논문번호 12-04-04
22

Journal of IKEEE.Vol.16,No.4,304~309,December 2012
http://dx.doi.org/10.7471/ikeee.2012.16.4.304

다중코어 GPU를 위한 병렬처리 보간 알고리즘 구현

Implementation of Parallel Processing Interpolation Algorithm for Multicore GPU

이 광 엽*, 김 치 용**

Kwang-Yeob Lee*,Chi-Yong Kim**

Abstract

As resolution for displays is recently more and more increasing, the amount of data abd calculation that graphic hardware needs to process are also increasing. Especially the amount of data processing by rasterizer is rapidly increasing. This paper used an algorism using coordinates in center of gravity and area for triangle instead of using bilinear algorism[1] used by conventional interpolation, which is to make it easier for parallel processing by rasterizer. This paper implemented designed rasterizer under FPGA environment ,and compared it with conventional rasterizer and verified it. This rasterizer is proved to have approximately 50% higher performance compared to conventional one.

요 약

최근 디스플레이의 해상도가 높아짐에 따라 그래픽 하드웨어가 처리해야할 데이터량과 연산량이 증가 하고 있다. 특히 레스터라이저의 데이터 처리량이 크게 증가 하고 있다. 본 논문은 높은 해상도의 많은 데이터를 빠르게 처리하기 위하여 레스터라이저를 병렬로 설계 하였다. 본 논문은 레스터라이저의 병렬화를 용이하게 하기 위하여 기존 보간 단계에서 사용하는 Bilinear 알고리즘[1] 대신 삼각형의 무게중심 좌표와 넓이를 이용하는 알고리즘을 사용하였다. 설계한 레스터라이저를 FPGA 환경에서 구현하여 기존 레스터라이저와 비교 검증 하였다. 기존 레스터라이저와 비교 결과 성능이 약 50퍼센트 상승 하였다.

Key words : rasterizer, polygon, interpolation, scanline

Ⅰ. 서론

3D 그래픽스는 많은 데이터 연산을 요구한다. 3D 그래픽스 처리 초기에는 CPU가 모든 연산 처리를 담당하였으나, 더욱 정교하고 복잡도 높은 3D 그래픽스의 표현을 위해 데이터 처리량이 증가하여, CPU만으로 처리하기에는 큰 부담이 되었다. 3D 그래픽스를 위한 별도의 가속기를 설계하거나, 범용 프로세서에 확장하여 설계하는 방법을 사용한다. 최근 급격하게 발전하고 있는 모바일 임베디드 디바이스 시장에서도 마찬가지 현상을 보여주고 있다.

3D 그래픽스 하드웨어 처리 과정은 크게 지오메트리 단계와 래스터라이제이션 단계로 구분한다. 모델 정점의 좌표 변환, 이동, 크기 변환을 통해 3D모델의 좌표를 계산하는 것이 지오메트리 과정이고, 변환된 3D모델의 각 폴리곤 단위로 내부 픽셀 색상을 보

*Professor, Dept. of Computer Engineering, Seokyeong University
kylee@skuniv.ac.kr
**Professor, Dept. of Computer Science, Seokyeong University, Corresponding author
This work was sponsored by Industrial Strategic Technology Development Program funded by the Ministry of Knowledge Economy (10039188, SoC platform development for smart vehicle info-tainment system).
Manuscript received Aug. 27. 2012; revised Oct. 24. 2012; accepted Oct. 29, 2012

간하는 것이 래스터라이제이션이다.[2] 래스터라이제이션 단계는 픽셀 단위로 접근하기 때문에 3D 그래픽스의 파이프라인에서 가장 많은 시간을 소요하는 단계이다.

최근 모바일 디스플레이의 해상도가 높아지고, 높은 성능의 3D 그래픽스 구현능력이 중요해짐에 따라 폴리곤 내부의 모든 픽셀을 연산해야하는 레스터라이제이션 단계의 연산량이 증가 하고 있다.

이렇게 큰 폭으로 증가한 연산량을 빠르게 처리하기 위해서는 래스터라이제이션을 담당하는 하드웨어어의 병렬화 설계가 해결 방안이 될 수 있다.

현재 래스터라이제이션에서 이중 선형 보간 알고리즘을 이용한 스캔라인 방식이 주로 사용되고 있지만, 이 알고리즘은 폴리곤 내부 픽셀을 연산하기 전에 폴리곤이 가지는 3개의 에지 픽셀의 값에 대한 선행 연산이 필수적이므로 부분적인 병렬화 이상의 효율을 가지기 어렵다.

본 논문 에서는 선행연산이 필요 없는 보간 알고리즘을 이용하여 레스터라이제이션 파이프라인 전체를 병렬 처리할 수 있는 레스터라이저를 설계 및 검증 하였다.

II. 보간알고리즘

보간 알고리즘은 폴리곤에 색을 칠할 때에 정점의 색을 보간 하여 내부의 픽셀의 색상 값을 결정하는 알고리즘 이다. 보간을 하는 방법에는 무게중심 좌표를 이용하는 방법과 이중 선형 알고리즘을 이용하는 방법이 있다.

1. 이중선형 보간 알고리즘

그림 1의 삼각형 내부의 픽셀 V의 색을 이중 선형 보간 알고리즘을 이용하여 구하면 다음과 같다. 픽셀 G와 픽셀 Y를 보간 하면 선분 Y 상의 모든 픽셀의 색이 결정된다. 선분 GS의 길이와 선분 $\overline{YS}$의 길이의 비율이 가중치가 된다. 또한 두 선분의 길이의 비율은 그림에서 y방향 길이의 비율인 a1 : a2 와 동일하다. 즉 $\overline{GS}$: $\overline{YS}$ = a1 : a2 이다. 픽셀 S를 G와 Y의 함수로 표현 하면 식 (1)과 같다. 같은 방법으로 픽셀 T를 R과 G를 함수로 표현하면 식 (2)와 같다. 픽셀 S와 T의 값이 결정되면 주사선을 따라 가면서 내부 픽셀의 색상 값을 결정한다. V의 픽셀 값은 S와 T의 y값이 같으므로 식 (3)과 같이 구할 수 있다. 픽셀 S와 픽셀 T의 값을 구하는 데에는 y 방향의 선형 보간을 사용한다. 그로부터 픽셀 V의 색상 값을 구하는데 x방향의 선형 보간을 사용한다. y, x방향의 선형 보간을 모두 사용 했으므로 이 방법을 이중 선형 보간이라 한다.

$$S \quad \frac{a1}{a1+a2}Y + \frac{a1}{a1+a2}G \qquad (1)$$

$$T = \frac{b1}{b1+b2}R + \frac{b1}{b1+b2}G \qquad (2)$$

$$V = \frac{c1}{c1+c2}T + \frac{c1}{c1+c2}S \qquad (3)$$

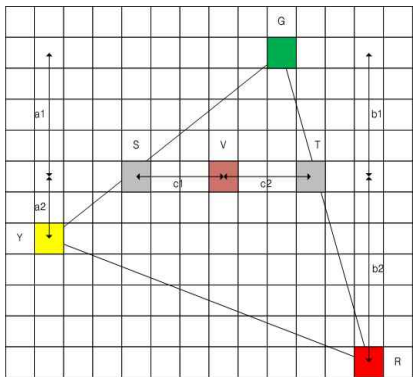

Fig 1. Bilinear Interpolation
그림 1. 이중선형보간법

이러한 이중 선형 보간 알고리즘은 연산식이 간단하다는 장점이 있으나 그림 1에서 확인할 수 있듯이 픽셀 V의 색상 값을 연산하기 위해서는 픽셀 S와 픽셀 T의 색상 값을 선행 연산 하여야 한다. 이러한 픽셀 간의 데이터 의존성이 래스터라이저 병렬화의 효율을 절감 시킨다.

2. 무게중심 좌표를 이용한 보간 알고리즘

무게중심 좌표를 이용한 보간 알고리즘을 이용하면 그림 1의 T와 S를 구하는 선행연산 없이 각 픽셀의 색상 값을 구할 수 있다. 그림 1에서처럼 이중 선형 보간 알고리즘을 사용하면 T와 S를 구하는 동안 이전과 이후 스테이지 연산기는 연산이 끝날 때까지 대기해야 한다. 하지만 무게중심 좌표를 이용한 보간 알고리즘을 이용하면 데이터 의존성이 없어져 래스터라이저의 병렬화 효율이 증가 한다.

무게 중심좌표와 삼각형의 넓이를 이용하여 픽셀 값을 보간 하는 방식은 다음과 같다.

그림 2에서 V의 a값은 선분 R과 평행을 이루는 선분의 기울기이다. 기울기 a인 선분과 선분 $\overline{GR}$사이의 거리는 선분$\overline{GR}$을 밑변으로 하는 삼각형의 높이를 의미한다. 삼각형 VGR과 삼각형 YRG는 밑변 $\overline{GR}$이 같고 높이가 다른 삼각형이다. 따라서 이 높이는 삼각형 넓이의 비율로 구할 수 있다. 즉 다음과 같은 식으로 표현가능하다.

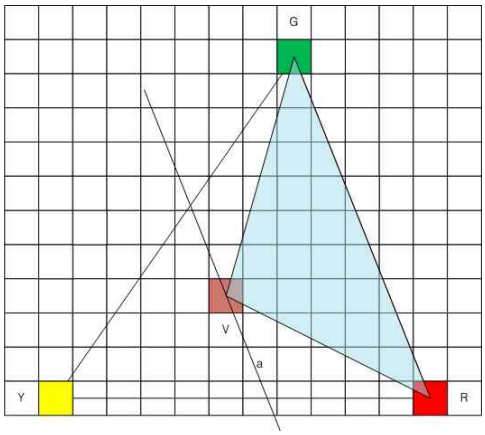

Fig 2. Center of gravity Interpolation
그림 2. 무게중심 보간법

$$V \quad area(VGR)/area(YGR) \tag{4}$$
$$V_\beta = area(VGY)/area(YGR) \tag{5}$$
$$V_\gamma = area(VYR)/area(YGR) \tag{6}$$

여기서 $V_\alpha, V_\beta, V_\gamma$는 삼각형 YGR에 대한 각 삼각형 VGR, VGY, VYR의 비율이다. 위와 같이 정점 Y,G,R의 좌표와 V의 좌표 값을 알고 있으면 정점 V에 대한 가중치 값인 $V_\alpha, V_\beta, V_\gamma$ 값을 구할 수 있다.

$$V_r = V_\alpha + V_{\beta_r} + V_{\gamma_r} \tag{7}$$
$$V_g = V_{\alpha_g} + V_{\beta_g} + V_{\gamma_g} \tag{8}$$
$$V_b = V_{\alpha_b} + V_{\beta_b} + V_{\gamma_b} \tag{9}$$

위 식과 같이 각 삼각형의 정점 V에 대한 r값의 비율인 $V_{\alpha_r} + V_{\beta_r} + V_{\gamma_r}$값을 모두 더하면 정점 V의 r 값이 된다. 이와 같이 $V_\alpha, V_\beta, V_\gamma$ 값을 이용하여 삼각형 내부 픽셀 V의 좌표에 해당하는 r, g, b 값을 구할 수 있다.

III. Rasterizer Architecture

본 논문에서 설계한 래스터라이저의 파이프라인은 그림 3과 같다.

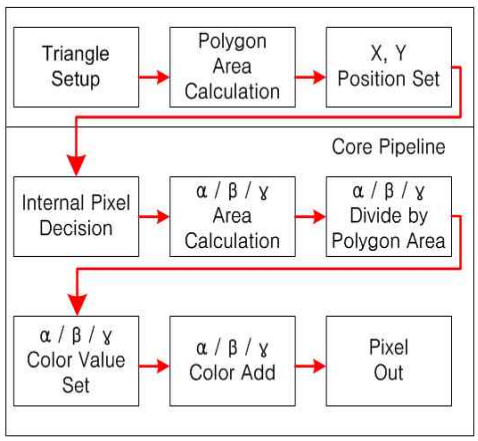

Fig 3. Pipeline for the designed core
그림 3. 설계한 코어의 파이프 라인

상위 모듈에서 정점 좌표 와 칼라 값을 받아 전체 삼각형의 넓이를 구하고 연산해야할 X, Y 좌표를 코어에 전송 한다. 코어 내부의 모듈들은 파이프라인으로 설계하여 데이터 처리 효율을 높였다. 파이프 라인의 Internal Pixel Decision 스테이지는 X,Y 좌표를 받아 이 좌표가 현재 삼각형 내부에 있는지 외부에 있는지를 판단한다.

Area Calculation 스테이지는 그림 2에서와 같이 삼각형을 3개로 분할하여 각 α, β, γ 삼각형의 넓이를 구한다. Divide Polygon Area 스테이지 에서는 각 α, β, γ 삼각형의 넓이를 전체 삼각형의 넓이로 나눈다. Color Value Set Stage는 현재 X,Y 좌표에 대한 α, β, γ 가중치 값을 구하여 각 삼각형의 R,G,B 값을 구한다. Color Add 스테이지 에서는 각 R,G,B 값을 더해 최종 픽셀의 색상 값을 출력 하게 된다.

본 논문에서는 레스터라이저를 그림 4와 같이 다중코어 형태로 설계 하여 처리 속도를 향상 시켰다. Coordinates Manager모듈에서 각 코어에 처리할 X,Y 좌표를 한 클럭마다 배분 시켜 준다. 각 코어는 파이프라인을 거쳐 해당하는 X,Y 좌표의 색상 값을 출력한다.

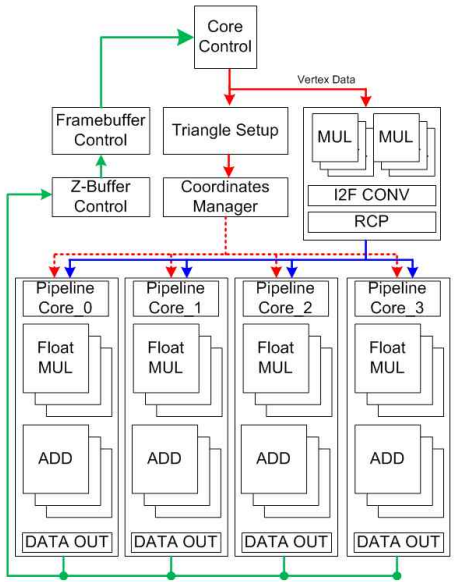

Fig 4. Structure for multi-core rasterizer
그림 4. 다중코어 레스터라이저 구조

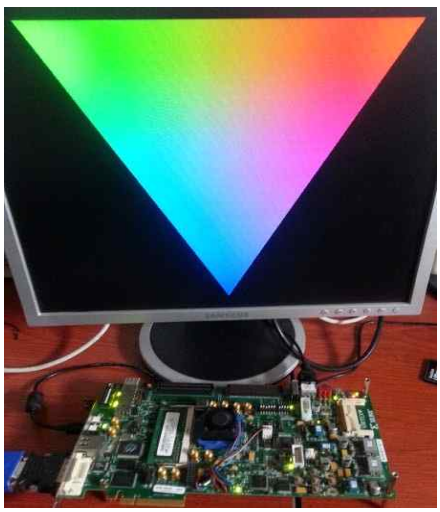

Fig 5. Multi core rasterizer implemented in FPGA
그림 5. FPGA에 구현한 다중코어 레스터라이저

설계한 다중코어 레스터라이저를 검증하기 위해서 Xilinx사의 FPGA 보드인 Vertex6 ML605 보드를 사용 하였다. 동작 주파수는 100 MHz로 구성하였다.

코어 증가에 따른 처리량을 알아보기 위해 320x240 크기의 Frame Buffer에 삼각형을 그림5와 같이 설정하고 이를 처리하는 소요 Cycle을 시뮬레이션 하였다. 코어가 증가함에 따라 총 데이터를 처리하는 Cycle이 감소하는 것을 그림 6과 같이 확인할 수 있었다.

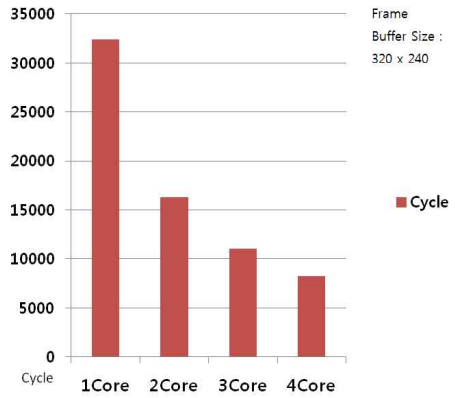

Fig 6. Cycle depending on increase of core
그림 6. 코어 증가에 따른 소요 Cycle

해상도가 증가함에 따라 총 데이터를 처리하는 Cycle이 어떻게 변하는지 알아보기 위하여 단계별로 Frame Buffer 크기를 증가 시켜 총 데이터를 처리하는 Cycle을 그림 7과 같이 측정 하였다.

그림 7과 같이 해상도가 높아짐에 따라 처리해야 할 데이터 량이 급격히 높아지는 것과 해상도에 따른 데이터를 처리하는데 소요되는 사이클을 확인 하였다.

그림 8은 동작주파수가 35 MHz인 기존 레스터라이저와 동일한 동작 주파수인 35 MHz에서 초당 픽셀 처리량을 비교한 것이다. 동작주파수 35 MHz에서 초당 154.7Mpixels을 처리한다.

그림 9는 동작 주파수가 10 MHz인 기존 레스터라이저와 동일한 동작 주파수 10 MHz에서 초당 픽셀 처리량을 비교한 것이다. 동작주파수 10MHz에서 52.6Mpixels을 처리한다.

설계한 다중코어 레스터라이저는 최대 100 MHz로 동작 할 수 있으며 이때 최대 초당 525 Mpixels을 처리 할 수 있다.

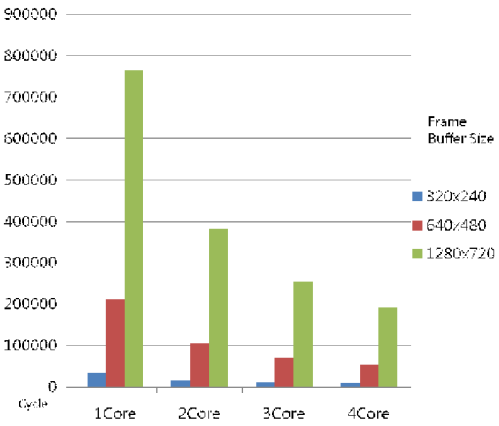

Fig 7. Required cycle depending on increase of resolution
그림 7. 해상도 증가에 따른 소요 Cycle

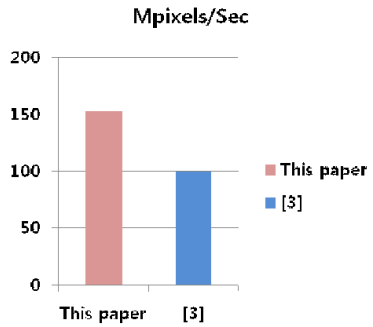

Fig 8. Comparison for pixel processing per second
그림 8. 초당 픽셀 처리량 비교

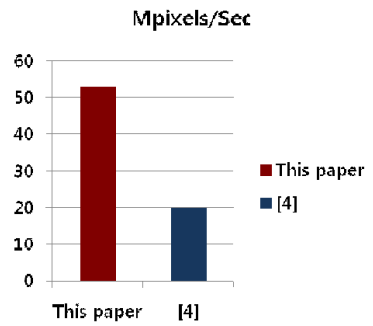

Fig 9. Comparison for pixel processing per second
그림 9. 초당 픽셀 처리량 비교

그림 10은 기존 레스터라이저와 로직 사이즈를 비교한 것이다. 로직 사이즈는 병렬화로 인한 리소스 증가로 기존 레스터라이저에 비하여 증가 하는 것을 확인 하였다.

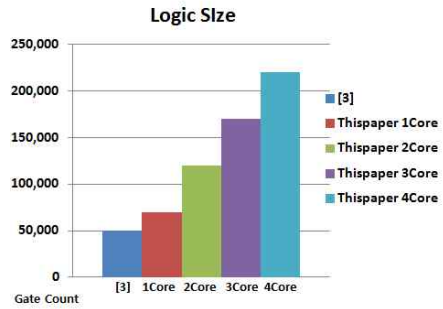

Fig 10. Comparison for Logic size
그림 10. 로직 사이즈 비교

Ⅳ. 결론

설계한 다중코어 레스터라이저는 기존 레스터라이저에 비해 병렬화가 쉽고 병렬화로 인한 성능 향상이 크다는 것을 확인 하였다. 추후 연산기 사용량을 좀 더 줄이고 보간 이후 부분인 텍스쳐 처리부까지 설계하여 병렬화로 인한 성능 향상을 비교해 볼 예정이다.

References

[1] Gribbon, K.T. Bailey, D.G. ,A novel approach to real-time bilinear interpolation, Electronic Design, Test and Applications, 2004. DELTA 2004. Second IEEE International Workshop on. ,2004 , Page(s): 126 - 131.

[2] Dool-Bong Jeon, Kwang-Youb Lee ,A design of a 3D graphics rasterizer with a culling and clipping, TENCON 2007 - 2007 IEEE Region 10 Conference, 2007 , Page(s): 1 - 4 .

[3] Kyungsu Kim; Hoosung-Lee; Seonghyun Cho; Seongmo Park, Implementation of 3D graphics accelerator using full pipeline scheme on FPGA , SoC Design Conference, 2008. ISOCC '08. International , 2008, Page(s): II-97 - II-100.

[4] Jeong-Ho Woo, Ju-Ho Sohn Byeong-Gyu Nam, Hoi-Jun Yoo, Mobile 3D Graphics SoC from Algorithm to Chip, Wiley, 2010, pp 150.

BIOGRAPHY

Lee Kwang Yeob (Life member)

1985. 8 Seogang University, Dept. of Electronics Engineering(BS)
1987. 8 Yonsei University, Dept. of Electronics Engineering(MS)
1994. 2 Yonsei University, Dept. of Electronics Engineering(Ph.D)
1989~1995. 2 Hyundai Electronics Inc., Senior Researcher
1995.3~ Seokyeong Univeristy, Dept. of Computer Engineering, Professor
<Research interests> Microprocessor, Embedded System, 3D Graphics System

Kim Chi Yong (Member)

1981. 2. Kyungpook National University, Dept. of Statics(BS)
1986. 2. Seoul National University, Dept. of Computing Science & Statistics(MS)
1993. 2. Seoul National University, Dept. of Computing Science & Statistics(Ph. D)
1995.3~ Seokyeong Univeristy, Dept. of Computer Science, Professor
<Research interests> Stochastic process, Mathematical analysis, Computer arithematic