재구성 Cell을 이용한 Photon mapping SIMD프로세서 설계

*류현우, 김영진, 이현수
경희대학교 컴퓨터공학과
e-mail : atner@khu.ac.kr, jerryjin@khu.ac.kr, leehs@khu.ac.kr

Photon Mapping SIMD Processor Design using Reconfigurable Cell

*Hyun-Woo Ryu, Young-Jin Kim, Hyon-Soo Lee
Dept. of Computer Engineering, Kyung Hee University

Abstract

The synthesis of the 3D images is the most important part of the virtual reality. The photon mapping is the best method for reality in the 3D graphics. This paper presents an architecture for photon mapping applications on SOC devices. The proposed architecture reduces the computation time to photonmap search and radiance estimation. Also this architecture is implemented by a SIMD processor which trades parallelism for frequency of operation.

I. 서론

3D 영상처리기법에서 질감을 높이기 위한 효과적인 방법으로 전역 조명처리 기법이 사용된다. 전역 조명처리 기법은 오브젝트 단위로 빛에 대한 투과, 굴절광을 고려하는 기법이다.

대표적인 전역 조명처리 모델은 Ray Tracing[4]과 Radiosity가 있다. Ray Tracing은 눈의 관점에서 방향을 역추적 하여 조명값을 구성하는 기법이며, 정반사에 대한 연산만 이루어지기 때문에 난반사 적용이 되지 않는다. 그리고 Radiosity는 사물의 표면에서 발산되는 빛 에너지를 Mesh 기반으로 처리하여 빛의 반사에 대한 처리를 하는 방법이다. 그러나 모든 난반사 요소들을 미리 계산해야 하므로 시점의 변화가 있을 경우 연산량이 급격하게 증가하는 단점을 갖는다.

위의 단점들을 해결하기 위해서 Photon mapping 모델이 제안되었다[2]. Photon mapping 모델은 빛의 입자인 Photon의 개념을 적용하고 있다. 광원에서 방출된 빛의 입자를 3차원 공간상의 물체와 충돌하여 난반사 처리를 수행 한다. 하지만 Photon mapping의 문제점은 Photon의 수가 증가함에 따라 광도 값 계산에 대한 연산이 기하급수적으로 증가한다. 그러므로 Photon mapping연산을 병렬적으로 수행할 수 있는 하드웨어 구조가 필요하다.

본 연구는 Photon mapping 단계에서 연산시간의 효율적인 처리를 위한 Cell 기반의 SIMD프로세서를 설계하였다.

제안한 구조는 16개의 Cell로 구성되어 있으며, Cell은 각각 3개의 연산 Unit을 가지고 있다. 또한 Cell들은 병렬로 연산을 수행하도록 설계하였으며 데이터 교환의 효율성을 높이기 위해서 4차원 Hypercube Network를 사용하였다.

II. Photon mapping

2.1 Photon mapping 시스템

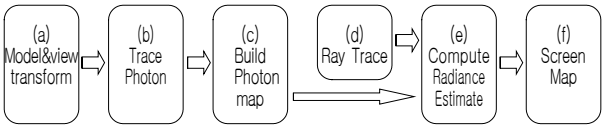


그림 1. Photon mapping을 적용한 3D영상처리 단계

그림1은 Photon mapping을 사용한 3D 영상처리의 단계이다.

첫 번째 단계는 3차원 좌표를 공간상에 오브젝트를 생성하고 두 번째 단계에서는 광원으로부터 생성된 Photon은 물체와 충돌시 충돌 좌표와 Photon power 값을 생성하게 된다. 세 번째 단계는 충돌좌표를 자료구조에 저장하게 되고 네 번째 단계에서 시점을 기준으로 화면을 구성하기 위한 좌표값을 생성한다. Photon map과 Ray Trace 정보를 이용하여 다섯 번째 단계에서 밀도 추정값을 구한 후 마지막 단계에서 3D 영상을 구할 수 있다[1].

6단계 중에서 Ray Trace 단계와 Radiance Estimaion 단계가 가장 시간이 오래 걸리는 연산이다. 그 중 Trace Photon은 조건 분기가 많이 발생하여 병렬성을 추출하기 어렵다. 그러나 Radiance Estimate은 여러 개의 Photon을 이용하여 밀도추정 계산을 하기 때문에 Process의 수가 늘어날수록 계산속도도 향상된다.[5] 그러므로 본 본 논문에서는 Radiance Estimation 단계에서 Photon map 검색과 밀도추정에 대하여 처리속도 향상방안에 대하여 연구하였다.

2.2 Photonmap 검색

광원에서부터 방출된 Photon들은 오브젝트와 충돌하면서 충돌 좌표와 광도값이 생성된다. 충돌된 Photon 정보들을 저장하기위해 Photon에 대한 정보를 블록 단위로 저장하는 방식[1]을 사용하였으며 Photonmap 검색의 방법으로 최근접 이웃 추정법을 사용하였다.

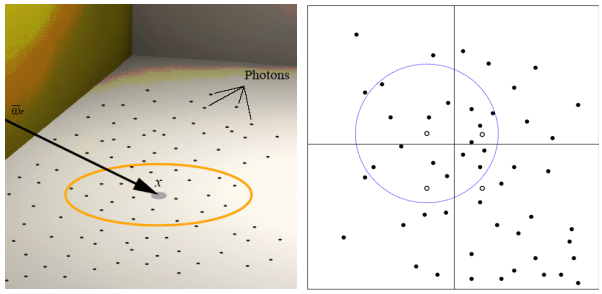
표 1. Photon search의 단계

Step 1 Photon map Search
- step① $(X_1-X_2)^2+(Y_1-Y_2)^2+(Z_1-Z_2)^2$ 값을 계산
- step② 해당 값을 r^2과 비교하여 원하는 photon을 검색

표1은 Photonmap 검색의 각 단계별 연산의 과정을 나타낸다. X_1,Y_1,Z_1은 Radiance Estimation 값을 구하기 위한 중심점 좌표이고 X_2,Y_2,Z_2은 Grid 안에 포함된 Photon 좌표이다.

Photon 검색의 단계에서 두 점간의 거리는 비교에 대한 결과 값만 필요하기 때문에 본 논문에서는 연산의 속도를 높이기 위해 제곱근을 제거하여 비교하는 방식을 사용하였다[2].

2.3 Radiance Estimation 알고리즘



(a) Radiance Estimation (b) 블록단위 Photonmap

그림 2. Radiance Estimation을 위한 블록단위 검색

Radiance Estimation은 photonmap에 저장되어 있는 photon과 오브젝트와의 충돌 정보를 이용하여 3차원 공간상의 광도 값을 구하는 방법이다.

Radiance Estimation을 구하기 위해서는 각 Photon의 광도 값에 대한 밀도를 적용해야 한다. 일반적인 방법으로는 최근접 이웃추정 법으로 Photon을 검색한 뒤 각 Photon 마다 Kernel을 적용시켜 Density Estimation을 구하는 방법을 사용한다[2]. Radiance Estiamtion 연산은 수식1과 같다.

$$f(x)=\frac{1}{nd_k(t)}\sum_{i=1}^{n}K\left(\frac{x-X_i}{hd_k(t)}\right) \qquad (1)$$

d : 관찰점x에서 k번째 지점까지의 거리
n : 근접 photon의 개수
h : smooth 매개변수
K : Kernel function

수식1의 Kernel function에 Rectangle[2]을 사용하였고 최근접 이웃 추정법을 적용하면 수식 2와 같다.

$$L_r(x,\vec{\omega}_r)\approx\frac{1}{\pi r^2}\sum_{p=1}^{N}\Delta\Phi_p(x,\vec{\omega}_p) \qquad (2)$$

r : 밀도 추정 값을 구하고자 하는 범위의 반지름
N : 근접 photon의 개수
$\Delta\Phi$: Photon power

III. 제안한 Cell기반 SIMD프로세서의 설계

3.1 제안한 시스템 구조

제안한 시스템은 Cell 기반의 SIMD 구조로 구성된다. 16개의 Cell은 명령어에 따라 연산이 재구성되며 Hypercube Network를 통하여 데이터가 전달된다. 각 Cell은 Photon에 대한 좌표값과 photon power을 저장하기 위해서 Register File을 사용하였다.

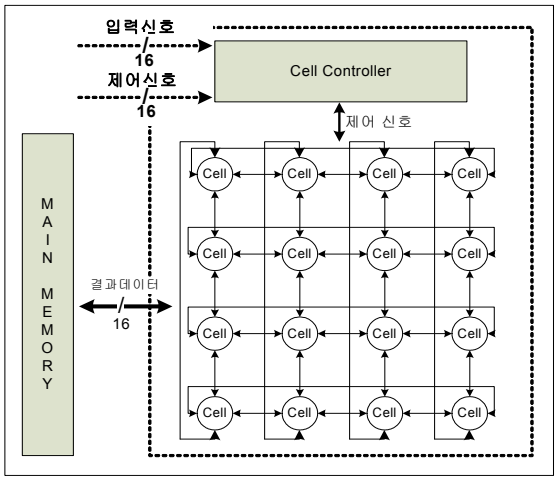


그림 3. 제안한 전체 시스템 구조

3.2 CELL 구조

Cell은 제어신호에 따라 표2와 같은 명령을 수행한다. Photon map 검색과 Radiance Estimation 연산을 할 때 연산에 따라 동적으로 재구성된다.

표 2. CELL 명령의 종류

Instruction	설 명
A+B	덧셈연산
$(A-B)^2$	photon 간의 거리연산에 이용
if A>B, output 0	중심점과의 범위초과 여부에 이용

하이퍼큐브 네트워크 연결에 따라 인접노드에서 들어온 데이터는 ALU에서 연산을 한 뒤 Register File에 저장하거나 인접 노드에 전송하게 된다.

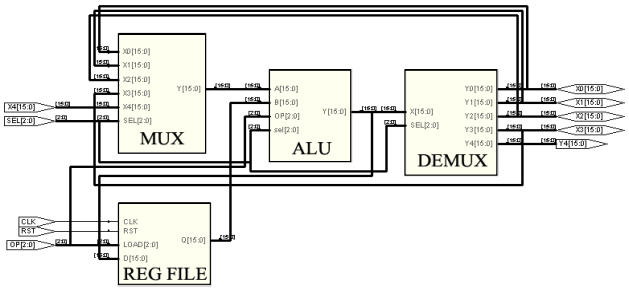


그림 4. Cell 구조

모든 명령어는 한 클럭에 연산이 수행되며 16개의 Cell은 1 Iteration 동안 4개의 Photon처리를 할 수 있다.

3.3 CELL간의 Interconnection Network

Cell간의 연결을 Hypercube Network로 연결하였고 16개의 Cell은 동시에 4개의 Photon에 대한 계산을 수행할 수 있다. 그림 5는 Photonmap의 검색연산을 수행하기 위한 각 노드와 해당 노드의 데이터를 나타낸다.

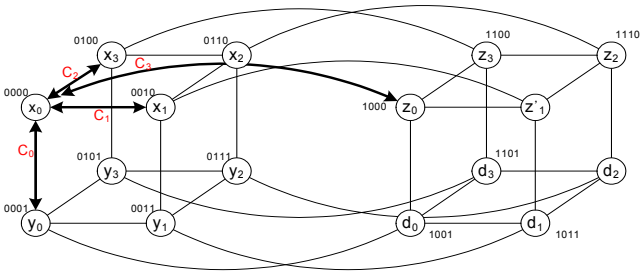


그림 5. 4차원 Hypercube Network

4차원 Hypercube로서 하나의 노드에 4개의 인접 노드와 연결되어 있고 연결 네트워크는 인덱스에 각 비트를 1개씩 변환하면 x,y,z,w 축으로 4가지 연결 경로를 갖으며 각 노드는 양방향으로 전송이 가능하다.

3.4 Photonmap 검색과 밀도추정 연산과정

Photon mapping의 Search와 밀도추정 연산을 수행하기 위해서 각 네트워크 단계별로 Cell에 명령어를 입력하여 동적으로 연산의 흐름이 이루어지도록 한다.

그림5에서 Hypercube network는 Photon의 x, y, z에 대한 인덱스에 할당을 하고 신호에 따른 연결경로를 표현하였다.

첫 번째 단계에서 각 Cell에 중심점의 좌표와 Photon충동 좌표를 입력하여 차의 제곱을 구하고 Z축에 대한 네트워크 경로를 통하여 두 결과 값을 더하여 Y Cell과 D Cell에 저장한다.

두 번째 단계에서 4차원 축의 연결을 하여 $(X_1-X_2)^2+(Y_1-Y_2)^2+(Z_1-Z_2)^2$연산이 수행되고 D Cell에서 해당 photon이 근접범위에 포함되어 있는지 비교하게 된다. 그리고 비교연산에 의한 결과 데이터로 다음 단계인 밀도추정 계산을 수행하게 된다. 만약 해당 Photon이 밀도추정 연산에 포함되지 않는다면 해당 값을 0으로 치환한다.

세 번째 단계에서 범위 내에 포함되어 있는 Photon은 x, y, z축에 대해 Photon Power 데이터의 합을 구한다.

Ⅳ. 시뮬레이션

4.1 하드웨어 구현

본 논문은 Verilog-HLD로 설계하였으며 설계된 시스템의 기능 검증을 위하여 Quartus Ⅱ와 Modelsim을 이용하여 시뮬레이션을 수행 하였다.

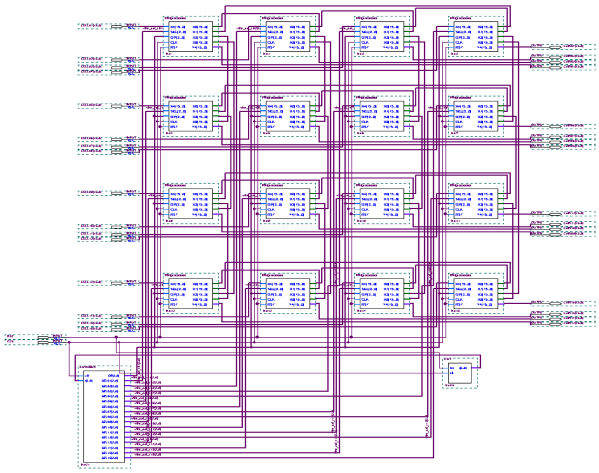


그림 6. 전체 시스템 구조도

그림 6은 Synplify를 이용하여 합성한 구조이다. 중앙에 위치한 16개의 블록은 Photon mapping 연산을 처리하는 Cell이고, 각 Cell에는 레지스터, MUX 그리고 ALU로 구성되며 SIMD제어 신호에 따라서 병렬적인 연산이 이루어진다.

4.2 성능 평가

본 논문에서 제안한 구조의 성능은 표3과 같다. 1 Clock 는 100ns로 동작하고, 1 Iteration동안 4개의 Photon에 대한 연산이 이루어지며 총 900ns가 소요된다. 그리고 시스템 이용률은 각 Cell의 ALU, Register 단위로 계산을 하였다.

표 3. 제안한 구조의 성능

구 분	제안한 구조
1 Iteration	9 clock
gate 수	5592
1 Iteration에 대한 처리속도	900 ns
이용률	62%
CELL 수	16

제안한 구조는 Control Block에서 SIMD제어와 Hypercube Network연결 제어를 담당하기 때문에 복잡도가 증가하였다.

Ⅴ. 결과

본 논문에서는 조명처리 기법에서 Photon mapping 연산의 시간이 가장 많은 Photon map 검색과 Radiance Estimation의 효과적인 처리를 위하여 재구성 Cell을 이용한 SIMD 구조를 설계하였다.

Cell 기반의 프로세서 유닛을 사용함으로서 하나의 구조에 모든 연산을 수행할 수 있으며, 4차원 Hypercube Network를 사용함으로서 Cell간의 데이터 교환의 효율성을 높일 수 있었다.

제안한 구조의 효율성을 높이기 위해서는 Instruction에 따라 클럭할당을 달리하여 계산의 효율성을 높이고, Photonmap 검색단계에서 모든 데이터가 밀도추정 연산에 포함되지 않는 좌표를 효과적으로 찾아내어 연산의 수를 줄이는 방법에 대해서 연구를 수행할 것이다.

참고문헌

[1] Timothy J. Purcell, Craig Donner, Mike Cammarano, Henrik Wann Jensen, and Pat Hanrahan, "Photon Mapping on Programmable Graphics Hardware", Proceedings of the ACM SIGGRAPH / EUROGRAPHICS Conference on Graphics Hardware, pp 41-50, 2003.

[2] Jensen, Henrik Wann, "Realistic Image Synthesis Using Photon Mapping", AK Peters, Ltd. Natick, Massachusetts, pp 75-83, 2001.

[3] Sanchez-Elez, Tabrizi, Bagherzadeh, Anido, Fernandez, "Interactive ray tracing on reconfigurable SIMD MorphoSys", Design Automation and Test in Europe Conference and Exhibition, pp 144 - 149, 2003.

[4] Turner Whitted, "An improved illumination model for shaded display", Communications of the ACM23(6), pp 343-349, 1980.

[5] Christopher A. Burns, "Global Illumination on Parallel Architectures", Senior Thesis, University of Texas Department of Computer Sciences, 2004.

[6] MA V. C. H, MCCOOL M. D, "Low Latency Photon Mapping Using Block Hashing", In Proceedings of the ACM SIGGRAPH / Eurographics Conference on Graphics Hardware, pp 89-99, 2002.