

다중 프로세서와 다중 GPGPU의 스케줄링을 이용한 고속 홀로그램 생성 방법

*이윤혁 **서영호 *김동욱

*광운대학교 전자재료공학과 **광운대학교 인제니움 학부

winner9100@kw.ac.kr

A High Speed Hologram Generation Method Using Scheduling of Multi-GPGPU and Multi-Processor

*Lee, Yoon-Hyuk **Seo, Young-Ho *Kim, Dong-Wook

*Dept. of Electronic Materials Eng., Kwangwoon Uni.

**Ingenium college of liberal arts, Kwangwoon Uni.

요약

홀로그램을 생성하기 위해서 많은 양의 계산을 필요하기 때문에 고속 홀로그램 생성 방법이 필요하다. 본 논문에서는 다중 프로세서와 다중 GPGPU의 스케줄링을 이용하여 고속화 하는 방법을 제안하고 구현하였다. 다중 프로세서를 이용하여 입력과 출력부분을 나누어 동기화 동작을 줄이고, 버퍼를 이용하여 커널과 커널 사이의 대기 시간을 줄일 수 있도록 스케줄링 하였다. nVidia사의 GTX680(Kepler구조) 2개를 이용하여 구현하였을 때, 이전 연구에서 제안한 방법에 비하여 약 70% 정도 계산시간을 줄일 수 있다.

1. 서론

컴퓨터 생성 홀로그램(CGH)은 기준파(reference wave)와 기준파가 물체에 반사되어 나오는 물체파(object wave)의 간섭현상을 수학적으로 모델링 하여 계산된 홀로그램이다. 이 기술은 하나의 물체 점(object point)에서 홀로그램 평면에 모든 화소에 대하여 계산하고 모든 물체 점에 대하여 반복 수행하여 누적하기 때문에 물체 점과 홀로그램의 크기에 따라 계산 량이 급격히 증가한다[1]. 방대한 계산 량으로 인하여 소프트웨어만으로 실시간 홀로그램을 생성하는 것은 매우 어렵기 때문에 특정 하드웨어를 이용하여 생성하는 방법들이 연구되고 있다[1][2]. 이러한 연구들은 크게 두 가지로 분류할 수 있는데, 그 중 첫 번째는 FPGA또는 ASIC을 이용하여 전용 칩으로 구현하는 방법이 있고, 두 번째는 병렬프로세서 기반의 소프트웨어 방식이 있다. 본 논문에서는 병렬프로세서 기반인 GPGPU(General Purpose Graphic Processing Unit)와 다중 프로세서의 스케줄링을 이용하여 가속화 방법을 제안하고, 기존의 스케줄링을 이용한 가속화 방법과 비교하였다.

2. GPGPU를 이용한 스케줄링 방법

2.1 커널 내에서의 스케줄링 방법

이전 연구[3]에서 제안한 커널 내에서의 스케줄링 방법을 구현하기 위해 커널 내의 쓰레드는 2차원의 NxN의 구성해야 한다. 이 방법은 총 3단계로 구성되는데, 각 단계별 쓰레드의 동작은 그림 1에 나타내었다. 그림 1(a)는 첫 번째 단계인 N개의 물체 점의 정보를 GPGPU

의 글로벌 메모리[4]에서 공유 메모리[4]로 전송하는 단계이다. 이 때 첫 행의 쓰레드만 메모리 복사 동작하므로 N을 위프[4]단위로 설정하는 것이 유리하다. 그림 1(b)와 (c)는 두 번째 단계인 공통행[3]을 계산하여 다시 공유메모리에 저장하는 단계이다. 그림 1(b)는 x축에 대한 공통행으로 각 열은 x좌표의 변화에 따른 공통행이고, 각 행은 N개의 물체 점에 대한 공통행이다. 그림 1(c)는 y축에 대한 공통행으로 그림 1(b)와 행과 열의 역할이 반대이다. 그림 1(d)는 마지막 단계로 미리 구한 공통행을 이용하여 NxN의 홀로그램 블록을 N개의 물체 점에 대하여 누적하는 단계이다.

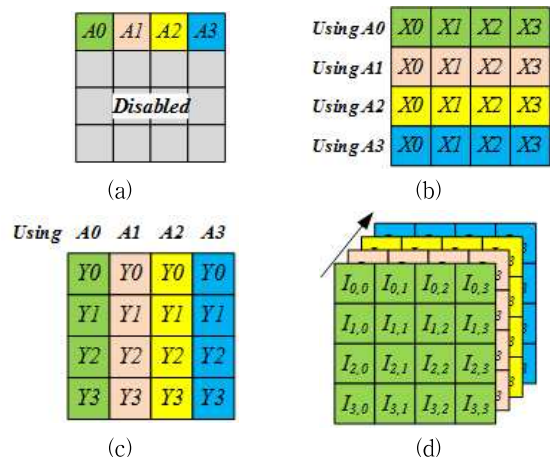


그림 1. 커널 내 스케줄링의 단계별 쓰레드 동작

2.2 장치 간 스케줄링 방법

이전 연구[3]에서 제안한 장치 간 스케줄링은 nVidia사의 Kepler 버전 이상의 GPGPU구조에서 지원하는 직접메모리접근(Direct Memory Access, DMA)을 이용하여 그림 2와 같이 중간 홀로그램의 누적과정을 제거 할 수 있다. 이 방법은 호스트와 모든 GPGPU를 파이프라인화하여 가속하는 방법이다.

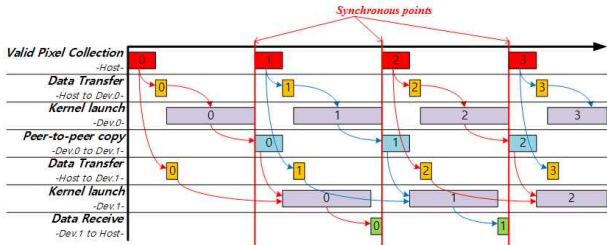


그림 2. 직접메모리접근(DMA)을 이용한 장치 간 스케줄링 방법

2.3 제안하는 스케줄링 방법

이전 연구[3]에서 제안한 방법은 호스트의 프로세스는 하나로 구성되어 있어 동기화 동작 및 유효 물체 점을 나누는 과정 등 커널 전후처리 과정으로 인하여 커널과 커널 사이에 GPGPU의 대기상태가 생긴다. 제안하는 방법은 호스트를 다중 프로세스로 구성하여 그림 3과 같이 호스트의 한 쓰레드는 유효 물체 점의 정보를 나누어 버퍼에 입력하고, 다른 쓰레드는 커널 하나에서 계산이 완료될 때 동기화하여 출력하도록 구성한다. 이 방법은 입력, 출력, GPGPU가 서로 독립적으로 동작하고 서로 버퍼를 통하여 데이터 전송이 이루어진다. 입력 쪽 호스트 프로세스는 동기화를 하지 않기 때문에 버퍼에 공간이 있을 경우 계속 입력을 줄 수 있고, 하나의 커널이 종료하더라도 버퍼에 있는 데이터를 이용하여 대기상태 없이 바로 다음 커널을 실행할 수 있다. 그림 4에 제안하는 방법의 스케줄을 나타내었다.

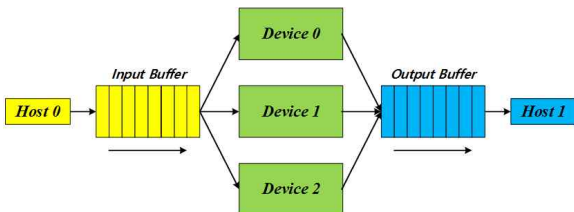


그림 3. 제안하는 다중 프로세서와 GPGPU의 구조

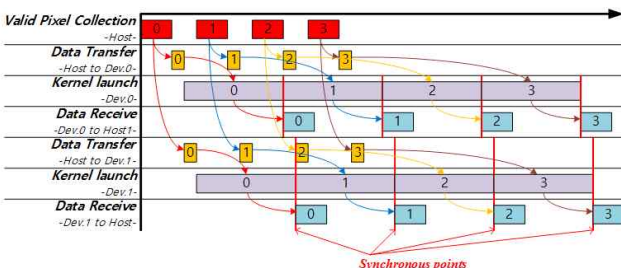


그림 4. 제안하는 방법의 타이밍도.

3. 구현 결과

제안한 방법은 nVidia사의 GTX680 두 개를 사용하고, CUDA와 OpenMP를 이용하여 GPGPU프로그래밍과 다중 프로세서를 구현하였다. 표 1은 이전 연구와 비교 같은 영상으로 홀로그램을 생성한 결과이다. 생성한 홀로그램의 크기는 모두 1,024 x 1,024 크기로 고정하였다. 제안하는 방법을 적용하였을 경우 약 70%의 속도개선이 있다.

표 1. 이전 연구와의 비교

영상	유효 물체점수	이전 연구[3]	제안한 방법
Rabbit	6,123	150.7 ms	52.7 ms
Sujin[5]	12,624	305.7 ms	95.3 ms
Baby[6]	15,554	382.8 ms	116.8 ms
Billiard[7]	24,024	571.5 ms	179.5 ms

4. 결론

본 논문에서는 다중 프로세서와 다중 GPGPU의 스케줄링을 통하여 커널의 대기시간을 줄여 고속화 하는 방법을 제안하였다. 제안한 방법을 이용하여 구현 결과 이전 연구에 비하여 약 70% 계산시간을 감소시키는 것을 확인하였다.

감사의글

이 논문은 2014년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2014R1A2A1A11052433).

참고문헌

- [1] Y.H. Lee, Y.H. Seo, J. S. Yoo, D. W. Kim, "High-Performance Computer Generated Hologram by Optimized Implementation of Parallel GPGPUs", Journal of the Optical Society of Korea, Vol. 18, No. 6, Dec. 2014.
- [2] Y.H. Seo, Y.H. Lee, D.W. Kim, "ASIC chipset design to generate block-based complex holographic video", Applied Optics, Vol. 56, No. 9, March. 2017.
- [3] Y.H. Lee, Y.H. Seo, D.W. Kim, "Fast Hologram Generation Method Using Scheduling of Multi-GPGPUs", The Korean Institute of Broadcast and Media Engineers Summer Conference, 2016, F1.3-13.
- [4] nVidia CUDA Toolkit on-line documentation, <http://docs.nvidia.com/cuda/index.html>
- [5] Y.H. Seo, Y.H. Lee, J.M. Koo, W.Y. Kim, J. S. Yoo, D. W. Kim, "Digital holographic video service system for natural color scene", Optical Engineering, SPIE, Vol.52(11), Nov. 2013.
- [6] <http://www.dofpro.com/cgigallery.htm/>
- [7] <http://vision.middlebury.edu/stereo/data/>