CUDA를 이용한 최대-최소 8진트리 생성 기법※

임종현, 신병석
인하대학교 컴퓨터 · 정보 공학과
daddybear@inha.edu, bsshin@inha.ac.kr

Min-Max Octree Generation Using CUDA

Jong-Hyeon Lim, Byeong-Seok Shin
Dept. of Computer Science and Information Engineering, Inha University

요 약

볼륨 렌더링은 볼륨 데이터로부터 유용한 정보를 추출하여 시각화 하는 방법이다. 일반적으로 볼륨 렌더링에서 사용하는 데이터가 크기 때문에 실시간 처리가 가능한 수준의 빠른 렌더링을 위한 가속기법들이 중요하다. 최대-최소 8진트리는 고속 볼륨 렌더링을 위한 자료구조이지만, 볼륨데이터가 클수록 생성시간이 오래 걸리는 문제가 있다. 본 논문에서는 CUDA를 이용하여 GPU에서 최대-최소 8진트리의 생성을 가속화 하는 방법을 제안한다. 먼저 볼륨데이터에 Space Filling Curve를 적용하여 3차원의 데이터를 연속적인 1차원 배열형태로 변환한다. 이렇게 변환된 데이터로부터 최대-최소 8진트리 자료구조를 만들어 빈공간 도약기법에 적용함으로써 렌더링 속도를 향상시킬 수 있다.

ABSTRACT

Volume rendering is a method which extracts meaningful information from volume data and visualizes those information. In general, since the size of volume data gets larger, it is very important to devise acceleration methods for interactive rendering speed. Min-max octree is data structure for high-speed volume rendering, however, its creation time becomes long as the data size increases. In this paper, we propose acceleration method of min-max octree generation using CUDA. Firstly, we convert one-dimensional array from volume data using space filling curve. Then we make min-max octree structures from the sequential array and apply them to acceleration of volume ray casting.

Keyword : GPU-based Volume Rendering, Min-Max Octree, CUDA

접수일자 : 2009년 09월 29일
심사완료 : 2009년 10월 22일
※ 이 연구는 인천정보산업진흥원이 주관하는 "실감형3D영상원천기술개발사업"의 지원으로 수행되었습니다.
※ 교신저자(Corresponding Author) : 신병석
주소 : 인천시 남구 용현동 253 인하대학교(402-751), 전화 : 032)860-7452, E-mail : bsshin@inha.ac.kr

1. 서 론

볼륨 렌더링은 3차원 볼륨 데이터를 가시화하기 위한 도구이다[1]. 하지만 일반적으로 볼륨데이터는 용량이 크기 때문에 렌더링 시간이 오래 걸리는 단점이 있다. 최근에는 GPU의 성능이 향상되면서 볼륨 렌더링의 가속화 연구가 중요한 이슈가 되고 있다. 최근 연산 능력이 매우 커진 GPU는 3차원 데이터를 매우 빠르게 가시화할 수 있도록 도와준다.

NVidia™에서 개발한 CUDA(Computer Unified Device Architecture)를 이용한 볼륨 렌더링 기법도 활발히 연구되고 있다[2,3]. CUDA는 기존의 그래픽연산처리에 특화되어 있는 GPU를 비그래픽스 분야에도 적용할 수 있도록 하는 병렬처리 아키텍처이다.

기존의 볼륨 렌더링 가속기법 중 대표적인 것으로 최대-최소 8진트리 기반의 빈공간 도약기법이 있다. 이 방법은 볼륨 데이터의 밀도값 중 특정블럭의 최대값과 최소값을 가지고 있는 8진트리를 생성하고, 이 데이터를 바탕으로 GPU에서 렌더링을 할 때 OTF(opacity transfer function)와 비교하여 투명하다고 간주되는 공간을 도약함으로써 렌더링 시간을 단축시키는 기법이다. 하지만 GPU에서는 8진트리를 생성할 수 없기 때문에, 이 방법은 GPU기반 볼륨렌더링에 적용하기 어렵다.

본 논문에서는 3차원 데이터를 연속적인 1차원 데이터로 변환해주는 SFC(space filling curve)와 CUDA를 이용함으로써, GPU에서 볼륨 렌더링 할 때 이용하는 최대-최소 8진트리를 빠르게 생성하는 방법을 제안한다. 이 방법은 기존에 제안된 8진트리 생성방법[3]을 기반으로 하고 있으며 두 개의 8진트리를 만들어서 하나에는 각 블록의 최대값을 저장하고 나머지 하나에는 최소값을 저장한다.

2장에서는 볼륨 렌더링을 가속화 하는데 사용되는 여러가지 알고리즘들을 소개하고, 3장에서는 CUDA를 이용하여 8진트리를 생성하고 렌더링 하는 방법을 기술한다. 4장에서는 기존의 방법과 비교한 결과를 보이고 결론을 맺는다.

2. 관련 연구

볼륨 광선 투사법(volume ray casting)은 가장 높은 화질의 영상을 생성하는 볼륨 렌더링 방법이다[1]. 영상의 각 화소에서 가상의 광선을 발사하여, 광선 상의 샘플점에서 색상과 투명도 값들을 구한 후 이들을 누적하여 최종 색상값을 결정한다. 기본적인 볼륨 광선 투사법은 빈 공간에서 도 매번 투명도를 계산하여 진행하였기 때문에, 가속화를 위해서 이러한 빈 공간을 빠르게 도약하는 방법들이 많이 제안되었다.

거리-맵 기반 방법은 볼륨 내부의 각 지점에서 가장 가까운 비투명 복셀까지의 거리를 저장하는 방법이다. 이 방법은, 광선이 볼륨 영역을 탐색할 때, 저장되어 있는 거리만큼 공간 탐색의 간격을 크게 하여 공간 탐색의 효율을 향상시킨다[4].

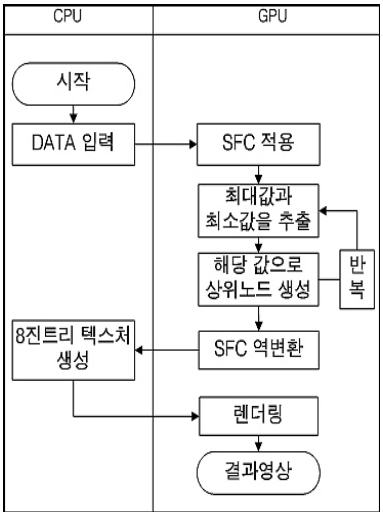
8진트리를 이용하는 방법도 있다[5,6]. 계층화된 자료구조를 이용하여 상위 노드에서 빈 공간을 발견하면 그 하위 노드들은 검색 하지 않고 건너뛸 수 있다. 볼륨 렌더링을 위해서 8진트리의 각 블록들을 대표하는 최대 밀도값과 최소 밀도값을 저장하는 방법이 있다. 이는 샘플점이 특정 블록에 놓이게 되면 사용자가 원하는 밀도값 범위와 해당 블록에 저장되어 있는 최대, 최소 밀도값을 비교하여 사용자가 정한 범위 안에 있지 않으면 해당 블록을 투명하다고 간주하고 해당 블록 경계까지 도약한다. 최대-최소 8진트리는 구조가 간단하여 볼륨 광선 투사법에 많이 사용된다.

최근 연산 능력이 커진 GPU는 3차원 볼륨 데이터를 매우 빠르게 가시화할 수 있도록 한다. 이에 따라 GPU를 활용하여 볼륨 렌더링의 속도를 가속화 시키는 방법들이 제안되었다[7,8]. 이 기법은 GPU 내부의 그래픽스에 특화된 병렬 처리 장치인 정점 셰이더와 픽셀 셰이더를 사용한다.

3. 4차원 볼륨 데이터의 효율적인 렌더링

볼륨 렌더링에서는 대용량데이터를 처리하므로, 일반적으로 연산속도가 매우 느리다. 따라서 실시간으로 변화하는 데이터를 렌더링하기 위해서는 효과적인 가속기법이 필요하다. 대표적인 가속기법으로 최대-최소 8진트리를 이용하는 방법이 있다. 이 자료구조는 전처리 단계에서 원본 볼륨데이터로부터 생성한다. 입력데이터가 고정된 경우에는 한번만 자료구조를 생성하면 되지만, 그렇지 않을 경우 8진트리를 반복적으로 생성해야 하므로 속도가 느려진다.

SFC는 3차원 입력데이터를 연속적인 1차원 배열로 변환하여 재배치시켜주는 알고리즘이다[9]. 이 알고리즘을 이용하면 병렬처리 시스템 상에서 트리 구조를 상향식으로 생성시킬 수 있다. 본 논문에서는 병렬처리 시스템에서 최대-최소 8진트리를 효과적으로 생성하기 위해 SFC를 적용하였고, 그 뒤 8진트리 자료구조를 구성하였다.



[그림 1] 본 논문에서 제안하는 방법의 처리절차

3.1 Space Filling Curve

SFC는 3차원의 자료구조를 1차원으로 변환해주는 방법 중 하나로, 정렬방식에 따라 몇가지로 구분한다[9]. Hilbert 곡선과 Z방향곡선은 공간 채우기 곡선 알고리즘의 여러 가지 기법 중 대표적인 기법이다. 여기서는 3D 데이터에 적용하였을 때, 단순한 사이클에 의해 정렬하는 Z방향 곡선을 사용하였다.

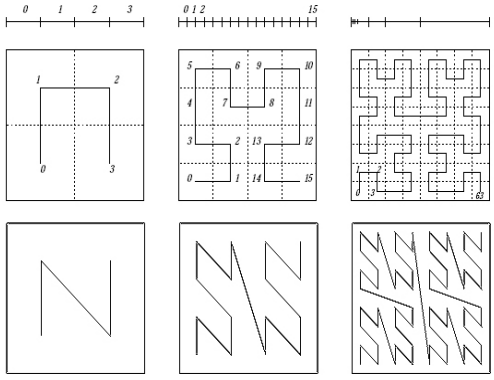
Z방향 곡선은 다음과 같은 방법으로 처리된다. 3차원 공간에서 임의의 좌표 $A=(X,Y,Z)$에 대해서 각 좌표값을 2진수로 변환했을 때, 다음과 같다.

$$A = \{x_1x_2x_3....x_k,\ y_1y_2y_3....y_k,\ z_1z_2z_3....z_k\}$$

아래와 같이 각 자리수를 연결하여 나열하면,

$$A' = x_1y_1z_1x_2y_2z_2x_3y_3z_3...x_ky_kz_k$$

가 되고, A′을 다시 10진수로 변환하면 변경될 좌표가 된다. 원본데이터의 모든 좌표에 대해서 위와 같은 식을 적용하면 1차원 좌표로 재배열된다. 이와 같이 배열형태로 재구성된 자료구조는 병렬시스템에서 최대-최소 8진트리를 생성할 때, 포인터를 사용하지 않고 연속적으로 읽고 쓸 수있게 함으로써 더 좋은 효율을 보인다. 그림2는 대표적인 SFC 기법 중 하나인 Hilbert 곡선과 Z방향 곡선을 보여준다. 그림에서 볼 수 있듯이, 입력데이터는 곡선의 종류에 따라 다른 순서로 재정렬된다.

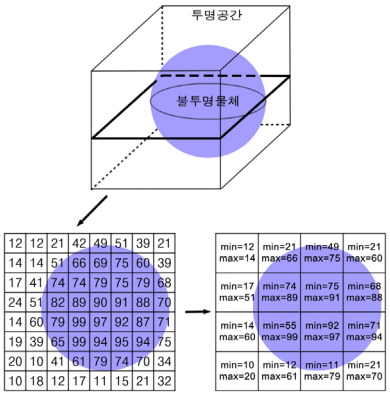


[그림 2] Hilbert 곡선(위)과 Z방향 곡선(아래)

3.2 최대-최소 8진트리 생성

최대-최소 8진트리는 각 블록마다 그 블록을 대표하는 최대 밀도값과 최소 밀도값을 저장시켜놓은 8진트리이다. 한 화소로부터 광선이 발사되면, 최대-최소 8진트리의 특정 블록과 만나게 된다. 이 때, 사용자가 원하는 영역의 밀도값과 블록 안에 저장되어 있는 최대 밀도값, 최소 밀도값을 비교하여 빈공간인지 여부를 확인하고, 광선이 현재의 샘플점이 위치하고 있는 블록의 경계까지 도약을 하게 된다. 하지만 이 방법은 원본 볼륨 데이터가 바뀌면 최대-최소 8진트리를 새로 생성해야 한다. CPU에서 8진트리를 생성 할 경우에는 실시간 처리가 어렵기 때문에, GPU에서 병렬처리를 하여 생성시간을 가속화 한다.

먼저 첫 번째 단계에서는, SFC에 의해 재정렬된 데이터를 읽어서 최하위노드들의 상위노드의 개수만큼 쓰레드를 생성한다. 생성된 각 쓰레드는 자신의 하위노드인 8개의 데이터를 읽어서 크기를 비교한다. 비교된 값들 중 가장 큰 값과 작은 값은 두 개의 자료구조를 만들어서 따로 저장한다. 두 번째 단계에서는 생성된 최대값 8진트리의 특정 노드에서 8개의 하위노드 데이터를 읽어서 그 중 최대값을 해당노드에 저장하며, 최소값 8진트리에서도 동일한 방법으로 최소값을 해당노드에 저장한다. 두 번째 단계는 렌더링부분에서 공간도약을 위해 필요한 레벨만큼 반복적으로 수행된다.



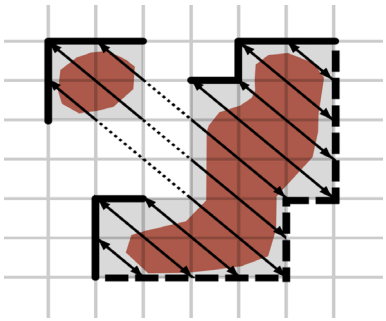
[그림 3] 최대-최소 8진트리

제안하는 방법에서는 기존 방법[7]대로 블록의 크기가 8^3이 될 때까지 연산을 반복하여 8진트리 자료구조를 생성하였다. 만약, 이 연산을 더 반복하여 8^3보다 작은 크기만큼 자료구조를 생성하고 빈공간 도약에 적용한다면 트리 탐색에 있어서 아직 제한적인 GPU의 연산능력의 한계 때문에 효율적이지 못하다[7]. 이렇게 반복수행하여 생성된 최대-최소 8진트리는 다시 CPU로 전송되며, 텍스쳐를 생성하여 전송된 값을 R과 G채널에 저장한다. 생성된 텍스쳐는 다시 GPU로 전송되어 렌더링 된다. 그림3에서는 볼륨렌더링에서 최대-최소 8진트리가 어떻게 적용되는지 보여준다.

3.3 렌더링

GPU기반의 볼륨렌더링은 [그림 4]와 같이 볼륨 텍스쳐 바운딩 박스의 후면부의 깊이를 구하고 전면부의 깊이와 차이를 계산하여 각 픽셀에서의 광선의 진행방향을 정한다. 이 방향값들은 2차원 텍스쳐에 저장된다. 그러고 나서 화살표와 같이 관측방향으로 단위거리만큼 이동하며 색상값을 샘플링하여 누적해나간다. 이 연산은 하드웨어 픽셀 셰이더에서 지원하는 픽셀 파이프라인 수만큼 병렬로 처리되기 때문에 속도가 빠르다.

먼저 GPU로부터 텍스쳐화 되어서 전송된 최대-최소 8진트리 자료구조는 볼륨데이터의 빈 공간에서는 샘플링하지 않도록 함으로써 렌더링속도를 향상시킨다. 광선이 진행될 때, 해당 샘플 포인트에서 OTF값과 최대-최소 8진트리에 저장된 밀도값을 비교한다. 이때 최대-최소값의 범위가 OTF의 비투명(non-transparent) 영역에 속하면, 계속 누적연산을 진행한다. 반면에, 투명한 영역에 속하면 샘플링을 하지 않음으로써 그만큼 렌더링 시간을 단축시킬 수 있다[5].



[그림 4] GPU기반의 볼륨 렌더링 방법

4. 실험 결과

본 실험은 Intel(R) Core™ 2Duo CPU E6400 2.00GHz에 2GB의 주 메모리를 갖는 시스템에서 수행되었다. 그래픽 카드는 512GB의 메모리를 갖는 NVidia GeForce™ 9800GT를 사용하였다.

[표 1]은 CPU와 GPU기반으로 각각 8진트리 생성하여 걸린 시간을 보여준다. 원본 볼륨데이터는 256^3과 512^3 크기를 갖는다. 8진트리 자료구조는 8^3 크기까지 생성하였다. 본문에서 제시한 방법은 CPU와 GPU간의 데이터 이동시간까지 고려한 결과이고 원본 볼륨 데이터의 크기에 따라 약 38%와 49%의 성능이 향상되었다. 이는 데이터의 크기가 작을 때는 생성시간의 차이가 크지 않아서 성능향상이 작지만, 대용량 볼륨 렌더링의 경우 8진트리의 생성시간이 상대적으로 크게 감소하여 더욱 효율적임을 의미한다.

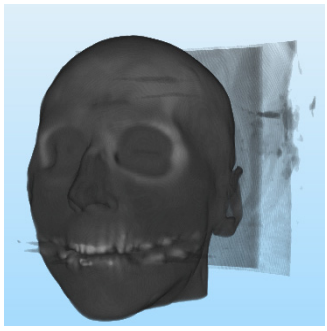
[표 1] CPU/GPU에서의 8진트리 생성시간 비교

데이터 크기	플랫폼	소요시간(sec)
256^3	CPU	0.70
	GPU	1.65
512^3	CPU	0.43
	GPU	0.83

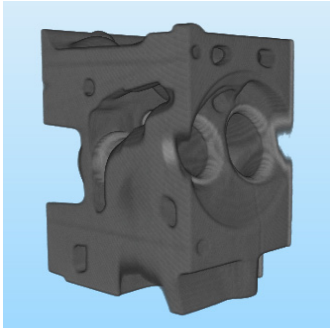
[표 2]는 본문에서 제안한 방법과 기존방법[7]의 렌더링시간을 비교한 것이다. [표 2]에는 볼륨 데이터의 크기에 따라 약 26%와 31%의 성능향상을 보였다. [그림 5]와 [그림 6]은 렌더링한 결과이며, 기존의 방법과 본 논문의 방법을 비교했을 때 차이를 보이지 않았다.

[표 2] 본문에서 제시한 방법과 기존의 GPU기반 렌더링과의 처리시간비교

실험방법	데이터 크기	소요시간(sec)
제안한 방법	256^3	0.562
	512^3	0.981
기존 방법	256^3	0.763
	512^3	1.441



[그림 5] 2563 크기의 head 데이터



[그림 6] 5123 크기의 엔진 데이터

5. 결 론

본 논문에서는 볼륨렌더링의 가속기법 중 최대-최소 8진트리 기법에서 8진트리 생성을 그래픽 하드웨어를 이용하여 빠르게 생성하는 방법을 소개하

였다. 기존에는 GPU에서 트리구조 생성이 불가능했지만, CUDA기반에서 SFC를 이용하여 볼륨데이터를 재배열한다음에 8진트리 구조를 생성하면 GPU에서도 가능하다. 본 논문의 방법은 GPU기반에서 8진트리 생성을 병렬로 처리하기 때문에, 기존의 CPU기반의 8진트리 생성에 비해 성능이 향상됨을 확인할 수 있었다.

참고문헌

[1] M. Leveoy, "Display of surface from volume data", IEEE Computer Graphics and Applications, Vol.8, No.3, pp. 29-37, 1998.

[2] NVidia, http://developer.download.nvidia.com/compute/cuda/2_0/docs/NVIDIA_CUDA_Programming_Guide_2.0.pdf

[3] Prekshu Ajmera, Rhushabh Goradia, Sharat Chandran, Srinivas Aluru, "Fast, Parallel, GPU-based Space Filling Curves and Octrees", I3D2008.

[4] R. Yagel and Z. Shi, "Accelaerating volume animation by space-leaping", IEEE Visualization 1993, pp. 62-69, 1993.

[5] J. Danskin and P. Hanrahan, "Fast algorithms for volume ray tracing", Volume visualization 1992, pp. 91-98, 1992.

[6] J. Wilhelms and A. V. Gelder, "Octree for faster isosurface generation," ACM Trans. on Graphics, Vol. 11, No. 3, pp. 201-227, 1992.

[7] J. Kruger and R. Westermann, "Accelation Techniques for GPU-based Volume Rendering", IEEE Visualization, pp. 38-46, 2003.

[8] H. Scharsach, "Advanced GPU Raycasting", CESCG, pp. 69-76, 2005.

[9] J K Lawder and P J H King, "Using Space-lling Curves for Multi-dimensional Indexing", Lectures Notes in Computer Science, Vol. 1832, pp. 20-35, 2000.

임 종 현(Jong-Heyon Lim)

2007년 2월 안양대학교 디지털미디어학부(학사)
2007년 ~ 현재 인하대학교 정보공학부(석사)

관심분야 : 볼륨 렌더링

신 병 석(Byeong-Seok Shin)

1990년 2월 서울대학교 컴퓨터공학과(학사)
1992년 2월 서울대학교 컴퓨터공학과(석사)
1997년 2월 서울대학교 컴퓨터공학과(박사)
2000년~현재 인하대학교 컴퓨터공학부 부교수

관심분야 : 실시간 렌더링, 볼륨 그래픽스, 의료 영상