볼륨렌더링을 위한 병렬화 방법들의 성능평가

°김진호, 김남규, 김지인, 정갑주
건국대학교 컴퓨터공학과

Performance Evaluation of Parallelization Methods for Volume Rendering

Jin Ho Kim, Nam Kyu Kim, Jee In Kim, Karp Joo Jeong
Department of Computer Science and Engineering, Konkuk University

요 약

많은 처리시간을 요구하는 대규모 3차원 데이터의 영상화(대규모 볼륨렌더링)에서는 병렬처리가 반드시 요구된다. 대규모 볼륨렌더링의 처리시간은 크게 데이터입력 시간과 입력된 데이터의 영상화 (연산) 시간으로 구성된다. 따라서 데이터 입력 과정과 연산 과정 모두를 병렬화할 필요가 있다. 입출력 병렬화 및 알고리즘 병렬화는 각각 독립적으로 적용가능 하다. 본 논문에서는 (1) 순차 볼륨렌더링, (2) 병렬연산 기반 볼륨렌더링, (3) 병렬입출력 기반 볼륨렌더링, (4) 병렬연산 및 병렬입출력 기반 볼륨렌더링 등 네 가지 경우를 각각 구현하여 성능을 비교하였다. 실험결과에서는 병렬연산 및 병렬입출력이 동시에 적용되는 (4)가 가장 좋은 성능을 보이는 것으로 나타났다.

1. 서론

볼륨 렌더링은 중간 단계를 거치지 않고 3차원 입체 영상을 컴퓨터 화면으로 직접 사상하여, 2차원의 영상을 만드는 기술로 과학 및 공학 등의 여러 분야에서 매우 중요한 역할을 하고 있다.[1] 그러나, 볼륨렌더링에 사용하는 볼륨 데이터의 크기가 크고 이에 따라, 방대한 양의 메모리와 방대한 처리시간을 요구한다. 따라서, 볼륨렌더링을 대화적인 응용프로그램에 적용하기 위해서 병렬화는 반드시 필요하다.[2,3] 대규모 볼륨렌더링의 처리시간은 크게 데이터입력 시간과 입력된 데이터의 영상화 (연산) 시간으로 구성된다. 따라서 데이터 입력의 병렬화 및 연산의 병렬화가 모두 요구된다. 그러나, 의료 영상 병렬 처리에서는 일반적으로 볼륨 데이터가 이미 메인 메모리에 적재되어 있다고 가정을 하고 연산의 병렬화만을 다루므로, 현실감이 부족하고, 대용량의 데이터의 경우 병렬시스템의 프로세싱 노드의 메모리 용량에 의하여 처리 가능한 입력 데이터의 크기가 제한되는 단점이 있다. 그런데, 입출력 병렬화 및 알고리즘 병렬화는 각각 독립적으로 적용가능 하다. 따라서 볼륨렌더링의 병렬화 전략은 크게 병렬연산, 병렬입출력, 병렬연산 및 병렬입출력 등 세 가지로 나눌 수 있다.

본 논문에서는 이 세 가지 전략과 순차 볼륨렌더링 등 네 가지 경우를 직접 구현하고 성능비교를 통해 각 방법을 평가한다. 본 연구를 위해 (1) Brute-Force 방식의 순차 볼륨렌더링 알고리즘을 구현했고, (2) 순차 알고리즘을 완전 데이터중복 방식에 의해 연산의 병렬화를 했고, (3) 순차 알고리즘에 병렬입출력을 적용했고, (4) 순차 알고리즘에 병렬연산과 병렬입출력을 모두 적용했다. 각각의 경우에 대해서 리눅스 클러스터 상에서 성능평가를 했다. 실험결과에서 병렬입출력과 병렬연산이 모두 적용된 경우가 가장 좋은 성능을 보였다.

본 논문 구성은 다음과 같다. 2절에서 적용된 병렬처리 기법 및 병렬입출력 기법을 설명한다. 3절에서 성능평가 결과를 제시하고 각 병렬화 방법을 평가한다. 4절에서는 결론과 향후 과제를 제시한다.

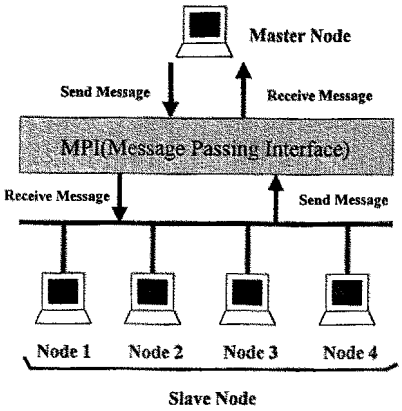
2. 병렬 처리 기법 및 병렬 입출력 기법

본 절에서는 알고리즘을 병렬화 한 기법과 입출력을 병렬화 한 기법에 대해서 설명한다.

2.1 병렬 처리 기법

볼륨렌더링 알고리즘은 방문 순서에 따라 이미지 순회(Image order), 오브젝트 순회(Object order), 혼합 순회(Hybrid order) 방식으로 나누어진다[3]. 본 연구에서는 이미지 순회 방식인 광선 추적법(Raycasting)을 기반으로 볼륨 렌더링 알고리즘을 구현하였다.[5]

볼륨 렌더링을 위한 병렬처리는 Master/slave 방식([그림1])을 택했고 MPI를 사용해서 구현하였다[6]. 주 노드(Master node)는 각 종 노드(Slave node)에 데이터와 처리량을 메시지로 배분해주고 처리된 결과를 메시지 통해 전달받은 다음 최종 결과를 출력한다.



[그림 1] 병렬처리 시스템 구조

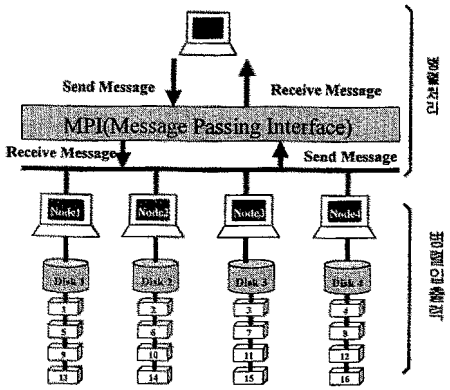
각 노드에 작업을 할당(Task Partition)하는 방법에는 이미지 분할(Image Partition) 방법과 오브젝트 분할(Object Partition) 방법이 있다[3]. 본 연구에서는 이미지 분할 방법을 사용하여 각 노드에 이미지의 정방형 타일(Rectangular tiles) 모양으로 분할하여 배분하였다. 각 노드에 대한 데이터 분산(Data distribution)은 병렬처리 도중에 통신부하가 가장 적도록 볼륨 데이터 전체를 복제하여 배분하였다.

2.2 병렬입출력 기법

방대한 볼륨 데이터를 메모리에 옮기기 위해서는 많은 시간을 요구한다. 특히, 이미지 분할 방법을 사용하여 병렬 처리를 하는 경우, 볼륨 데이터의 어느 부분을 각 노드에 할당해야하는 어려움이 있다. 이러한 문제는 공유메모리 기반의 병렬 머신을 사용하면 해결될 수 있다[3]. 그러나, 이러한 머신은 가격이 고가이다. 본 연구에서는 병렬입출력 기법을 사용해서 저가의 리눅스 기반 PC Cluster에서 이러한 문제를 해결하려 한다.

병렬입출력 기법은 각 노드에 분할되는 데이터 입출력으로 인해 생기는 속도 증가의 단점을 해결하기 위해, 입출력을 병렬화 하여, 데이터 입출력 속도를 향상시키는 기법이다.

본 연구에서는 광선 추적법 알고리즘이 가지는 데이터 참조 특성을 고려하여 병렬입출력 기법을 설계하였다. (1) 광선경로에 위치한 복셀들의 순서에 따라 순차적으로 데이터를 참조하는 특성(즉, *Data Prefetching*이 가능)을 가지고 있다. (2) 각각의 광선들은 독립적으로 처리될 수 있으면, 서로 가까운 거리에 위치한 광선들은 비슷한 시기에 처리될 가능성이 높은 특성(*Locality*)을 가지고 있다. 이러한 특성을 이용하여 데이터를 특정한 크기의 블록 단위로 분할하였고 이러한 블록들은 각 노드의 디스크에 순차적 할당 방법으로 저장하였다.(그림2 참조) [7]



[그림 2] 병렬입출력 시스템 구조

3. 알고리즘 성능평가

본 절에서는 4가지의 알고리즘에 대해서 간략하게 소개하고 각 알고리즘의 수행 속도를 제시하며 성능을 평가한다.

1) 순차 알고리즘(Sequential Algorithm)

Brute-Force 방식의 광선 추적법 알고리즘이다. 세이딩은 퐁세이딩(Phone Shading) 방법을 사용하였고 물질의 분류는 Isovalue Contour Surface 방법과 Region Boundary Surface 방법을 사용하였다.[4]

2) 병렬 알고리즘(Parallel Algorithm)

순차 알고리즘을 MPI를 사용하여 병렬화한 알고리즘이다.

3) 병렬입출력 순차 알고리즘(Parallel I/O Sequential Algorithm)

순차 알고리즘에 병렬입출력 기법을 적용한 알고리즘으로 데이터 입출력을 병렬처리 하였다.

4) 병렬입출력 병렬 알고리즘(Parallel I/O Parallel Algorithm)

병렬 알고리즘에 독립적으로 병렬입출력을 적용한 알고리즘이다.

본 연구는 리눅스 기반의 PC Cluster상에서 수행하였고 실험환경은 [표 1]과 같고 실험 데이터는 [표 2]와 같다.

	Master node(1대)	Slave node(4대)
CPU	450Mhz	350Mhz
메모리	256M	128M
LAN	10Mbs (Slow Ethernet Switch)	

[표 1] 시스템 사항

	데이터 해상도	데이터 크기
Syn256	256x256x256x4	약 64M
Bighead	512x512x128x4	약 128M

[표 2] 실험 데이터

순차알고리즘과 병렬입출력 순차알고리즘은 주 노드를 사용하였다. 병렬알고리즘과 병렬입출력 병렬알고리즘은 주 노드와 종 노드를 모두 사용하였다. 병렬알고리즘의 경우, 데이터는 주 노드의 디스크에 존재하고 종 노드들이 렌더링 초기단계에서 NFS를 통해 읽어 들이는 것으로 구현하였다.

전체 실험결과는 [표 3]과 [표 4]에 제시되었다. [표 4]에서 저속 LAN을 통한 데이터 입력이 엄청난 부하를 야기시킴을 보인다. [표 3]에 제시된 각 알고리즘의 성능을 살펴보면, 병렬입출력과 병렬연산이 동시에 사용되는 경우 저속의 통신속도에도 불구하고 상당한 성능향상을 보였다. 고속 LAN이 사용되는 경우 현격한 성능향상이 가능할 것으로 보인다.

알고리즘	노드수 / 데이터	1	2	3	4	5
순차알고리즘	Syn256	252				
	Bighead	1500				
병렬입출력 순차알고리즘	Syn256	570				
	Bighead	1669				
병렬알고리즘	Syn256	423	290	321	371	359
	Bighead	1724	1083	1012	1044	1019
병렬입출력 병렬알고리즘	Syn256	615	316	216	179	141
	Bighead	1804	1024	683	635	485

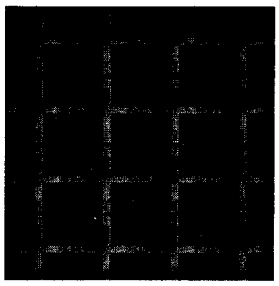
[표 3] 알고리즘 시간 (단위:초)

	병렬알고리즘					순차알고리즘
노드수 / 데이타	1	2	3	4	5	1
Syn256	163	169	233	307	308	8
Bighead	328	346	467	616	638	14

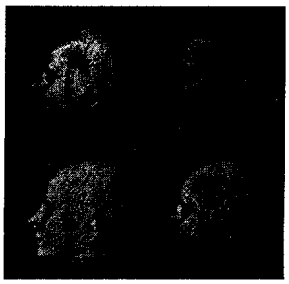
[표 4] 병렬 알고리즘과 순차알고리즘 데이터 입출력 시간 (단위:초)

각각의 결과를 분석해 보면 다음과 같다. 병렬알고리즘의 성능을 살펴 보면 속도가 향상되다가 다시 악화되는 것이 보인다. 이는 여러 종 노드들이 동시에 NFS를 통해 주 노드 디스크에서 데이터를 읽어 들이려고 시도함에 의해 발생하는 병목현상에 따른 부하 때문이다. 이는 *병렬연산에서 입출력 부하가 심각함*을 보여주는 결과이다.

병렬입출력 기법만 사용된 경우의 성능은 일부 실험과정의 실수로 두 개 이상의 노드들에 대한 결과가 부정확하게 얻어져 완전한 분석이 어렵지만 병렬연산과 병렬입출력이 동시에 적용된 경우를 볼 때 고속 LAN이 사용된다면 노드 수가 증가함에 따라 충분한 성능향상이 있을 것으로 보인다.
병렬연산과 병렬입출력이 동시에 사용된 경우, 5개 노드까지 지속적으로 속도 향상이 있어 노드 수가 증가하더라도 더 낳은 성능향상을 얻을 수 있을 것으로 예상된다. 이러한 성능향상은 병렬입출력으로 인해 데이터 입출력 부하가 여러 노드들로 분산된 때문이다. 특히 이러한 분산에 의해서 저속 통신망에 의한 병목현상도 해소시키는 효과를 얻을 수 있다.
따라서 병렬입출력을 사용하여 알고리즘을 병렬처리를 하는 경우 이상적으로 속도를 향상시킬 수 있다는 결론이 나온다. 특히 고속 LAN이 사용되는 경우 훨씬 더 많은 성능향상이 있을 것으로 기대된다.
[그림 4]의 Syn256 실험 데이터는 Isovalue Contour Surface 물질분류 방법을 사용하여 렌더링 하였고 [그림 5]의 Bighead 실험 데이터는 두 가지의 물질분류 방법을 사용하여, 위의 두 그림은 Region Boundary Surface 방법을 사용하고 아래의 두 그림은 Isovalue Contour Surface 방법을 사용하여 렌더링 하였다.



[그림 4] Syn256 (256×256)



[그림 5] Bighead (512×512)

4. 결론 및 향후과제

본 연구를 통해 병렬 볼륨렌더링 알고리즘은 병렬 연산 및 병렬입출력의 기법을 함께 사용하는 경우 가장 효과적인 성능향상이 가능함을 알 수 있었다. 향후과제로는 볼륨렌더링에 최적화 알고리즘(Octree, Early Ray Ttermination)등을 적용하고 이에 알맞은 병렬입출력 기법을 설계하여 구현할 예정이다.

참고문헌

[1] A Kaufman, editor. "Volume Visualization", IEEE Computer Society Press, 1991.
[2] Craig M. Wittenbrink. Survey of Parallel Volume Rendering Algorithms. Proceedings of International Conference on Parallel Distributed Processing Techniques and Applications, July 1998.
[3] P. Lacroute, "Real-Time Volume Rendering on Shared Memory Multiprocessors Using the Shear-Warp Factorization", Proceedings 1995 Parallel Rendering Symposium, ACM Press, Atlanta GA, October 1995, pp.15-22.
[4] Marc Levoy "Volume rendering, display of surfaces from volume data," IEEE computer Graphics and Applications, Vol 8, no. 5, pp29-37, May 1988.
[5] M. Levoy, "Efficient Ray Tracing of Volume Data," ACM Transactions on Graphics, Vol 9, No. 3, pp.245-261, May 1990.
[6] Message Passing Interface Forum. MPI: A Message-Passing Interface Stanard, 1.0 edition, May 5 1994.
[7] 김남규, 정갑주, "고성능 3차원 볼륨 렌더링을 위한 병렬입출력 시스템," Proceedings of 26th KISS Spring Conference, 1999.