

고성능 3차원 그래픽 가속기를 위한 타일 트레이버설 방식의 파이프라인된 스캔 컨버전 유닛 설계

전원호^o 최문희 박우찬 한탁돈 김신덕
연세대학교 컴퓨터과학과
{whchun, mhchoi, chan, hantack, sdkim}@kurene.yonsei.ac.kr

Design of the Pipelined Scan Conversion Unit based on Tile Traversal Method for High Performance 3D Graphics Accelerator

Won-Ho Chun^o Moon-Hee Choi Woo-Chan Park Tack-Don Han Shin-Dug Kim
Dept. of Computer Science, Yonsei University

요 약

3차원 영상을 처리하는데 있어 래스터라이제이션은 프레임 버퍼에 저장될 픽셀을 구하는 과정이다. 여러 개의 픽셀로 구성되는 폴리곤을 렌더링하기 위해서 스캔라인 방식 또는 반 평면 함수를 이용한 타일 트레이버설 방식 등이 사용되고 있다. 본 논문에서 기반으로 하고 있는 타일 트레이버설 방식은 스캔라인 방식에 비해 메모리 효율 및 텍스처 캐쉬의 지역성에서 이점을 가지고 있으나 복잡한 탐색 과정 때문에 파이프라인 구조로 구현하기는 어렵다. 본 논문에서 제안하는 구조는 분기 예측 기법을 적용하여 트레이버설 과정에서의 분기로 인해 발생하는 파이프라인 지연을 기존의 트레이버설 구조에 비해 약 30% 정도 줄임으로써 고성능 3차원 그래픽 가속기에 적합한 스캔 컨버전 유닛을 제안하였다.

1. 서론

최근 수 년 동안 3차원 그래픽 가속기는 더욱 실감나고 부드러운 영상을 실시간으로 처리할 수 있도록 고성능화되고 있다. 이를 위해 수많은 물체로 구성된 장면들을 이루고 있는 엄청난 양의 3차원 데이터를 그래픽 파이프라인 안에서 빠르게 처리할 수 있도록 다양한 알고리즘을 기반으로 한 구조들이 제시되고 있다[1].

3차원 그래픽 파이프라인 단계 중 래스터라이제이션 (rasterization) 과정은 폴리곤 (polygon)을 이루고 있는 각각의 픽셀 (pixel) 값을 결정하는 과정이다. 이 단계에서는 스캔 컨버전 (scan conversion) 및 블렌딩 (blending) 과 Z-버퍼 처리를 하게 된다. 래스터라이제이션은 위의 단계들을 거치면서 생성될 프래그먼트 (fragment)를 구하기 위해 폴리곤을 트레이버설 하게 되는데, 스캔라인 방식 또는 반 평면 함수를 이용한 타일 트레이버설 방식을 기반으로 하고 있다[1][2][3][6].

타일 트레이버설 (tile traversal) 알고리즘은 트레이버설 하는데 있어 타일링 방식을 적용시킨 것으로서, 스캔라인 (scan-line) 알고리즘에 비해 메모리 페이지 크로싱 (page crossing) 발생 및 텍스처 캐쉬 (texture cache)의 적중 실패율 (miss ratio)을 낮추어 성능을 향상 시키는 장점을 가지고 있다. 하지만 상대적으로 분기 (jump) 하는 부분이 많아 파이프라인으로 구성하기가 어려워 Neon과 같은 대표적인 구조에서도 그 성능을 최대한 발휘하지 못하고 있다[5][6].

이에 본 논문에서는 스캔 컨버전 구조를 설계하는데 있어 메모리 효율을 위해 타일 트레이버설 알고리즘을 기반으로 하고

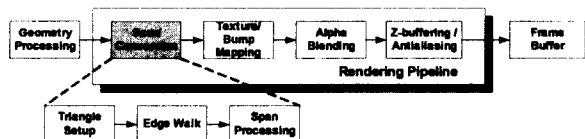
고성능을 위해 분기예측 (branch prediction) 기법을 적용하여 파이프라인 지연 (stall)을 최소화 함으로써 파이프라인 방식의 타일 트레이버설 스캔 컨버전 유닛을 설계하고 있다.

본 논문의 구성은 다음과 같다. 2장에서는 기본적인 래스터라이제이션 과정과 타일 트레이버설 알고리즘에 대하여 살펴보고 3장에서는 파이프라인 방식의 비교 모델인 스캔라인 구조와 제안될 타일 트레이버설 구조에 대해 설명하도록 한다. 그리고 4장에서는 스캔라인 방식을 기준으로 트레이버설 방식의 분기 비율에 대해 실험 하였고, 5장에서 결론 및 향후 연구 계획에 대하여 논하도록 하겠다.

2. 렌더링 파이프라인 및 타일 트레이버설 알고리즘

2.1 그래픽 렌더링 파이프라인

[그림1]에서 보는 바와 같이 3차원 데이터는 크게 기하학 처리와 렌더링 과정을 거쳐 프레임 버퍼 (frame buffer)로 보내진다. 렌더링 부분은 스캔 컨버전 부분과 맵핑 (mapping)관련 부분, 블렌딩 (blending), Z-buffer 처리부분으로 구성되고, 스캔 컨버전 부분은 다시 삼각형 셋업 (triangle setup)과 변 생성 (edge walk), 스캔 처리 (span processing) 단계로 구성 되어진다[1][2][3].



[그림 1] 그래픽 렌더링 파이프라인

이 논문은 과학기술부에서 지원하는 국가지정연구실 사업과제 (과제번호 2000-N-NL-01-C-133) 연구비에 의해 수행됨.

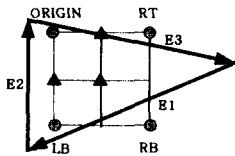
폴리곤 트래버설은 스캔 컨버전 부분에서 이루어지는 작업으로 변 생성과 스캔 처리과정에서 스캔라인 방식 또는 타일 트래버설 방식을 적용시킬 수 있다. 스캔라인 방식에 비해 트래버설 방식은 탐색 과정이 복잡하기 때문에 이 과정을 효과적으로 처리해 주기 위한 구조가 요구된다.

2.2 트래버설 알고리즘

폴리곤을 트래버설하기 위해 반 평면 변 함수 (half-plane edge function)를 이용할 수 있다. 반 평면 변 함수는 폴리곤의 내부와 외부를 판별할 수 있는 함수로서 그 값이 양이면 변의 오른쪽, 음이면 변의 왼쪽, 0이면 변 위에 점이 존재하게 되고, 그 식은 다음과 같다[4]:

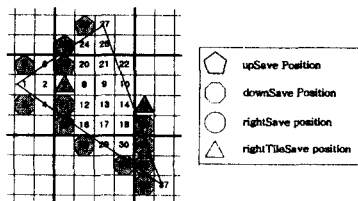
$$E(X,Y) = (X - X_0)A_y - (Y - Y_0)A_x.$$

폴리곤을 트래버설하는 기본 단위로는 스텝틀을 이용하는데, [그림2]에서와 같이 스텝틀상의 여러 개의 탐색 점들은 폴리곤과의 교차 여부를 판별하며, 원으로 표시된 네 꼭지점은 추가적으로 스텝틀의 이동에 관한 역할도 담당하게 된다.



[그림 2] 삼각형을 교차하는 2x2 스텝틀의 예

스텝틀을 이동하기 위해서는 위, 아래, 오른쪽 정보에 대한 context가 필요하다. 하나의 context에는 변 함수 값과 x, y, z, r, g, b, a와 같은 픽셀 (pixel)을 칠하는데 필요한 모든 정보가 저장된다. 스텝틀은 이동할 때마다 위, 아래, 오른쪽을 조사하여 유효한 위치를 context에 저장하게 되고 저장된 context의 내용을 불러들여 스텝틀 이동이 이루어지게 된다. 폴리곤의 트래버설은 더 이상 가져올 내용이 없으면 끝마치게 된다. [그림3]은 타일 트래버설 방식의 트래버설 순서를 보여주고 있다. 굵은 선은 타일 경계를 표시하고, 각각의 작은 사각형은 스텝틀을 가리킨다. 이 알고리즘에서 타일라인은 수직적으로 움직이고 스텝틀라인은 수평적으로 움직인다[5][6].



[그림 3] 타일 트래버설 방법

타일 트래버설 방식을 이용하면 스텝틀을 이용해서 쉽게 여러 개의 프래그먼트들을 병렬적으로 생성할 수 있고, 메모리 접근에 있어 페이지 크로싱이 일어나는 빈도를 줄이게 되어 메모리 효율에 좋으며, 텍스처 캐쉬의 사용에 있어 적중 실패율을 낮추어 성능 향상을 꾀할 수 있다. 그러나 트래버설하는 과정에서 스캔라인 방식처럼 연속적으로 진행하지 않고 중간에 분기하는 경우가 많이 발생함으로 파이프라인으로 구성하기가 매우 어렵다. [그림3]에서 음영처리 되어있는 부분이 분

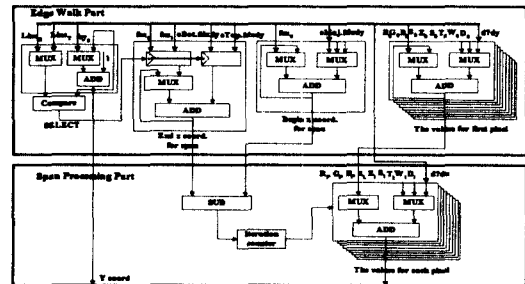
기가 발생하는 부분이다. 이러한 부분은 타일의 크기가 고정되기 때문에 다음에 어디로 갈 것인지를 예측할 수 있어 분기하는 부분을 미리 불러들일 수 있다.

3. 타일 트래버설 하드웨어 구조

본 논문에서는 타일 트래버설 방식을 기반으로 하여 메모리와 텍스처 캐쉬의 효율성을 높이고, 파이프라인 구성을 기반으로 하여 고성능의 스캔 컨버전 처리가 가능한 구조를 제안하고자 한다.

3.1 스캔라인 방식의 파이프라인 구조

제안된 구조를 살펴보기에 앞서 기본적인 변과 스캔에 대한 파이프라인 구성을 살펴보면 [그림4]와 같다.



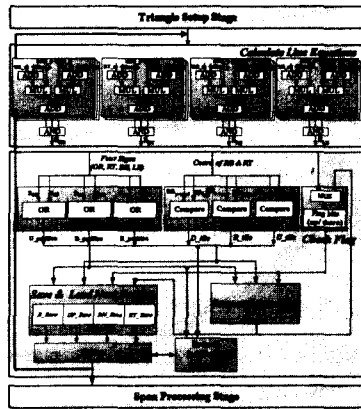
[그림 4] 변 생성과 스캔 처리 파이프라인

변 생성 단계에서는 삼각형을 스캔으로, 선을 픽셀로 분할하는 역할을 담당하고, 스캔 처리 단계에서는 스캔 라인을 따라 연속적으로 프래그먼트를 생성하는 역할을 하며, 각 단마다 처리되어야 할 요소들은 병렬로 동시에 처리된다. 위 구조에서 각 단마다 지연시간 (latency)은 2사이클이고, 1사이클 마다 픽셀을 생성할 수 있는 고성능 구조이다. 3.2절에서 제안된 타일 트래버설 구조는 위 구조의 변 생성 단계와 동일한 역할을 한다. 그러나 트래버설 과정이 복잡하기 때문에 스캔라인 방식의 구조에 비해 긴 지연시간을 갖는다. 대표적으로 타일 트래버설 방식을 채택한 구조인 Neon에서는 스캔라인 방식에 비해 상대적으로 성능이 저하되는 트래버설 부분을 파이프라인으로 구성할 수 없다고 밝히고 있다[5][6]. 그러나, 고성능 3차원 그래픽 가속기를 설계하기 위해서는 메모리와 텍스처 캐쉬의 효율을 향상시키기 위해 타일 트래버설 방식을 채택하는 동시에 빠른 데이터 처리를 위해 스캔라인 방식과 같이 파이프라인 구조를 가진 트래버설 유닛을 구성해야 한다.

3.2 제안하는 타일 트래버설 하드웨어 구조

본 논문에서 제안하는 구조는 타일 트래버설에 있어 분기 예측을 적용한 파이프라인 구조이다. 이 구조에서는 타일 트래버설 하는데 총 3사이클의 지연시간이 걸린다. 고성능 타일 트래버설을 수행하기 위해서는 최소한 1사이클마다 스텝틀 하나가 처리되어야 하는데 이 방식은 트래버설 중간에 분기하는 부분이 삼각형의 크기에 따라 10~50%정도를 차지하고 있고, 분기 비율은 삼각형의 크기가 작아질수록 증가한다. 따라서 분기하는 부분마다 3사이클씩 손실 (penalty)이 생겨 파이프라인 효율이 급격하게 떨어지게 된다.

[그림5]에서 보여지는 제안된 구조는 크게 2단계로 이루어져 있다. 첫 번째 단계에서는 반 평면 함수에 대한 처리를

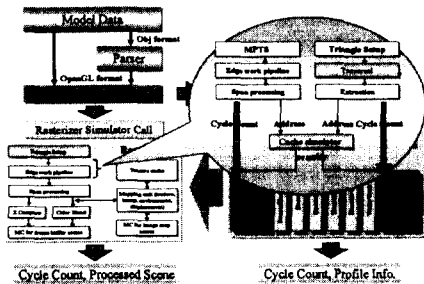


[그림 5] 제안하는 타일 트래버설 하드웨어 구조

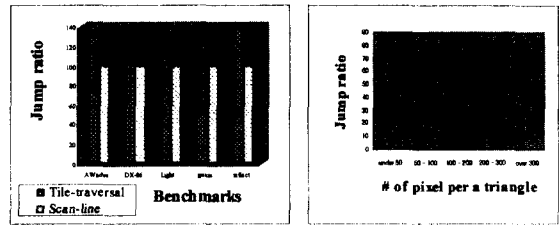
담당하고, 두 번째 단계에서는 최종 이동 방향을 결정하기 위한 플래그 비트를 조사하는 부분으로서, 저장된 context 내용과 타일 경계 및 방향에 대한 조사를 위한 부분으로 구성되어 있다. 다음은 이 구조에서의 전체적인 동작 방식에 대해 설명하겠다. 이전 단계로부터 입력된 각 스템프의 좌표와 변화 값이 변 함수 유닛으로 들어와 계산이 되면 위, 아래, 오른쪽 방향에 대한 진행여부가 가능한지를 알 수 있다. 두 번째 단계에서는 context의 유효 여부에 대한 조사와, 타일 경계의 조사 및 방향성 조사를 하여 최종 진행 방향을 결정하게 되고, 이동에 필요한 4개의 context들은 별도의 레지스터(register)에 저장된다. 기본 이동 방향은 오른쪽 방향이고 타일 경계 부분에 도달하면 분기가 발생하게 된다. 트래버설 할 때 스템프가 분기하는 부분은 타일의 크기와 진행 순서가 미리 결정되어 있기 때문에 경우마다 어디로 이동할 것인지에 대해 예측이 가능하다. 따라서, 분기 예측 제어(control)와 방향 선택 부분이 상호작용 하면서 좌표 값을 미리 계산하고 위치 관계를 파악하여 스템프가 기본 방향을 벗어난 분기가 일어나기 전에 그 다음의 context를 읽어 들임으로써 파이프라인 지연을 최소화할 수 있다. 그러므로, 제안하는 구조는 메모리 효율 및 텍스처 캐쉬의 성능을 향상시키는 동시에 고성능의 스캔 컨버전 처리가 가능하도록 하고 있다.

4. 실험 및 결과 분석

본 논문에서는 [그림 6]과 같이 OpenGL과 유사한 공개 API인 Mesa Graphic Library를 기반으로 실험 환경을 구축하였다[7]. 실험 벤치마크로는 Mesa에서 제공하는 데모들과 SPECviewperf에서 제공하는 영상 모델들을 사용하였다[7][8].



[그림 6] 실험 환경



[a] [그림 7] 실험 결과 [b]

[그림7]의 [a]는 각 벤치마크에 대해 스캔라인 방식과 타일 트래버설 방식을 적용했을 경우의 분기 비율을 보여준다. 타일 트래버설 방식의 경우 스캔라인 방식에 비해 약 14~38% 정도 분기 횟을 알 수 있다. [b]는 삼각형의 크기에 따른 분기 비율을 보여주는데 삼각형의 크기가 작을수록 그 비율이 급격히 높아짐을 알 수 있다. 위의 실험 결과를 토대로 본 논문에서는 분기 예측을 기법을 적용하여 분기 비율을 최소화 하는 고성능의 구조를 제안하고 있다.

5. 결론 및 향후 계획

본 논문에서는 고성능 3차원 그래픽 가속기 설계를 목적으로 데이터의 효율적인 처리를 위해 분기 예측 기법을 적용한 파이프라인 방식을 기반으로 하고, 효율적인 메모리 및 텍스처 캐쉬의 사용을 위해 타일 트래버설 방식을 기반으로 한 구조를 제안하였다. 그 결과 Neon 구조와 같이 페이지 크로싱 문제를 최소화 하였고, 평균 30% 정도 발생하는 분기 현상을 통해 초래되는 파이프라인 지연을 최소화하였다. 향후 본 연구에서는 분기 예측 기법이 적용된 구조를 보다 발전 시키고, 이를 영상 기반 또는 볼륨 기반 렌더링 기법에도 적용시킬 예정이다.

6. 참고문헌

[1] Anders Kugler, "The Setup for Triangle Rasterization," 11th EUROGRAPHICS Workshop on Computer Graphics Hardware, August 26-27, Poitiers, France, 1999.
 [2] Foley, Van Dam, Feiner and Hughes, *Computer Graphics PRINCIPLES AND PRACTICE SECOND EDITION in C*, ADDISON WESLEY, 1997.
 [3] Donald Hearn and M. Pauline Baker, *COMPUTER GRAPHICS C version Second Edition*, PRENTICE HALL, p883-885, 1997.
 [4] Juan Pineda, "A Parallel Algorithm for Polygon Rasterization," SIGGRAPH 88 Conference Proceedings, ACM Press, New York, p17-20, August 1988.
 [5] Joel McCormack, Robert McNamara, Cris Gianos, Larry Seiler, Norman Jouppi, Ken Correll, Todd Dutton & John Zurawski, "Neon: A (Big) (Fast) Single-Chip 3D Workstation Graphics Accelerator," Research Report 98/1, WRL, Compaq Computer Corporation, July 1999.
 [6] Joel McCormack and Robert McNamara, "Tiled Polygon Traversal Using Half-Plane edge Functions," 2000 SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware, Interlaken, Switzerland, p15-21, August 2000.
 [7] Mesa library, <http://www.mesa3d.org>
 [8] SPECviewperf™, <http://www.specbench.org/gpc/opc.static/opview.htm>
 [9] NVIDIA, *Transform, Lighting and Rasterization System Embodied on A Single Semiconductor Platform*. US patent, US6198488 B1, Mar. 6, 2001.