교투영, 투시투영

Abstract VolumePro is a real-time volume rendering hardware for consumer PCs. However it cannot be used for the applications requiring perspective projection such as virtual endoscopy since it provides only orthographic projection. Several methods have been presented to approximate perspective projection by decomposing a volume into slabs and applying successive parallel projection to them. But it takes a lot of time since the entire region of every slab should be processed, which does not contribute to final image. In this paper, we propose an efficient perspective projection method that makes the use of several sub-volumes with cropping feature of VolumePro. It reduces the rendering time in comparison to slab-based method without image quality deterioration since it processes only the parts contained in the view frustum.

Key words : VolumePro, volume rendering, slab, subvolume, orthographic projection, perspective projection

1. 서론

볼륨 렌더링은 복셀들의 집합으로 표현된 볼륨 데이터로부터 의미있는 정보를 추출하여 가시화하는 기법이다[1]. 기하학적 모델을 기반으로 하는 기존의 렌더링 방법과 달리 볼륨 렌더링을 이용하면 물체의 내부를 표시하거나 반투명한 물체를 표현하는 것이 용이하다. 그러나 볼륨 데이터를 저장하기 위해서는 상대적으로 대용량의 메모리가 필요하고 연산 횟수가 많기 때문에 질

과 영상을 얻어내기까지 상당한 시간이 소요되는 문제가 있다. 이러한 문제 때문에 실시간 처리를 필요로 하는 사용자들의 요구를 만족시키지 못하고 있다.

볼륨 데이터의 렌더링 속도를 향상시키기 위해 다양한 알고리즘들이 고안되었으나 소프트웨어적인 방법만으로 실시간 볼륨 렌더링을 구현하는 데는 많은 제약이 따른다. 볼륨 렌더링 전용 워크스테이션이나 3D 텍스처 매핑(3D texture mapping) 하드웨어를 이용하면 대화형 실시간 렌더링이 가능하지만 이런 시스템은 가격이 매우 비싸다. 최근에는 가격이 저렴하면서 우수한 성능을 가진 범용 그래픽 가속기가 소개되었다. 하지만 이와 같은 하드웨어들은 볼륨 렌더링을 위하여 최적화된 것이 아니므로 상당한 소프트웨어적 처리가 필요하다[2]. 미쓰비시사에서 개발한 볼륨프로 하드웨어는 저가의 일반 PC에서도 고속으로 볼륨 렌더링 영상 생성이 가능

* 이 논문은 2001년도 한국학술진흥재단의 지원에 의하여 연구되었음
(과제번호 : KRF-2001-041-E00268)

† 학생회원 : 인하대학교 전자계산공학과
g2011498@inhavision.inha.ac.kr

** 종신회원 : 인하대학교 컴퓨터학부 교수
bsshin@inha.ac.kr

논문접수 : 2002년 11월 1일
심사완료 : 2003년 11월 27일

하기 때문에 적합한 해결방법으로 인식되어왔다[3].

그러나 볼륨프로는 직교투영만 가능하도록 제작되었기 때문에 투시투영이 필요한 분야에는 적용하기 어렵다. 즉 사용자가 물체를 외부에서 바라볼 경우에는 좋은 결과를 내지만 가상 내시경과 같이 카메라가 물체의 내부에 위치하고 투시투영 효과를 필요로 하는 데는 활용할 수 없다. 이것을 보완하기 위해 부분볼륨에 직교투영을 반복적으로 수행하여 투시투영을 모사하는 방법이 소개되었다[4]. 여기서는 볼륨 데이터를 다수의 얇은 슬랩들로 나누고 각 슬랩별로 직교투영한 영상을 혼합하여 투시투영을 한 것과 같은 효과를 내도록 한다.

슬랩 기반 방법은 슬랩 두께를 적절히 조절했을 때 투시투영에 근사한 영상을 얻어낼 수 있지만 슬랩의 두께만 조절할 수 있기 때문에 시각 절두체에 포함되지 않는 부분까지 렌더링을 하는 문제가 있다. 본 논문에서는 슬랩 기반 방법의 단점을 개선하기 위해 서브볼륨 기반 방법을 제안한다. 서브볼륨은 볼륨의 두께뿐만 아니라 폭과 너비까지 지정할 수 있기 때문에 시각 절두체에 포함되는 부분만 렌더링 하도록 조절할 수 있다. 따라서 슬랩 기반 방법과 동일한 화질을 유지하면서 처리시간을 단축시킬 수 있다.

논문의 구성은 다음과 같다. 2절에서는 렌더링 속도를 향상시키기 위한 기존의 연구들을 비교 고찰한다. 3절에서는 본 논문에서 제안하는 서브볼륨 기반 방법을 기존의 슬랩 기반 방법과 비교하여 설명하고 서브볼륨 기반 방법의 장점을 설명한다. 4절에서는 가상내시경 시스템에 두 가지 방법을 적용하여 실험한 결과를 보이고 마지막으로 결론을 맺는다.

2. 관련연구

볼륨 데이터로부터 3차원 영상을 만들어내는 방법은 크게 표면 렌더링(surface rendering)과 직접 볼륨 렌더링(direct volume rendering)으로 나뉜다. 표면 렌더링은 볼륨 데이터로부터 물체의 표면 정보를 표시하는 다각형 모델(polygon model)을 재구성한 후 기존 렌더링 알고리즘들을 활용하여 3차원 영상을 만들어낸다[5]. 표면 렌더링은 속도가 빠르지만 근육, 지방층 등 병변의 발생가능성이 높은 연조직을 가시화하는 분야에는 적절하지 못하다. 직접 볼륨 렌더링에서는 볼륨 데이터의 각 픽셀들이 가지는 스칼라 값(scalar value)과 그 값들의 공간적 변화량(gradient)을 기반으로 표면 정보를 추정하여 영상을 만들어 낸다[6]. 직접 볼륨 렌더링은 표면 렌더링에 비해 화질이 좋고 반투명한 인체 해부 구조물을 표현하는데 유용하지만 처리 시간이 오래 걸리는 단점이 있다. 대표적인 것으로는 볼륨 광선 투사법(volume ray casting)이 있다.

최근의 볼륨 렌더링 응용 프로그램들은 실시간에 가까운 고속 영상 생성을 요구하고 있어 여러 가지 가속화 방법들이 이용된다. 광선 궤적의 공간 일관성을 이용한 템플릿 기반 렌더링(template-based rendering)[7]이나 쉬어-워프 렌더링(shear-warp rendering)[8]이 여기에 속한다. 그러나 이것들을 이용하는 경우에도 실시간 볼륨렌더링을 하기는 어려우며, 더욱이 투시투영에 적용할 경우는 직교투영을 하는 경우와 비교하여 가속효과가 크게 떨어지는 문제가 있다.

대화식 실시간 영상 생성을 위해서는 하드웨어 기반의 방법이 필요하다. 특수하게 고안된 병렬 처리 컴퓨터나 3차원 텍스처 매핑 하드웨어를 이용하여 볼륨 데이터를 병렬 처리하는 연구가 과거부터 계속되어왔다[9-11]. 이들은 화면 기반 분할, 데이터 기반 분할에 의해 입력된 볼륨 데이터를 여러 개로 나누고 각각의 처리단위(processing unit)들이 볼륨 데이터의 일부를 처리한 후 집약하도록 한다. 그러나 병렬 처리가 가능한 다중 프로세서 시스템은 비용이 많이 들어 일반적으로 사용하기 어려운 문제가 있다. 최근에는 볼륨프로와 같이 PC상에서 실시간 볼륨 렌더링이 가능한 저가의 하드웨어가 소개되었다. 초기 모델은 렌더링 방법으로 쉬어-워프 분해 방법을 사용하여 화질이 다소 저하되는 문제가 있었으나 최신 모델은 고화질을 보장하는 볼륨 광선 투사법을 활용하여 양질의 영상을 얻어낼 수 있다.

그러나 볼륨프로는 직교투영만 가능하도록 제작되었기 때문에 투시투영이 필요한 응용분야에는 적용하기 어렵다. 이것을 보완하기 위해 부분적인 직교투영을 반복하여 투시투영을 모사하는 방법이 소개되었다[4]. 여기서는 볼륨프로 하드웨어가 제공하는 절단면(cut plane) 기능을 이용하여 볼륨 데이터를 다수의 얇은 슬랩들로 나누고 각 슬랩별로 직교 투영한 영상을 크기변환(scaling)한 후 잘라내기(clipping) 과정을 거친다. 마지막으로 생성된 중간 영상들을 혼합(blending)하여 투시투영을 한 것과 유사한 효과를 내도록 하는 것이다. 슬랩 기반 방법은 슬랩 두께를 적절히 조절했을 때 투시투영에 근사한 영상을 얻어낼 수 있지만 슬랩의 두께만 지정할 수 있기 때문에 시각 절두체에 포함되지 않는 부분까지 렌더링을 하게 되고, 이로 인하여 처리시간이 길어지는 문제가 있다.

3. 서브볼륨을 이용한 투시투영 모사 기법

여기서는 볼륨프로 하드웨어가 제공하는 크로핑 기능을 이용하여 투시투영 영상을 효율적으로 생성해낼 수 있는 방법을 설명한다.

3.1 슬랩 기반 투시투영 기법

볼륨프로에서는 하드웨어의 기능을 응용 소프트웨어

에서 제어할 수 있도록 VLI(Volume Library Interface)라는 응용 프로그램 인터페이스를 제공한다[12]. VLI는 C++클래스 집합으로서 여기서 제공되는 API들을 이용하여 블롭프로의 각종 기능들을 제어할 수 있다. VLI의 다양한 기능들 중에는 블롭 데이터의 특정 부분을 잘라내어 부분영상을 만들어 낼 수 있는 기능들이 있다. 부분블롭 데이터로는 슬랩과 서브블롭이 있다. 슬랩은 그림 1(a)에서처럼 일정 거리만큼 떨어진 두개의 평행한 평면으로 정의되는 공간이고, 서브블롭은 그림 1(b)에서처럼 전체블롭 내에서 렌더링되어질 복셀수를 제한시킨 부분블롭이다.

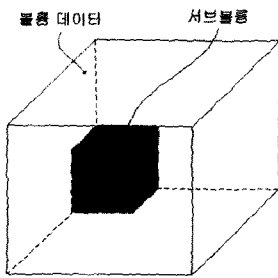
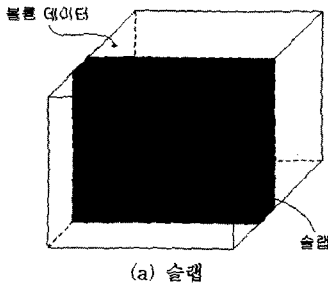


그림 1 슬랩과 서브블롭

슬랩을 이용하여 투시투영을 모사하는 경우 최종 영상이 만들어지는 단계는 그림 2와 같다.

먼저 카메라 위치와 방향이 정해지면 전체 블롭을 일정한 두께를 가지는 여러 슬랩들로 나눈다(그림 2(a)). 슬랩을 지정해주기 위해 VLI의 Create() API를 사용한다. 하나의 슬랩을 정의하기 위해서는 법선 벡터(normal vector), 원점으로부터의 거리, 두께와 기울기를 정해주어야 한다. 다음으로 각각의 슬랩들을 직교투영하여 중간 영상들을 만든다(그림 2(b)). 생성된 중간 영상들은 최종 영상의 크기에 비례하여 크기변환을 한다(그림 2(c)). 즉 중간영상에서 시각 질두께와 교차하는 부분이 최종영상 크기에 맞도록 변환한다. 크기변환된 영상은 최종 영상 크기에 맞게 잘라내기를 하고(그림 2(d)), 크기가 조절된 중간 영상들은 그래픽스 하드웨어가 제공하는 텍스처 혼합(texture blending) 기능으로 혼합하여 최종 영상을 얻어낸다(그림 2(e)).

그림 3은 슬랩을 이용한 투시투영기법을 도식화한 것이다. $Z_0 \sim Z_{n-1}$ 까지는 슬랩의 두께이고 D_0 는 카메라로부터 첫 번째 슬랩까지의 거리이다.

Create(double inA, inB, inC, inD, double inThickness, double inFallOff);

(inA, inB, inC) : 법선벡터, inD : 원점으로부터의 거리, inFallOff : 기울기

(1) 슬랩을 생성하는 API

슬랩을 생성할 때는 카메라로부터의 거리와 두께만을 지정할 수 있으므로 슬랩을 이용하여 투시투영을 하려면 각각의 슬랩 전체를 렌더링하여 중간영상을 만든 후 시각 질두께에 해당하는 부분을 전체 블롭 크기에 맞게 크기변환하는 작업이 필요하다. 실제 렌더링에는 시각 질두께에 포함되는 부분만 필요하지만 그림 3에서 보는 바와 같이 슬랩을 사용하면 실제 최종영상에 기여하지 않는 부분까지 계산하는 문제가 있다. 렌더링 단계에서

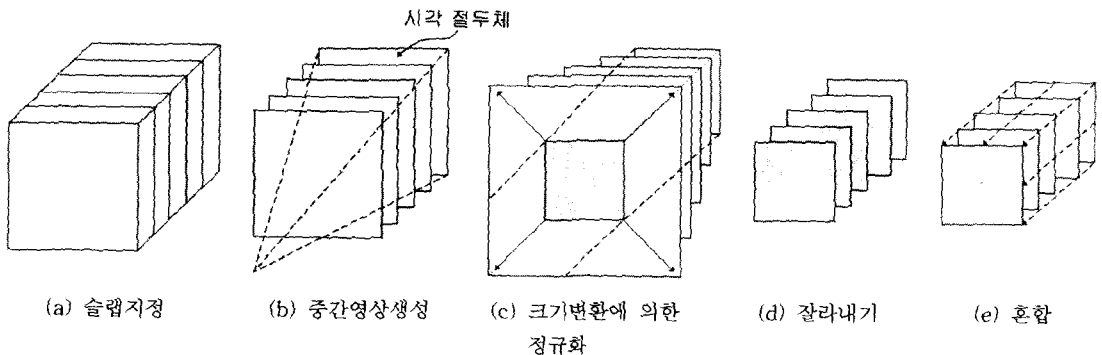


그림 2 슬랩을 이용하여 투시투영영상을 만드는 단계

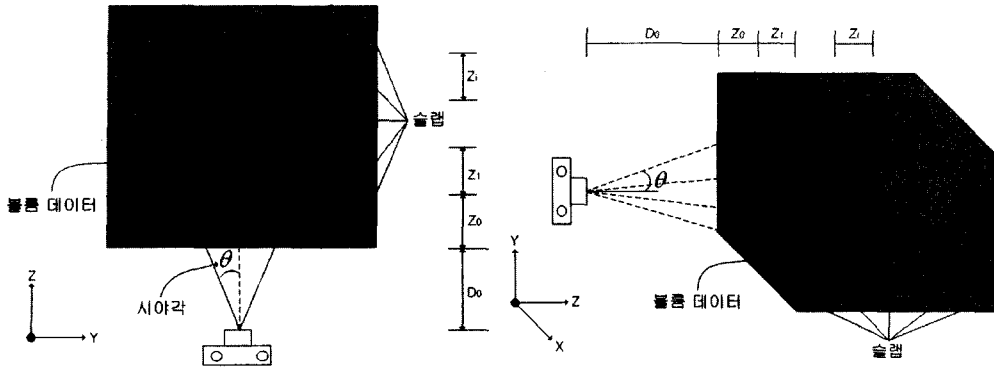


그림 3 볼륨프로필을 이용한 슬랩 렌더링

최종 영상과 무관한 부분까지 렌더링 하면 불필요한 비용이 소요된다. 또한 생성된 중간영상을 확대하는 과정에서 최종 영상에 반영되지 않는 부분들까지 확대해야 하므로 기억공간의 낭비가 발생한다.

3.2 서브볼륨 기반의 볼륨 렌더링

서브볼륨을 이용하여 투시투영을 모사하는 경우에는 관심영역(Volume of Interest : VOI)에 해당하는 볼륨만을 따로 생성할 수 있다. 이 방법은 시각 절두체와 교차하는 영역만을 계산하여 서브볼륨을 만들고 이것만을 렌더링함으로써 효율적인 렌더링이 가능하다.

그림 4는 시각 절두체에 맞게 생성된 각각의 서브볼륨을 표현한다. 그림에서 회색부분만 렌더링되고 흰색 부분은 무시된다. 따라서 서브볼륨별로 렌더링 시간이 감소할 뿐만 아니라 중간 영상을 저장하기 위한 기억공간도 감소된다. 그림 5는 서브볼륨을 이용하여 영상을 만들어내는 과정이다.

먼저 카메라 위치와 방향이 정해지면 전체 볼륨을 시각 절두체에 맞게 계산하여 여러 개의 서브볼륨을 만든다.(그림 5(a)). 서브볼륨을 만들기 위해서는 VLI의 VLICrop() API를 사용한다.

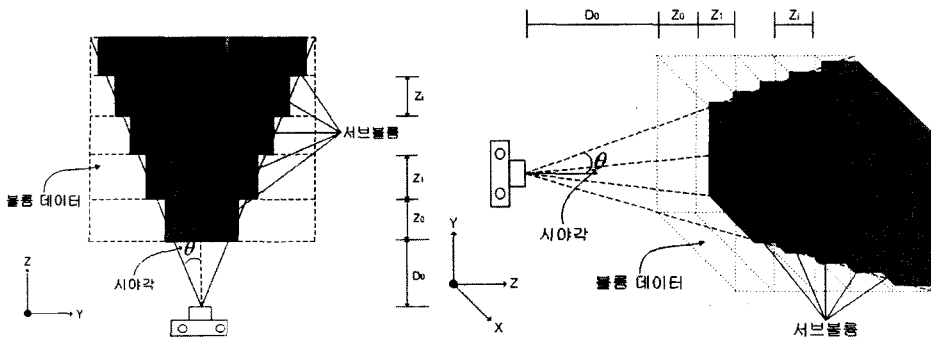


그림 4 관심영역 내의 서브볼륨들

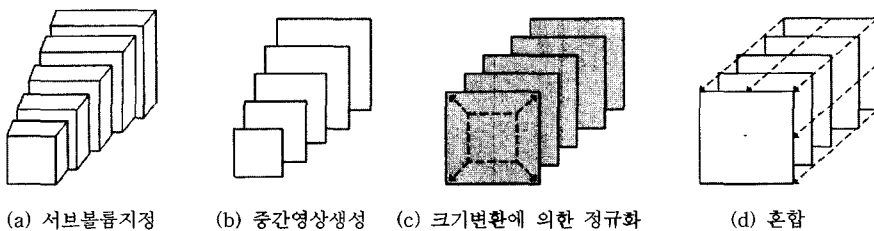


그림 5 서브볼륨을 이용하여 투시투영영상을 만드는 단계

$VLCrop(double\ inXmin,\ double\ inXmax,\ double\ inYmin,\ double\ inYmax,\ double\ inZmin,\ double\ inZmax,\ int\ inFlags = kSubVolume);$

(2) 서브블록을 생성하는 API

하나의 서브블록을 정의하기 위해서는 X, Y, Z축 방향에서 각각 최소값과 최대값인 $inXmin, inXmax, inYmin, inYmax, inZmin, inZmax$ 을 설정하면 된다. 즉 관심영역안의 블록 영역만을 정확하게 지정할 수 있다.

관심영역에 해당하는 서브블록을 지정하기 위한 매개 변수들은 다음과 같이 계산할 수 있다. i 번째 서브블록을 지정하기 위해 카메라로부터 해당 서브블록의 거리 D_i , 측면면의 폭 $Width_i$ 과 높이 $Height_i$ 는 다음과 같이 계산할 수 있다.

$$D_i = D_0 + i\Delta Z \quad (3)$$

$$Width_i = 2 \times \tan(\theta) \times (D_i + \frac{\Delta Z}{2}) \quad (4)$$

$$Height_i = 2 \times \tan(\gamma) \times (D_i + \frac{\Delta Z}{2}) \quad (5)$$

여기서 θ 와 γ 값은 시야평면(view plane)에서 수평, 수직 방향의 시야각(FOV angle)이다. (3)~(5)를 이용하여 서브블록들의 X, Y, Z축의 최소·최대값을 계산하여 원하는 서브블록을 만들 수 있다. (6)은 i 번째 서브블록을 만들어내기 위한 $VLCrop()$ API의 인스턴스이다. 여기서 $X_{max}, Y_{max}, Z_{max}$ 는 각각 전체블록의 크기이다.

$$VLCrop(X_{max}/2 - Width_i, X_{max}/2 + Width_i, Y_{max}/2 - Height_i, Y_{max}/2 + Height_i, D_i, D_i + \Delta Z) \quad (6)$$

두 번째 단계에서는 각각의 서브블록들을 렌더링하여 중간 영상을 만들어 낸다(그림 5(b)). 그림 6은 중간영상을 만들어내는 과정을 보여준다.

다음 단계에서는 자 서브블록별로 생성된 중간 영상들을 정규화한다(그림 5(c)). 초점이 멎히는 위치(초점거

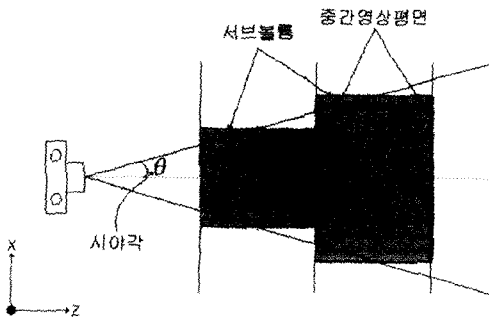


그림 6 중간영상생성

리)의 중간영상을 기준으로 하여 초점거리보다 가까운 곳에 위치한 중간 영상들은 확대하고, 초점거리보다 먼 곳에 위치한 중간 영상들은 축소한다. 슬랩을 사용하는 경우에는 중간 영상을 정해진 크기로 크기변환 후 최종 영상에 기여하는 부분을 잘라내야 하지만 서브블록의 경우는 생성되는 중간영상에 해당하는 모든 픽셀들이 최종 영상 생성에 기여하므로 별도의 잘라내기 작업이 필요하지 않다.

중간 영상들이 최종 영상크기로 크기변환되고 나면, 마지막으로 그래픽스 하드웨어가 제공하는 혼합기능으로 각각의 중간 영상들을 혼합하여 최종 영상을 만들어 낸다(그림 5(d)).

4. 성능분석

4.1 시야각과 렌더링 속도의 관계

그림 7에서 보느냐와 같이 슬랩을 이용하는 경우 카메라가 물체에 근접할수록, 시야각이 좁을수록 불필요하게 렌더링하는 부분들이 증가하여 전체적인 영상 생성 시간이 증가한다. 또한 시야각이 좁아질수록 중간 영상의 확대비율이 커지기 때문에 확대된 중간 영상을 저장하기 위해 상당량의 기억공간이 소요된다. 이중 대부분은 최종 영상과 무관하기 때문에 불필요한 메모리 사용이 발생하게 된다. 하지만 서브블록을 이용하는 경우에는 시야 절두체에 해당하는 부분만을 렌더링을 하기 때문에 시야각에 따라 렌더링 속도가 가변적으로 변하게 된다.

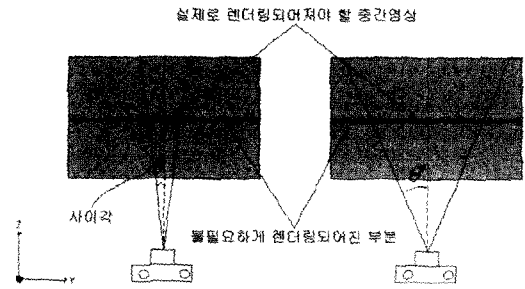


그림 7 슬랩 이용시 시야각의 증가에 따른 효과

i 번째 슬랩에서 불필요하게 낭비되는 면적 R_i 을 계산하면 다음과 같다. R 은 전체 블록에서 불필요하게 낭비되는 영역이다.

$$R_i = (X_{max} - Width_i) \times (Y_{max} - Height_i) \quad (7)$$

$$R = \sum_{i=0}^{n-1} R_i \quad (8)$$

4.2 서브블록 생성 기준과 두께가 렌더링 속도와 화질에 미치는 영향

서브블록 생성 기준과 두께는 렌더링 속도 및 화질과 깊은 관계가 있다.

그림 8(a)는 서브블록 생성 기준을 서브블록의 뒷면으로 지정하였을 경우이다. 불필요하게 렌더링되는 부분이 상대적으로 증가하여 렌더링 속도가 늦어진다. 이 경우 렌더링에서 누락되는 부분이 없어서 화질은 좋아지나 그림 8(b)과 그림 8(c)의 경우보다 렌더링 속도는 늦어진다. 그림 8(b)은 서브블록 생성 기준을 서브블록의 앞면으로 지정하였을 경우이다. 불필요하게 렌더링이 되는 부분이 없어 상대적으로 렌더링 속도는 빨라지지만 시각 절두체에 포함되는 많은 부분이 렌더링 단계에서 제외되어 최종 영상이 부자연스럽게 될 수 있다. 그림 8(c)은 서브블록 생성 기준을 서브블록의 중간부분으로 지정하였을 경우이다. 불필요하게 렌더링 되는 부분과 시각 절두체에 포함되지만 렌더링 단계에서 제외되는 부분을 모두 가지고 있지만 그림 8(a)에 비해서 상대적으로 렌더링 속도가 빠르고, 그림 8(b)에 비해서 상대적으로 렌더링 단계에서 제외되는 부분이 적다.

그림 8(c)와 그림 8(d)은 서브블록 두께와 렌더링 성능과의 관계를 보여준다. 서브블록의 생성 기준이 중간일 때 i 번째 서브블록에서 불필요하게 렌더링 되는 블록 UR_i 을 계산하면 (10)과 같다. UR 은 전체블록에서 불필요하게 렌더링되는 부분이다. $SubNum$ 은 서브블록의 개수이다.

$$SubNum = \frac{Z_{max} - Z_{min}}{\Delta Z} \quad (9)$$

$$UR_i = \frac{1}{8} \times \tan(\theta) \times \tan(\gamma) \times (\Delta Z)^3 \quad (10)$$

$$UR = \sum_{i=1}^{SubNum-1} UR_i \quad (11)$$

슬랩 기반 방법에서는 슬랩 두께와 상관없이 모든 부분을 렌더링 한다. 하지만 서브블록의 경우는 (10)에서와 같이 서브블록의 두께가 얇아수록 불필요하게 렌더링하는 부분의 체적이 상대적으로 작아지게 된다. 따라

서 서브블록의 두께가 얇아지면 불필요한 렌더링에 소모되는 시간도 함께 감소한다. 그러나 서브블록의 두께가 얇아지면 전체적으로 블록의 개수가 증가하게 되므로 크기변환하고 혼합할 중간 영상의 개수도 늘어나므로 전체적인 렌더링 시간이 늘어난다. 결국 서브블록 두께와 렌더링 시간과는 상충관계에 있다.

화질의 경우는 슬랩을 사용하는 경우나 서브블록을 사용하는 경우 모두 두께가 얇아수록 투시투영 영상에 가깝게 된다. 슬랩이나 서브블록의 두께를 1 복셀로 하면 투시 투영에 가장 가까운 영상을 만들어 내게 되고 두께를 전체블록의 크기와 같게 할 경우는 직교 투영 영상이 만들어진다.

4.3 서브블록 개수에 따른 렌더링 시간

그림 2를 참조하여 슬랩을 이용하여 렌더링할 경우 전체 렌더링 시간 비용(T_{slab})을 계산하면 다음과 같다.

$$T_{slab} = t_{sg}^{sl} + t_{im}^{sl} + t_{no}^{sl} + t_{cl}^{sl} + t_{bl}^{sl} \quad (12)$$

t_{sg}^{sl} : 시각 절두체에 맞게 설정된 슬랩 생성 시간 비용

t_{im}^{sl} : 중간 영상 생성 시간 비용

t_{no}^{sl} : 생성된 중간영상에 대한 정규화 시간 비용

t_{cl}^{sl} : 정규화된 중간영상을 잘라내는 시간 비용

t_{bl}^{sl} : 잘라낸 영상을 혼합하는 시간 비용

그림 5를 참조하여 서브블록을 이용하여 렌더링할 경우 전체 렌더링 시간 비용(T_{sub})을 계산하면 다음과 같다.

$$T_{sub} = t_{sg}^{sv} + t_{im}^{sv} + t_{no}^{sv} + t_{bl}^{sv} + t_{ur}^{sv} \quad (13)$$

t_{sg}^{sv} : 시각 절두체에 맞게 설정된 서브블록 생성 시간 비용

t_{im}^{sv} : 중간 영상 생성 시간 비용

t_{no}^{sv} : 생성된 중간영상에 대한 정규화 시간 비용

t_{bl}^{sv} : 정규화된 중간영상을 혼합하는 시간 비용

t_{ur}^{sv} : 불필요하게 렌더링되는 블록의 시간 비용

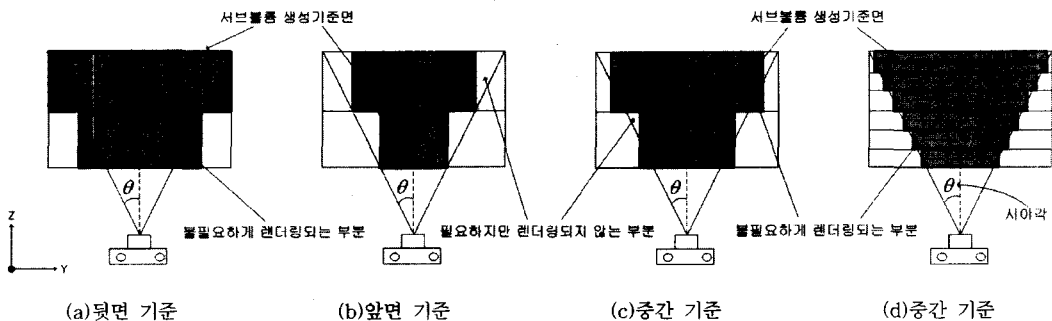


그림 8 서브블록 지정기준과 두께와의 렌더링 성능 관계

식 (13)에서 t_{sw}^{st} 의 경우 (11)의 불필요하게 렌더링되는 영역 UR의 처리 시간 비용이다. 블룸프로 하드웨어는 블룸 렌더링을 위한 전용의 하드웨어이기 때문에 슬랩 생성 비용과 서브블룸 생성 비용은 시간 비용은 차이가 없다($t_{sw}^{st} = t_{sw}^{sb}$). 슬랩을 이용하여 중간 영상을 생성할 경우 시각 절두체 내부에 있는 블룸 영역이 아닌 전체 영역에 대해서 중간 영상을 생성하기 때문에 중간 영상 비용이 서브블룸을 이용하여 중간 영상을 생성하는 비용보다 크다($t_{im}^{st} > t_{im}^{sb}$). 서브블룸을 이용하여 정규화를 할 경우 시각 절두체 크기만큼만 정규화 과정을 거치는 반면 슬랩을 이용할 경우 전체 블룸 크기를 기준으로 만들어진 중간 영상을 정규화하기 때문에 슬랩을 이용한 방법에서의 정규화 시간 비용은 서브블룸을 이용하여 정규화하는 시간 비용보다 크다($t_{no}^{st} > t_{no}^{sb}$). 혼합 과정은 시각 절두체에 맞게 생성된 영상을 혼합하는 것이므로 슬랩을 이용하여 혼합하는 비용과 서브블룸을 이용하여 혼합하는 비용은 차이가 없다($t_{bl}^{st} = t_{bl}^{sb}$). 슬랩을 이용할 경우 정규화된 중간 영상을 잘라내는 과정을 거쳐야 한다. 시각 절두체 앞쪽에 놓이는 중간 영상인 경우 뒤쪽에 놓이는 중간 영상보다 상대적으로 많은 양의 잘라내기 과정을 거쳐야 한다. 반면 서브블룸을 이용할 경우 생성 기준에 따라 불필요하게 렌더링되는 부분이 존재하지만 (11)에 의해 생성된 서브블룸에서는 모두 동일한 양이 존재한다. 슬랩을 이용할 경우 잘라내는 과정은 소프트웨어적인 처리를 해야 하는 반면 서브블룸을 이용할 경우 비록 불필요하게 렌더링되는 부분이 존재는 하지만 블룸프로 하드웨어를 이용하여 처리하기 때문에 슬랩을 이용하여 정규화된 중간영상을 잘라내는 시간 비용은 서브블룸을 이용하여 불필요하게 렌더링되는 블룸의 시간 비용보다 크다($t_{cl}^{st} > t_{cl}^{sb}$). 결과적으로 서브블룸에서의 렌더링 시간 비용(T_{sub})은 슬랩에서의 렌더링 시간 비용(T_{slab})보다 작다($T_{sub} < T_{slab}$).

5. 실험

본 논문에서는 256MB의 복셀메모리를 가진 VolumePro 500 모델을 사용하였다. VLI API의 버전은 1.2이고 OpenGL 버전은 1.1이다. 실험에 사용한 PC는 펜티엄IV 1.7GB, 1GB Main Memory, GeForce 2 MX 그래픽카드를 사용하였다. 또한 테스트용 실험데이터는 256×256×256 해상도로 복부를 촬영한 CT 데이터를 사용하였다.

그림 9의 좌측은 슬랩을 이용한 것이고, 그림 9의 우측은 서브블룸을 이용한 것이다. 최종 영상은 모두 두께가 32개의 복셀 크기를 갖도록 설정하였다. 시야각은 30

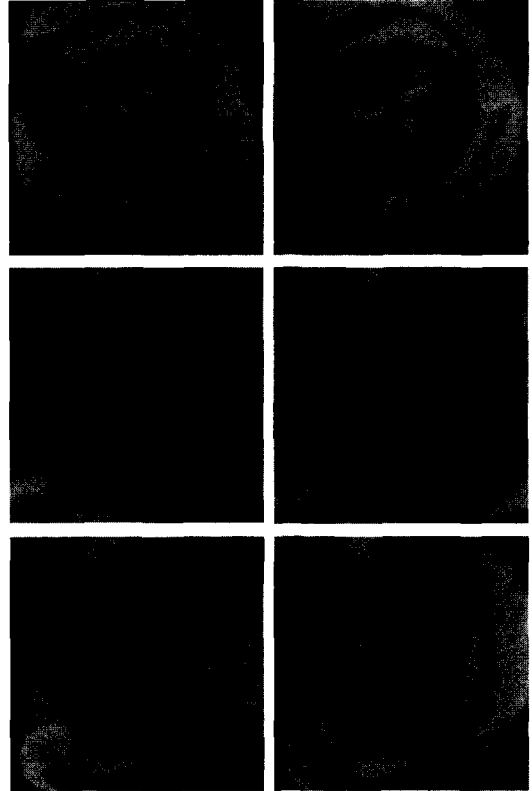


그림 9 슬랩기반 방법으로 얻은 영상(좌측)과 서브블룸 기반으로 얻은 영상(우측)

도이고, 서브블룸 생성 기준은 중간부분으로 하였다. 영상의 해상도는 400×400이고, Z축 방향으로 2배의 슈퍼샘플링(super sampling)한 것이다. 두 가지 방법을 적용했을 때 화질의 차이는 없음을 볼 수 있다.

표 1은 슬랩을 이용한 경우와 서브블룸을 이용한 경우의 최종 영상을 만들 때까지 시간을 측정한 것이다. 표 1을 보면 서브블룸을 이용한 방법이 슬랩을 이용한 방법보다 속도가 빠른 것을 알 수 있다. 직교투영을 이용하여 투시투영을 모사하는 경우에 오차를 가능한 줄여 좀 더 현실감 있는 투시투영의 효과를 얻기 위해서는 슬랩이나 서브블룸의 두께를 얇게 설정해야 한다. 표 1에서 보듯이 현실감 있는 투시투영효과를 가져올 수 있는 두께인 20~90 사이에서는 슬랩을 사용하는 경우보다 서브블룸을 사용한 경우 처리시간이 단축되었음을 알 수 있다.

표 2는 시야각에 따른 처리 속도의 변화를 보여준다. 표 2에서 보듯이 시야각이 좁음에 따라 전체적으로 처리해야 할 서브블룸의 크기가 줄어들어 처리속도가 약 11msec 정도 빨라진다. 상대적으로 시야각이 넓은 경우

표 1 두께에 따른 처리 속도의 변화 (시야각이 20도인 경우)

방법 \ 두께(복셀)	256	120	90	70	60	40
A. 슬랩(sec)	0.117	0.156	0.167	0.197	0.234	0.313
B. 서브볼륨(sec)	0.047	0.078	0.117	0.153	0.198	0.280
B/A (%)	59.9	50.1	29.8	22.3	15.4	10.5

방법 \ 두께(복셀)	30	20	10	5	2	1
A. 슬랩(sec)	0.392	0.553	1.063	2.086	5.128	10.124
B. 서브볼륨(sec)	0.353	0.514	1.020	2.042	5.081	10.063
B/A (%)	10.0	7.2	4.0	2.1	0.9	0.6

표 2 시야각에 따른 처리 속도의 변화 (두께가 90복셀인 경우)

방법 \ 각도(°)	1	3	5	10	20	30	40	45
서브볼륨(msec)	110.2	130.0	113.9	115.1	117.0	118.7	120.1	120.9

표 3 서브볼륨 생성 기준에 따른 처리 속도의 변화 (시야각이 20도인 경우)

생성기준 \ 두께(복셀)	90	70	60	40	30	20
뒤(msec)	126	161	205	288	362	522
가운데(msec)	117	153	198	280	353	514
앞(msec)	110	144	190	273	345	507

에는 처리해야 할 서브볼륨의 크기가 커지기 때문에 렌더링 처리 속도가 늦어진다.

표 3은 그림 8에서의 서브볼륨의 생성 기준에 따른 처리 속도의 변화를 보여준다. 표 3에서 보듯이 서브볼륨의 생성기준이 뒤쪽에 있을 경우 앞쪽에 있을 때보다 처리해야 할 서브볼륨의 크기가 커지기 때문에 렌더링 처리 속도가 늦어진다. 평균 약 16msec 정도의 처리 속도 차이를 보인다. 볼륨프로 하드웨어는 직교투영시 한 프레임을 생성하는데 30msec 정도가 소요되므로 서브볼륨 생성기준에 따른 속도변화는 전체적인 렌더링 속도를 좌우하는 중요한 요소가 된다.

6. 결론

볼륨프로는 직교투영만 가능하기 때문에 투시투영이 필요한 응용분야에는 적용하기 어렵다. 볼륨프로가 지원하는 절단면 기능을 이용하여 투시투영을 모사하는 방법이 소개되었으나 이 방법은 시각 절두체에 포함되지 않는 부분까지 렌더링을 하므로 처리시간이 길어지는 단점을 가지고 있다. 이것을 보완하기 위해 본 논문에서는 볼륨프로의 서브볼륨 기능을 이용하여 투시 투영을 모사하는 효율적인 방법을 제안하고 실험을 통하여 확인하였다. 현실감 있는 투시투영의 효과를 얻기 위하여 슬랩이나 서브볼륨의 두께를 얇게 설정하고 시야각이 좁은 경우에 서브볼륨을 이용하는 것이 슬랩을 이용하

는 경우보다 효과적이라는 결론을 얻었다. 이는 가상내시경이나 다른 의료영상부분에서와 같이 깊이 원근감이 중요한 경우에는 처리시간을 단축하는 결과를 얻을 수 있다.

향후 연구는 관측 방향이 기준축과 평행하도록 고정된 경우가 아니라 임의의 방향을 가리키는 경우에도 렌더링하는 방법을 고안하는데 집중될 것이다. 이를 해결하기 위해 기준 축을 중심으로 임의의 방향으로 회전시와핑(warping)을 하여 일정 범위 내에서는 관측방향을 자유롭게 조절할 수 있는 방법을 연구 중에 있다.

참 고 문 헌

- [1] Yagel, R., "Volume Viewing: State of the Art Survey," SIGGRAPH 97 Course Note 31, 1997.
- [2] Eric B. Lum, Kwan-Liu Ma, and John Clyne "A Hardware-Assisted Scalable Solution for Interactive Volume Rendering of Time-Varying Data," IEEE Transactions on Visualization and Computer Graphics, 8(3):286-301, July-September 2002.
- [3] Pfister, H., Hardenbergh, J., Knittel, J., Lauer, H. and Seiler, L., "The VolumePro Real-Time Ray-Casting System," Proceedings of SIGGRAPH 99, pp. 251-260, Los Angeles, CA, 1999.
- [4] A.Vilanova, R.Wegenkittl, A Konig and E.Groller. Mastering Perspective Projection through Parallely Projected Slabs for Virtual Endoscopy. SCCG'01-Spring Conference on Computer Gra-

- phics, pages 287-295, 2001.
- [5] Lorensen, W. E. and Cline, H. E., "Marching Cubes : A High Resolution 3D Surface Construction Algorithm," Computer Graphics, Vol. 21, No. 3, pp. 163-169, 1987.
 - [6] Levoy, M., "Display of Surfaces from Volume Data," IEEE Computer Graphics and Applications, Vol. 8, No. 3, pp. 29-37, 1988.
 - [7] Yagel, R. and Kaufman, A., "Template-based volume viewing," Computer Graphics Forum (Eurographics 92 Proceedings), pp. 153-167, Cambridge, UK, 1992.
 - [8] Lacroute P. and Levoy, M., "Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation," Computer Graphics (SIGGRAPH 94 Proceedings), Orlando, Florida, pp. 451-458, 1994.
 - [9] Westermann, R. and Eart, T., "Efficiently Using Graphics Hardware in Volume Rendering Applications," Computer Graphics, pp. 167-177, 1998.
 - [10] Yagel, R. Kaufman, A., Cabral, B., Cam, N. and Foran, J., "Accelerated volume rendering and tomographic reconstruction using texture mapping hardware," Symposium on Volume Visualization, pp. 91-97, 1994.
 - [11] Ma, K., Painter, J., Hansen, C. and Krogh, M., "A data distributed, parallel algorithm for ray-traced volume rendering," Proceedings of the 1993 Parallel Rendering Symposium, pp. 15-22, San Jose, 1993.
 - [12] Volume Library Interface User's Guide, Real Time Visualization. <http://www.rtviz.com>



임 석 현

1999년 2월 인하대학교 전자계산공학과, 물리학과 학사. 2001년 인하대학교 전자계산공학과 석사. 2001년 3월~현재 인하대학교 전자계산공학과 박사과정. 관심 분야는 블룸그래픽스, 하드웨어기반 렌더링, 의료 영상 처리, 영상기반 렌더링 등



신 병 석

1990년 서울대학교 공과대학 컴퓨터공학과. 1992년 서울대학교 대학원 컴퓨터공학과 석사. 1997년 서울대학교 대학원 컴퓨터공학과 박사. 2000년~현재 인하대학교 컴퓨터공학부 조교수. 관심분야는 실시간 렌더링, 블룸그래픽스, 의료영상