논문 2013-08-11

매니코어 프로세서를 이용한 벡터 기반 래스터화 알고리즘 구현 및 성능평가

(Implementation and Performance Evaluation of Vector based Rasterization Algorithm using a Many-Core Processor)

손 동 구, 김 종 면*

(Dong-Koo Shon, Jong-Myon Kim)

Abstract : In this paper, we implemented and evaluated the performance of a vector-based rasterization algorithm of 3D graphics using a SIMD-based many-core processor that consists of 4,096 processing elements. In addition, we compared the performance and efficiency of the rasterization algorithm using the many-core processor and commercial GPU (Graphics Processing Unit) system which consists of 7 GPUs and each of which have 512 cores. Experimental results showed that the SIMD-based many-core processor outperforms the commercial GPU system in terms of execution time (3.13x speedup), energy efficiency (17.5x better), and area efficiency (13.3x better). These results demonstrate that the SIMD-based many-core processor has potential as an embedded mobile processor.

Keywords : Rasterization algorithm, 3D graphics, Many-core processors, Graphics processing unit (GPU)

1. 서 론

과거에는 휴대용 기기가 통화나 메시지 전송과 같은 제한적인 음성 및 텍스트 기반의 간단한 처리 과정에 국한되었으나, 최근에는 고성능 스마트폰의 보급이 활발해짐으로써 게임이나 비디오 전송 등과 같은 복잡한 그래픽스나 이미지 기반 처리가 가능하게 되었다. 하지만 3차원 그래픽스 처리과정에는

여러 단계의 파이프라인으로 구성되어 있으며, 보다 정교한 그래픽스 표현을 위해서는 연산량이 기하급수적으로 증가하기 때문에 이를 전담하는 고속의 연산 장치가 필요하다 [1]. 더불어 게임 등과 같은 3차원 그래픽스를 처리하는 휴대용 기기는 고성능뿐만 아니라 저전력도 만족시켜야한다.

이러한 휴대용 기기에서 요구되는 고성능 및 저전력을 만족시키는 대안 중의 하나로 SIMD (Single Instruction Multiple Data) 기반 GPU (Graphics Processing Unit)가 유망하다 [2, 3]. 명령어 레벨 (instruction-level)이나 스레드 레벨 (thread-level)로 동작하는 다중 슈퍼스칼라 프로세서들은 실리콘 면적을 멀티포트 레지스터 파일 (multiported register file), 캐쉬 (cache), 파이프라인 (deep pipelined) 기능 유닛 등으로 사용한다. 반면에 SIMD 기반 GPU는 수백, 수천 개의 저비용 프로세싱 엘리먼트 (Processing Element, PE)를 이용하여 고성능뿐만 아니라 저전력도 만족시킨다 [4]. 특히 SIMD 기반 GPU는 지역성 (locality)이나 규칙성 (regularity)이 있는 2차원 패턴 이미지나 3D 그래픽스의 처리에 더욱 효과적인 구조이다.

* Corresponding Author (jmkim07@ulsan.ac.kr)

Received: 19 Oct. 2012, Revised: 21 Nov. 2012, Accepted: 25 Nov. 2012.

D.K. Shon, J.M. Kim: Ulsan University

※ 본 연구는 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구이며(No. 2012-0004962), 또한 중소기업청에서 지원하는 2012년도 산학연공동기술개발사업(No. C0031322)의 연구수행으로 인한 결과물이며, 그리고 지식경제부의 광역경제권 선도산업 육성사업의 일환인 “동남광역경제권 선도산업 지원단”의 2012년 연구비 지원으로 수행되었음

일반적으로 3차원 그래픽스 처리는 T&L (Transformation & Lighting) 과정과 래스터화 (Rasterization) 과정으로 나눌 수 있다 [5]. T&L 과정에서는 모델의 데이터가 소수를 포함한 형태이기 때문에 부동소수점 연산을 해야 하는 반면, 래스터화 과정에서는 출력 화면의 픽셀이 연계되어 연산되므로 정수체계를 이용하여 출력될 좌표를 계산해야 한다.

본 논문에서는 3차원 그래픽스 처리를 효과적으로 수행하기 위해 SIMD기반 매니코어 프로세서를 이용하여 벡터 기반 래스터화 알고리즘을 구현하였고 성능을 분석하였다 [5]. 본 저자의 이전 논문 [5]에서의 성능분석 뿐만 아니라 본 논문에서는 제안하는 SIMD기반 매니코어 프로세서의 잠재가능성을 평가하기 위해 기존의 상용 GPU와의 성능, 에너지효율 및 시스템 면적 효율을 비교 분석하였다.

본 논문의 구성은 다음과 같다. 2장에서는 실험 환경으로써 SIMD기반 매니코어 프로세서 모델 및 벡터 기반 래스터화 알고리즘에 대해 설명하고 또한 매니코어 프로세서를 이용한 벡터 기반 래스터화 알고리즘의 구현에 대해서도 설명한다. 3장에서는 본 논문에서 제안하는 SIMD기반 매니코어 프로세서를 평가하기 위한 실험 방법론 및 성능 평가 지표인 실행시간, 에너지효율 및 면적효율에 대해 설명한다. 또한, 본 장에서는 성능 비교를 위해 선택된 상용 GPU 시스템에 대한 사양을 간략하게 설명한다. 4장에서는 SIMD기반 매니코어 프로세서의 성능 분석 및 상용 GPU와의 성능 및 효율 비교 분석을 보여주며, 끝으로 5장에서 결론을 맺는다.

II. 실험 환경

1. SIMD 기반 매니코어 프로세서 모델

그림 1은 SIMD기반 매니코어 프로세서 아키텍처의 블록 다이어그램을 보여준다[6, 7]. 매니코어 프로세서는 여러 개의 프로세싱 엘리먼트(Processing Element, PE)와 이를 제어하는 어레이 제어 유닛(Array Control Unit :ACU)으로 구성되어 있고, 데이터가 각각의 PE에 균등하게 분배되면 PE들은 메쉬 배열 구조에서 명령어들을 동시에 수행한다. 각 PE는 다음과 같은 특징을 가진다.

- 32비트 폭의 256개 워드로 구성된 로컬 메모리
- 32비트 폭의 16개 3포트 범용 레지스터
- 기본적인 산술/논리 연산을 수행하는 ALU
- 64비트 곱셈 및 누산기(multiply accumulator)

그림 1. SIMD 기반 매니코어 프로세서 아키텍처

Fig. 1 SIMD based many-core processor architecture

- 멀티 비트 산술/논리 시프트 연산을 수행하는 배럴 시프트 (Barrel Shifter)
- 지역 정보를 이용해 각 PE들을 활성화 및 비활성시키는 Sleep 유닛
- 이웃하는 PE들과 데이터 통신을 위한 NEWS (north-east-west-south) 네트워크 및 직렬 I/O 유닛

2. 벡터 기반 래스터화 알고리즘

래스터화에서 많이 쓰이는 스캔 라인(Scan-Line) 알고리즘은 순차연산 기반 알고리즘이다. 따라서 계산과정에서 발생하는 앞 픽셀과 뒤 픽셀에 대한 정보가 필요하며, 이로 인한 픽셀 간 연계성 때문에 매니코어 프로세서에서의 구현은 효율성이 많이 저하되기 때문에 적절한 래스터화 알고리즘의 선택이 불가피하다. 본 논문에서 선택한 벡터 기반 래스터화 알고리즘은 하나의 폴리곤 내부의 어떤 픽셀이라도 동시에 처리가 가능하도록 고안된 병렬 알고리즘으로써 매니코어 프로세서에 적합하다.

래스터화 과정에서 사용되는 폴리곤의 모양은 다양하지만 본 논문에서는 정점 3개로 구성된 하나의 삼각형을 나타내는 폴리곤을 사용하였다. 이는 다양한 형태의 다각형을 원시 폴리곤인 삼각형으로 모두 표현할 수 있기 때문이다. 벡터 기반의 래스터화 과정에서 나오는 결과는 삼각형 형태의 폴리곤 내부의 픽셀 정보이다. 그림 2는 3개의 정점 $V_0(x_0, y_0)$, $V_1(x_1, y_1)$, $V_2(x_2, y_2)$ 로 구성된 하나의 폴리곤을 보여주며, $V_p(x_p, y_p)$ 는 폴리곤 내부의 임의의 한 픽셀이다. 픽셀이 위치한 V_p 정점의 위치는 $\overrightarrow{V_0}$ 와 $\overrightarrow{V_1}$ 의 각각 a 배와 b 배로 축소한 벡터의 합으로 나타낼 수 있다. 벡터 $\overrightarrow{V_0}$ 는 V_0 와 V_1 두 정점을 이용해서 표현되며, $\overrightarrow{V_1}$ 는 V_0 와 V_2 두 정점을 이용

그림 2. 폴리곤 내부의 픽셀 표현

Fig. 2 Expression of the pixel inside a polygon

해서 표현된다[5]. 이 관계를 수식으로 표현하면 다음과 같다.

$$V_p = V_0 + (V_1 - V_0) \times a + (V_2 - V_0) \times b \quad (1)$$

식 (1)에서 V_p 를 구하기 위해 계산된 a 와 b 값은 폴리곤 내부의 픽셀마다 모두 다른 값을 가지게 되는데 이를 이용하여 해당 픽셀의 색상 보간을 수행하게 된다. 위 수식 (1)에서 각 벡터를 x 축과 y 축으로 나누어 정리한 두 식을 연립하면 아래 식(2)와 (3)과 같이 a 와 b 를 계산할 수 있다.

$$a = \frac{(x_2 - x_0)(y_p - y_0) - (y_2 - x_0)(x_p - x_0)}{(x_2 - x_0)(y_1 - y_0) - (y_1 - x_0)(x_2 - x_0)} \quad (2)$$

$$b = \frac{(x_1 - x_0)(y_0 - y_p) - (y_1 - x_0)(x_0 - x_p)}{(x_2 - x_0)(y_1 - y_0) - (y_1 - x_0)(x_2 - x_0)} \quad (3)$$

이렇게 구한 a 와 b 를 이용하여 V_p 의 각 색상에 대한 보간 결과를 구할 수 있으며, 그 수식은 다음과 같다.

$$\begin{aligned} R_p &= R_0 + (R_1 - R_0) \times a + (R_2 - R_0) \times b \\ G_p &= G_0 + (G_1 - G_0) \times a + (G_2 - G_0) \times b \\ B_p &= B_0 + (B_1 - B_0) \times a + (B_2 - B_0) \times b \end{aligned} \quad (4)$$

3. 매니코어를 이용한 벡터 기반 래스터화 알고리즘

표 1은 벡터 기반 래스터화 알고리즘 수행 중에 필요한 데이터를 각 PE의 로컬메모리 영역에 할당한 내용을 보여준다. 각 PE와 로컬메모리에는 각 정점의 좌표와 채널값, 알고리즘의 수행 결과가 저장되는 영역, 그리고 최종결과가 저장되는 영역으로 나뉜다. 표 2는 벡터 기반 래스터화 알고리즘 수행을 위한 매니코어 시스템의 파라미터를 보여준다.

표 1. PE의 로컬메모리 사용 영역
Table 1. Local memory map of each PE

로컬 메모리 영역	용도
MEM(0~4)	V_0 의 좌표, RGB각 채널 값
MEM(5~9)	V_1 의 좌표, RGB각 채널 값
MEM(10~14)	V_2 의 좌표, RGB각 채널 값
MEM(15~31)	Reserved
MEM(32~95)	Color Channel Result(RGB)
MEM(96~99)	Reserved
MEM(100~115)	BITMAP_ADDR
MEM(116~255)	Stack

표 2. 매니코어 시스템 파라미터
Table 2. Many-core system parameters

Parameter	Value
# of PEs	4096
DPE(Data per PE)	64
Memory/PE	256
Clork Frequency	720MHz
Intercon. Network	Mesh
Image Size	512x512

알고리즘을 수행하기에 앞서 각 정점의 좌표와 컬러데이터를 로컬메모리에 저장하며, 각 PE는 64개의 픽셀을 담당한다. 동일한 명령어에 의해 각 PE의 픽셀들은 동시에 처리된다. 하지만 계산이 필요로 하지 않는 폴리곤 외부의 픽셀을 담당하는 PE는 SLEEP 명령어에 의해 비활성화된다. 반면에 SLEEP상태가 아닌 나머지 PE들은 각각의 내부 픽셀의 색상에 대한 보간된 정보를 계산한다. 그림 3과 4는 PE들의 활성 및 비활성상태에 대한 이미지 맵과 상태 판별 의사코드의 예를 보여준다.

III. 실험 방법론 및 성능 평가 지표

1. 실험 방법론

그림 5는 세 가지 레벨(애플리케이션, 아키텍처, 테크놀로지)로 구성되어 있는 SIMD 기반 매니코어 프로세서의 실험 방법론을 보여준다. 애플리케이션 레벨에서는 SIMD 기반 매니코어 프로세서용 정밀 사이클(cycle-accurate) 시뮬레이터를 이용하여 사이클 개수, 동적 명령어 빈도, 시스템 이용률(syst-

그림 3. 각 PE의 활성 및 비활성 상태의 이미지 맵
Fig. 3 Image map of active and inactive status for each PE

그림 4. 각 PE의 상태 판별 의사코드
Fig. 4 Pseudocode for activation and inactivation of each PE

em utilization) 등의 실행 데이터를 추출하였다. 아키텍처 레벨에서는 모델링된 아키텍처의 디자인 변수들을 계산하기 위해 Chai가 제안한 SIMD 기반 병렬 프로세서용 아키텍처 모델링 툴을 사용하였다 [8]. 테크놀로지 레벨에서는 각 아키텍처 모델들의 테크놀로지 변수(latency, power, clock frequency)를 계산하기 위해 Generic System Simulator (GENESYS)를 사용하였다 [9]. 마지막으로 세 레벨에서 구현된 데이터베이스를 조합하여 각 경우에 대한 실행시간, 처리량, 에너지 효율을 결정하였다.

2. 성능 분석 지표

표 3은 매니코어 프로세서의 성능 평가를 위한 세 가지 지표(실행시간, 에너지 효율, 면적 효율)를 보여준다. 실행 시간 (execution time)은 래스터화 알고리즘이 수행된 시간을, 에너지 효율 (energy efficiency)은 단위 에너지당 소비된 명령어 개수 (Giga-operations/Joule)를 나타내고, 시스템 면적 효율 (area efficiency)은 단위 시스템 면적당 소비

그림 5. 매니코어 프로세서를 위한 실험 방법론
Fig. 5 Simulation methodology for a many-core processor

표 3. 성능평가 지표 요약

Table 3. Summary of performance evaluation metrics

실행 시간	$t_{exec} = C/f_{clk}$
에너지 효율	$\eta_E = \frac{1}{t_{exec} \times Energy} \left[\frac{1}{s \times Joules} \right]$
면적 효율	$\eta_A = \frac{1}{t_{exec} \times Area} \left[\frac{1}{s \times mm^2} \right]$
C : 사이클 개수, f_{clk} : 클럭 주파수,	

표 4. 성능비교를 위해 선택된 GPU 시스템 사양

Table 4. GPU system specification for the purpose of performance comparison

CPU	Intel Xeon x5690 3.46Ghz * 2ea
Memory	144GB RAM
GPU	Nvidia Geforce GTX580 * 7ea
Clork Frequency	1510MHz
OS	CentOS 5.8
CUDA cores per GPU	512

표 5. 실험에 사용된 정점 데이터

Table 5. Vertex data used in the experiment

정점	X좌표	Y좌표	R	G	B
V ₁	10	10	0	200	100
V ₂	400	100	255	255	150
V ₃	50	400	200	0	100

그림 6. SIMD기반 매니코어 프로세서를 이용한 결과 폴리곤 이미지

Fig. 6 Resulting polygon image using SIMD based many-core processor

된 명령어 개수를 나타낸다.

표 4는 매니코어 프로세서와의 성능 비교를 위해 선택된 상용 고성능 GPU의 시스템 사양을 보여준다. GPU시스템은 2개의 CPU와 7개의 GTX580 GPU로 구성되어 있으며, 각 GPU는 512개의 가지고 있다.

IV. 결과 및 성능 분석

표 5는 실험에 사용된 정점데이터와 각 정점의 컬러정보를 보여준다.

그림 6은 정점 3개를 입력으로 하는 래스터화 알고리즘을 SIMD기반의 매니코어 프로세서를 이용하여 구현한 결과 폴리곤 이미지를 보여준다.

1. 실행 시간

본 실험에서는 제안하는 SIMD기반 매니코어 프로세서와 상용 GPU시스템의 정확한 성능 비교보다는 매니코어 프로세서의 잠재가능성을 보이기 위함이다. 그림 7은 래스터화 알고리즘을 SIMD기반 매니코어 프로세서와 상용 GPU시스템을 사용하여 구현할 때 소비된 실행시간을 보여준다. 모의실험결과, 매니코어 프로세서는 상용 GPU시스템보다 3.13배 speedup을 보였다.

2. 에너지 효율

에너지 효율은 단위 에너지당 수행할 수 있는 작업량이다. 그림 8은 SIMD기반 매니코어 프로세서와 상용 GPU시스템의 에너지 효율을 보여준다. 세로축의 수치는 상대적인 값이기 때문에 수치 자체보다는 그 값의 차이를 보는 것이 중요하다. 매니코어 프로세서는 상용 GPU시스템보다 약 17.5배

그림 7. 래스터화 알고리즘의 실행 시간 비교

Fig. 7 Execution time comparison of the rasterization algorithm

그림 8. 에너지 효율 비교

Fig. 8 Energy efficiency comparison

그림 9. 면적 효율 비교

Fig. 9 Area efficiency comparison

향상을 보였다. 에너지 효율의 증가는 시스템 배터리 수명을 증가시키는 결과를 가져온다.

3. 면적 효율

면적 효율은 시스템의 단위면적당 수행할 수 있는 작업량을 나타낸다. 그림 9는 매니코어 프로세서와 상용 GPU시스템의 면적효율을 보여준다. 에너지 효율 결과와 마찬가지로 면적효율에서도 매니코어 프로세서가 GPU시스템보다 13.3배 높은 효율을 보였다. 면적 효율의 증가는 주어진 시스템에서 구성요소의 효율을 증가시키는 결과를 가져온다.

V. 결 론

본 논문에서는 벡터 기반 래스터화 알고리즘의 고성능, 저전력 처리를 위한 SIMD기반 매니코어 프로세서를 제안하고 성능 평가를 수행하였다. 또한, 상용 GPU시스템과의 실행시간, 에너지효율 및 면적효율을 비교하였다. 매니코어 프로세서는 7개의 GPU로 구성되어 있는 상용 시스템보다 실행시간에서 3.13배 speedup, 에너지효율에서 약 18배 향상, 면적효율에서 13.3배의 향상을 보였다. 이러한 결과는 제안하는 SIMD기반 매니코어 프로세서가 모바일 프로세서 시장에서 잠재가능성을 보여준다.

참고문헌

- [1] H.J. Woo, "A design of the embedded 3D graphics rendering processor for portable devices", Thesis for master course, Yonsei University, 2004 (in Korean).
- [2] N. Singhal, J.W. Yoo, H.Y. Choi, I.K. Park, "Implementation and Optimization of Image Processing Algorithms on Embedded GPU," IEICE Trans. Inf. & Syst., Vol. E95-D, No. 5, pp.1475-1484, 2012.
- [3] I.K. Park, N. Singhal, M.H. Lee, S. Cho, C.W. Kim, "Design and Performance Evaluation of Image Processing Algorithm on GPUs," IEEE Trans. Parallel Distrib. Syst., Vol. 22, No. 1, pp.91-104, 2011.
- [4] Y.H. Ahn, Y.S. Hwang, K.S. Chung, "Kernel Level Power Management Solution for Multi-Core", Journal of IEMEK, Vol. 4, No. 2, pp.50-54, 2009 (in Korean).
- [5] D.K. Shon, J.M. Kim, "Rasterization Algorithm based on Vector for Many Core Processor", Proceedings on Summer Conference of KISPS, pp.66-68, 2012 (in Korean).
- [6] B.K. Choi, C.H. Kim, J.M. Kim, "Implementation of SIMD-based Many-Core Processor for Efficient Image Data Processing", Journal of the KSCE, Vol. 16, No. 1, pp.1-9, 2011 (in Korean).
- [7] Y.M. Kim, J.M. Kim, "Design and Verification of High-Performance Parallel Processor Hardware for JPEG Encoder", Journal of IEMEK, Vol. 6, No. 2, pp.100-107, 2011 (in Korean).
- [8] S.M. Chai, T.M. Taha, D.S. Wills, J.D. Meindl, "Heterogeneous architecture models for interconnect-motivated system design," IEEE Trans. VLSI Systems, special issue on system level interconnect prediction, Vol. 8, No. 6, pp.660-670, 2000.
- [9] J.C. Eble, V.K. De, D.S. Wills, J.D. Meindl, "A generic system simulator (GENESYS) for ASIC technology and architecture beyond 2001," Proceedings on International Conference of the Ninth Ann. IEEE ASIC, pp.193-196, 1996.

저 자 소 개

손 동 구

2012년 현재, 울산대 전기공학부 석사과정.

관심분야: 병렬프로세서 구조, GPGPU 프로그래밍, 임베디드시스템.

Email: dongkoo88@gmail.com

김 종 면

1995년 명지대 전기공학과 학사.

2000년 University of Florida 전기컴퓨터공학과 석사.

2005년 Georgia Tech. 전기컴퓨터공학과 박사.

현재, 울산대 전기공학부 교수.

관심분야: 임베디드 SoC, 컴퓨터구조, 병렬처리.

Email: jmkim07@ulsan.ac.kr