다중 GPU기반 홀로그램 생성을 위한 병렬처리 성능 최적화 기법

An Optimization Method for Hologram Generation on Multiple GPU-based Parallel Processing

국중진*

(Joongjin Kook)

요 약

홀로그램의 생성을 위한 연산은 포인트 클라우드의 규모에 따라 연산량이 기하급수적으로 증가하기 때문에 최근에는 다중의 GPU를 기반으로 CUDA 또는 OpenCL 라이브러리를 활용한 병렬처리가 이루어지고 있다. GPU기반의 병렬처리를 위한 CUDA 커널은 GPU의 코어 개수와 메모리 크기를 고려하여 쓰레드(thread), 블록(block), 그리드(grid)를 구성해야 하며, 다중 GPU 환경인 경우 GPU의 개수에 따른 그리드, 블록, 또는 쓰레드 단위의 분산처리가 필요하다. 본 논문에서는 CGH 생성에 대한 성능평가를 위해 포인트 클라우드의 포인트 개수를 10~1,000,000개 범위에서 점진적으로 증가시키면서 CPU, 단일 GPU, 다중 GPU 환경에서 연산 속도를 비교해 보았으며, 다중 GPU 환경에서 CGH(Computer Generated Hologram) 생성 연산을 가속화하기 위한 CUDA 기반의 병렬처리 과정에서 요구되는 메모리 구조 설계와 연산 방법을 제안한다.

■ 중심어 : CGH, 홀로그램, GPU, 병렬처리

Abstract

Since the computational complexity for hologram generation increases exponentially with respect to the size of the point cloud, parallel processing using CUDA and/or OpenCL library based on multiple GPUs has recently become popular. The CUDA kernel for parallelization needs to consist of threads, blocks, and grids properly in accordance with the number of cores and the memory size in the GPU. In addition, in case of multiple GPU environments, the distribution in grid-by-grid, in block-by-block, or in thread-by-thread is needed according to the number of GPUs. In order to evaluate the performance of CGH generation, we compared the computational speed in CPU, in single GPU, and in multi-GPU environments by gradually increasing the number of points in a point cloud from 10 to 1,000,000. We also present a memory structure design and a calculation method required in the CUDA-based parallel processing to accelerate the CGH (Computer Generated Hologram) generation operation in multiple GPU environments.

■ keywords : CGH, Hologram, GPU, Parallel Processing

Ⅰ. 서 론

디지털 홀로그래피 기술은 홀로그램 영상을 획득하는 기술과 홀로그램 영상을 디스플레이 하는 기술로 나눌 수 있다. 디지털 홀로그램 영상의 획득 방법에는 실제 사물의 3차원 정보를 획득하는 방법과 컴퓨터 그래픽으로부터 영상을 획득하는 방법이 있으나, 실제 사물에서 직접적으로 홀로그램 영상을 획득하는 기술은 간섭 현상을 응용하는 만큼 기술적인 한계성이 존재한다.

최근에는 컴퓨터의 하드웨어 사양과 컴퓨팅 기술이 급속한 발전에 힘입어 컴퓨터 기반의 홀로그램 생성(Computer-Generated Hologram: CGH) 기술은 디지털 홀로그래픽 디스플레이에 적용 가능한 다양한 홀로그램 콘텐츠의 생성을 가능하게 하고 있다. 실제 사물을 직접적으로 획득할 수 있는 기술로서는 주로 광 스캐닝 홀로그래피(Optical Scanning Holography)와 위상이동 간섭계(Phase Shifting Interferometer)를 이용한 홀로그램 영상 획득 기술이 대표적이다[1]. 디지털 홀로그램 영상 디스플레이 기술에 있어서 가장 기본적이며 핵심적인 소자는 빛의 진폭 또는 위상을 표현할 수 있는 소자인 공간 광 변조기(Spatial Light Modulator: SLM)이며, 이를 기본적인 영상 표시 장치로 구성하여 3차원 홀로그

* 정회원, 상명대학교 정보보안공학과

접수일자 : 2019년 05월 16일
수정일자 : 2019년 05월 22일
게재확정일 : 2019년 05월 23일
교신저자 : 국중진 e-mail : kook@smu.ac.kr

램 영상을 형성한다[2].

CGH는 3차원 객체에 대한 빛의 파장을 기록하고 재구성할 수 있도록 한다. 하지만 CGH의 해상도가 파장 차수(wavelength-order)의 해상도이기 때문에 큰 면적 및 높은 해상도를 갖는 SLM(Spatial Light Modulator)이 요구되어 디지털 홀로그래피 기반의 3D 디스플레이 시스템을 개발하는 데 제약이 따르며, 또한 CGH를 계산하는 데 필요한 엄청난 연산량도 극복해야 할 문제이다[3].

디지털 홀로그램 생성을 위한 대표적인 방법은 R-S 적분(Rayleigh-Sommerfeld Integral)이며, Fresnel 홀로그램과 Fourier 홀로그램 등은 이를 기반으로 한 근사화 방법들이다.

디지털 홀로그램 생성 과정의 연산 부하를 감소시키기 위한 방법으로 FGPA 기반의 가속화 기기인 HORN(Holographic Reconstruction)이 제안되었으나 FPGA를 이용하면 구현 과정이 매우 복잡하고 개발 기간이 오래 걸린다는 단점이 있다[4].

2007년 Nvidia는 CUDA라 불리는 새로운 GPU 아키텍처와 SDK(Software Development Kit)를 발표하였으며, GPU를 기반으로 연산을 가속화하기 위한 방안들이 다수 연구되었다[9-11].

CGH의 생성을 위해 GPU를 이용하는 경우 연산의 병렬화를 위한 메모리 구조의 설계가 요구된다. CUDA는 그림 1과 같이 그리드(grid), 블록(block), 쓰레드(thread)가 계층적으로 구성되어 병렬처리가 이루어지며, 동시에 처리 가능한 쓰레드의 개수를 나타내는 워프(warp) 단위를 고려하여 최적의 연산이 수행될 수 있도록 적절한 병렬화가 이루어져야 한다.

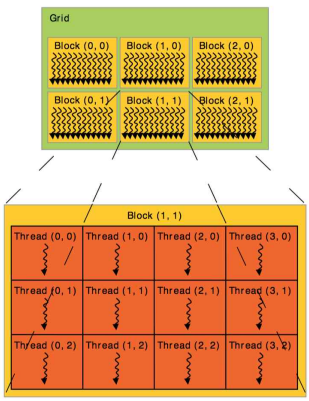

그림 1. Nvidia CUDA의 그리드(grid), 블록(block), 쓰레드(thread) 구조

본 논문에서는 CGH의 생성을 위한 GPU 기반 연산의 효율성을 위해 요구되는 메모리 설계 방법과 이를 토대로 한 CGH 연산의 성능을 평가해 보았다. GPU를 기반으로 한 병렬처리를 위해 CGH 생성 과정을 단계별로 모듈화하고, 독립적 연산이 가능한 모듈에 대해 각각의 CUDA 커널을 구성하였다. 또한, 다중 GPU의 지원을 위해 GPU가 여러 개인 환경에서는 전체 포인트 클라우드를 GPU 개수로 나누어 GPU 별로 분산 할당하여 연산이 가능하도록 설계하였다.

CPU, 단일 GPU, 다중 GPU 환경에서 10~100,000개의 포인트로 구성된 CGH 생성 과정에 소요되는 시간을 측정하여 비교하였으며, 포인트의 개수가 10,000개일 때 CPU 대비 약 3배 빠른 연산 속도를 나타냈으며, 20,000개 이상이 되었을 때 단일 GPU 대비 다중 GPU가 더 나은 성능을 나타냄을 확인할 수 있었다.

Ⅱ. 관련연구

N개의 포인트로 구성된 3D 객체에 대해 CGH의 계산을 위한 공식은 다음과 같이 표현될 수 있다.

$$I(x_h,\ y_h)=\sum_{j}^{N}A_j\cos\left(\frac{2\pi}{\lambda}\left(\frac{(px_h-px_j)^2+(py_h-py_j)^2}{2z_j}\right)\right) \quad (1)$$

식 (1)에서 I(xh, yh)는 CGH에서 빛의 강도이며, (xh, yh)와 (xj, yj, zj)는 CGH와 3D 객체의 좌표를 나타낸다. Aj는 3D 객체의 빛의 강도를 나타내며, λ는 참조 광원의 파장을 나타내며, $P_j=\pi p^2/(\lambda z_j)$에서 p는 CGH 평면에서 샘플링 간격을 나타낸다. 식 (1)의 연산 복잡도를 Big-O 표기법으로 표현하면 $O(NN_xN_y)$이며, N_xN_y는 CGH에서 x, y축 표본의 개수를 나타낸다[5].

간섭성 홀로그래픽 스테레오그램(Coherent Holographic Stereogram)은 디지털 홀로그램 생성 알고리즘 중 하나로써 R-S 적분 기반의 알고리즘보다 연산속도가 월등히 빠르다. 연산 가속화를 위한 주요 특징은 다음과 같다.

- 디지털 홀로그램 평면을 작은 세그먼트들로 분할한다.
- 3차원 객체를 구성하는 각 포인트로부터 홀로그램 평면의 각 세그먼트에 대한 공간 주파수 계산 과정의 연산량을 감소시킨다.
- 분할된 홀로그램의 주파수 평면에 각 포인트에 대한 복소 진폭 분포를 기록하고 각각의 분할된 홀로그램 주파수 평면을 고속 푸리에 변환(FFT)으로 처리한다.

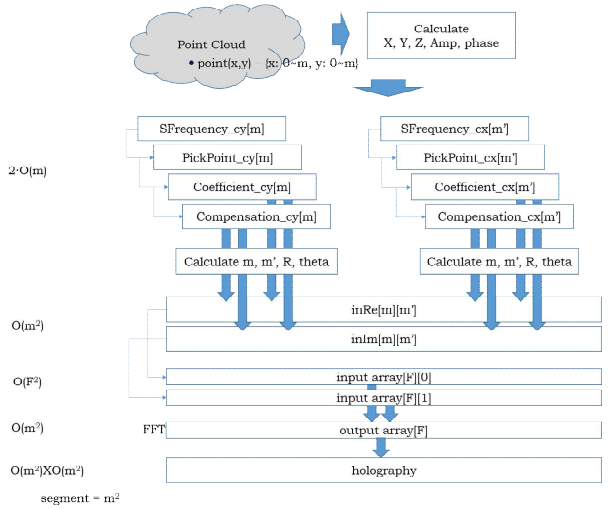

그림 2. 홀로그램 생성을 위한 CGH 연산 과정

CS기반의 홀로그램 생성 알고리즘은 PAS(Phase-added stereogram), CPAS(Compensated phase-added stereogram), APAS(Accurate phase-added stereogram), ACPAS(Accurate compensated phase-added stereogram), FPAS(Fast phase-added stereogram) 등으로 다양하게 발전되었다[6].

ACPAS는 APAS와 CPAS의 특성을 결합한 근사화 알고리즘이지만, 정확도 및 복원 영상의 화질은 R-S 기반의 복원 영상과 매우 유사하여 고속 및 고화질의 특성을 갖는 홀로그래픽 디스플레이 장치에 적합한 알고리즘이다.

CS 기반의 홀로그램 생성 알고리즘이 다양하게 발전됨에 따라 기존의 R-S 방식과 비교하여 연산 속도를 향상시켰지만, 그럼에도 불구하고 고화질과 실시간성을 요구하는 디지털 홀로그램 생성을 위해서는 더욱 빠른 연산 속도가 뒷받침되어야 한다.

일본의 치바대학에서는 1992년부터 FPGA를 기반으로 하는 홀로그램 생성기 HORN을 개발하였다. 가장 최근인 2008년에 개발된 HORN-6는 4개의 고성능 FPGA가 내장되어 있으며, 연산 속도 향상을 위해 Lookup 테이블 기반의 홀로그램 생성 연산을 수행하였다. 하지만 FPGA 기반의 홀로그램 생성은 구현 과정이 매우 복잡하고 개발 기간이 오래 걸린다는 단점이 있다[4].

2007년 Nvidia는 CUDA라 불리는 새로운 GPU 아키텍처와 SDK를 발표하였으며, GPU를 기반으로 연산을 가속화하기 위한 방안들이 다수 연구되었다.

CUDA는 SIMT(Single Instruction Multiple Threads)를 기반으로 데이터의 병렬 처리를 지원한다. CUDA에서는 병렬처리 대상이 되는 쓰레드의 집합을 그리드(grid)로 구성하는데, 그리드(grid)는 그림 2와 같이 하나의 그리드(grid)는 다수의 블록들로 구성되며, 하나의 블록은 다시 다수의 Thread들로 구성되는 형태이다. CUDA의 코어는 코어별로 독립된 Thread를 처리하는 형태로 병렬처리가 수행된다.

CUDASW++는 저사양의 GPU 환경에서도 CUDA를 기반으로 연산속도를 가속화하기 위한 라이브러리를 제안하였다[7]. 이 논문에서는Smith-Waterman 알고리즘 기반의 데이터베이스 검색 성능의 비교를 위해 여러가지 사양의 GPU, 그리고 단일 GPU와 다중 GPU를 통해 성능을 비교하였다.

2008년에 발표된 작업 기반(task-based) 및 데이터 기반(data-based) 병렬 컴퓨팅을 수행하기 위한 OpenCL을 기반으로 홀로그램 생성을 위한 연구가 이루어지기도 하였다[8].

Ⅲ. CGH 생성을 위한 메모리 구조와 알고리즘

1. CGH 생성을 위한 연산 과정

CGH 생성을 위한 포인트 클라우드의 구성과 연산과정을 도식화하면 그림 3과 같이 나타낼 수 있다.

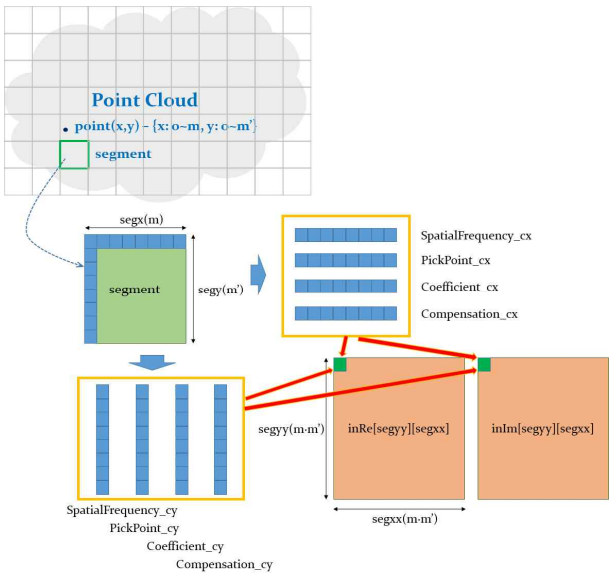

그림 3. 포인트 클라우드와 CGH 연산 절차

포인트 클라우드를 구성하는 각 포인트들을 세그먼트 단위로 그룹화하고 각 세그먼트에 포함된 포인트별로 식(1)의 연산을 거쳐 간섭패턴(interference pattern)을 계산하여 2차원 복소평면에 누적하는 연산이 반복적으로 이루어진다.

CPU를 기반으로 CGH 연산을 수행하기 위해서는 포인트 클라우드의 각 포인트에 대해 그림 4와 같이 반복적인 연산이 이루어져야 한다.

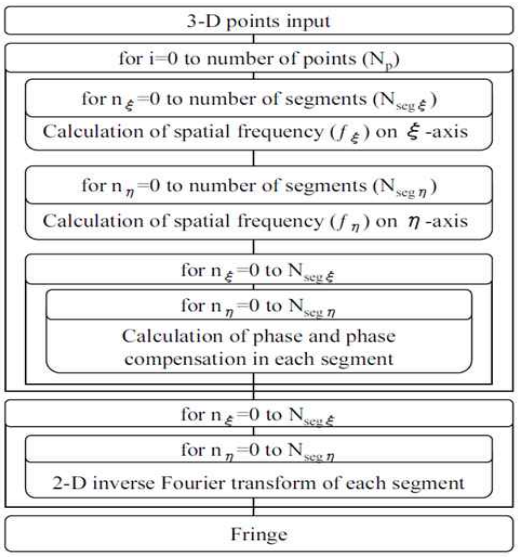

그림 4. CPU 기반의 CGH 생성 절차

반면 GPU를 기반으로 병렬처리를 수행하는 경우에는 그림 5와 같이 각 포인트에 대해 쓰레드와 블록을 지정하여 그리드(grid)를 구성함으로써 CUDA 기반의 병렬처리가 이루어지도록 해야 한다.

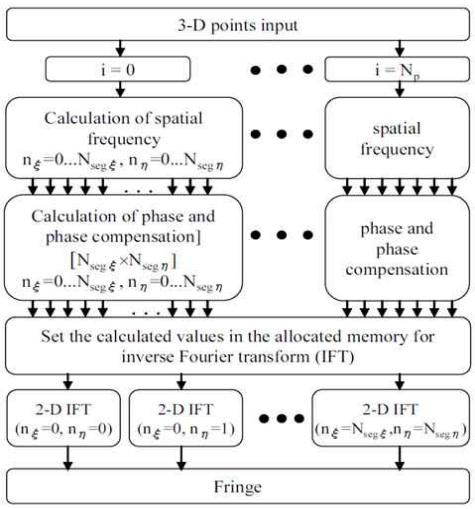

그림 5. GPU 기반의 CGH 생성 절차

CUDA를 기반으로 병렬처리가 이루어지는 과정에서 시스템 메모리와 GPU 메모리 간 데이터의 교환 과정이 필요하다. 포인트 클라우드를 세그먼트 단위로 분할하여 시스템 메모리에 적재하는 전처리 과정을 거친 후, 이 데이터는 병렬처리를 위해 GPU의 메모리로 전송되어야 한다. CUDA에서는 병렬처리를 위한 커널(kernel)에 의해 연산이 이루어지며, 연산의 결과는 다시 시스템 메모리로 전송된다. 그림 6은 시스템 메모리와 GPU 메모리 간 기본적인 데이터 교환 과정을 나타낸다.

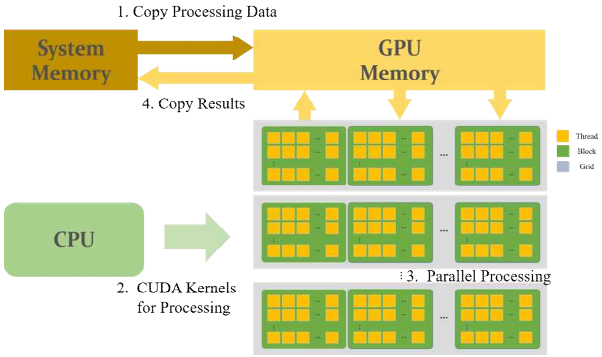

그림 6. 시스템 메모리와 GPU 메모리 간 데이터 전송 및 연산 과정

2. 다중 GPU 기반의 CGH 생성 연산

다중 GPU를 기반으로 홀로그램 생성을 가속화하기 위해서는 개념적으로 그림 7과 같이 포인트 클라우드를 GPU 개수만큼의 영역으로 분할한 다음 각 GPU에서 분할된 영역을 전담하여 병렬적인 연산이 이루어지도록 해야 한다. 이 때, 각 GPU는 개별적으로 자신의 메모리에서 연산을 수행하게 되지만, 최종적으로는 각 GPU에서 수행된 연산의 결과를 병합하는 절차가 요구된다.

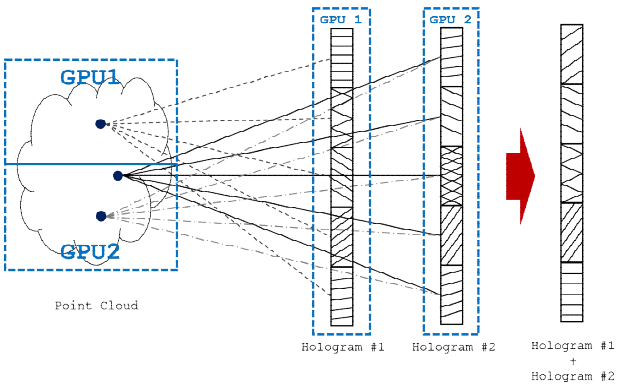

그림 7. GPU 기반의 CGH 생성을 위한 분할 연산과 병합

이러한 병렬처리는 각각의 GPU에 대해 개별적인 메모리 할당과 더불어 최종적으로 각각의 GPU에서 수행된 연산의 결과를 병합시켜야 하기 때문에 GPU 개수에 비례하여 메모리 참조 횟수가 증가하게 된다.

다중 GPU 기반의 병렬처리는 메모리 참조 횟수의 증가와 더불어 다수의 쓰레드에 의한 경쟁상황(race condition)을 유발한다. 그림 8과 같이 각 포인트에 대한 spatial frequency, phase, compensation 연산은 독립적으로 수행 가능하지만, 2차원 평면에 대한 메모리에 실수부와 허수부 값이 순차적으로 기록되는 과정과 FFT 연산을 수행한 결과가 누적되는 과정은 모두 전역 메모리에서 수행이 이루어지므로 병렬화가 불가능하

다.

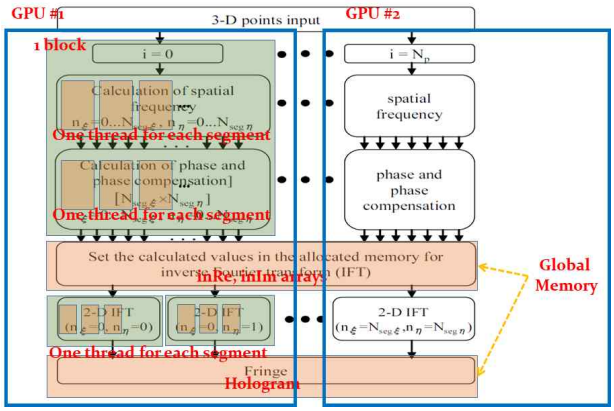

그림 8. 다중 GPU 기반의 병렬 처리 과정

따라서 각 쓰레드에서 수행된 연산의 결과는 그림 9, 그림 10과 같이 전역 메모리에서 누적 연산이 이루어지게 된다.

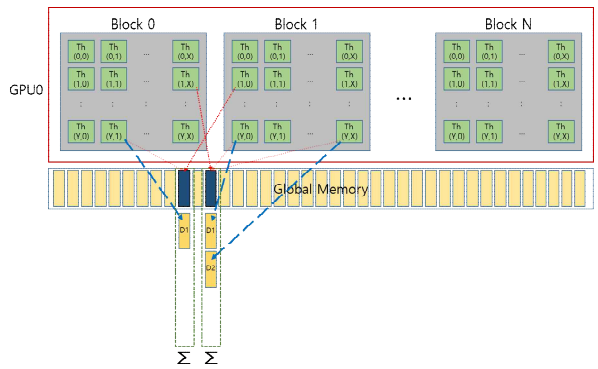

그림 9. 단일 GPU기반 CGH 연산 과정의 전역 메모리 참조

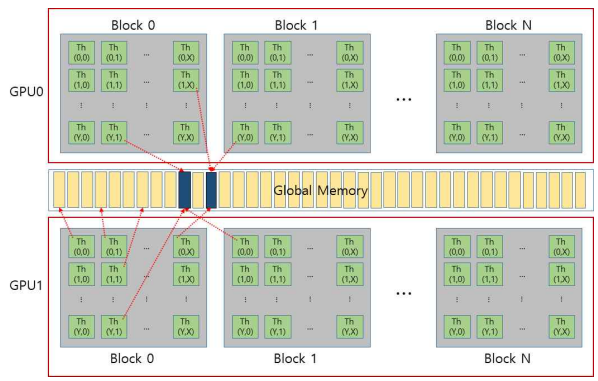

그림 10. 다중 GPU기반 CGH 연산 과정의 전역 메모리 참조

각각의 GPU에서 수행된 연산의 결과가 전역 메모리에서 누적 덧셈이 되는 과정에서 다수의 쓰레드에 의한 동시 접근이 시도되는 경우 데이터의 오염(corruption)이 발생되어 최종 결과는 일관성을 가질 수 없게 된다. 따라서 전역 메모리에 대해 상호배제(mutual exclusion)를 보장할 수 있도록 세마포어(semaphore), 뮤텍스(mutex), 또는 메모리 배리어(memory barrier)와 같은 수단이 추가적으로 사용되어야 한다. 하지만, 이러한 메모리 보호 수단은 결과적으로 연산 속도의 저하를 발생시키는 원인이 된다.

Ⅳ. CGH 생성 성능 평가 및 결론

본 논문에서는 다중 GPU 환경에서의 CUDA 기반 CGH 연산 과정에서 다중 GPU 사용에 대한 성능 향상의 효과를 검증하기 위해 포인트 클라우드의 포인트 개수를 10~1,000,000개 범위에서 점진적으로 증가시키면서 CGH 생성에 대한 성능을 비교해 보았다. 실험을 위한 환경의 구성은 표 1과 같다.

표 1. CGH 생성을 위한 하드웨어 사양

H/W	Model	Quantity
CPU	Intel i7 6700 Quad core 4.0GHz	1
Memory	DDR4 8G PC4-21300	4(32GB)
GPU	MANLI GeForce GTX780 Ti	2

먼저 10~10,000개로 구성된 포인트 클라우드에 대한 CPU 기반의 CGH 연산과 단일 GPU 기반의 CGH 연산 수행 속도를 비교해 본 결과는 그림 11과 같다.

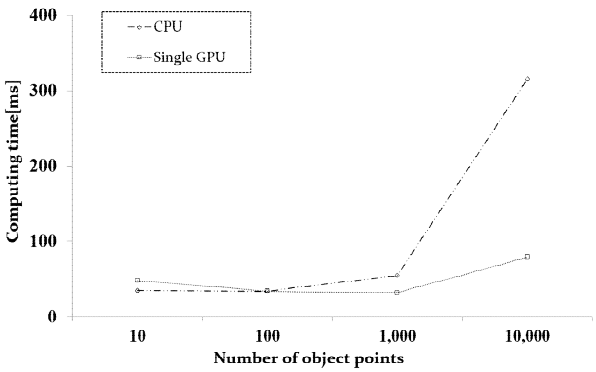

그림 11. CPU와 단일 GPU 기반의 CGH 연산 성능 (포인트 10~10,000개)

그림 12는 100,000개 포인트에 대한 CPU와 단일 GPU 기반 CGH 생성 시간을 비교한 결과이다.

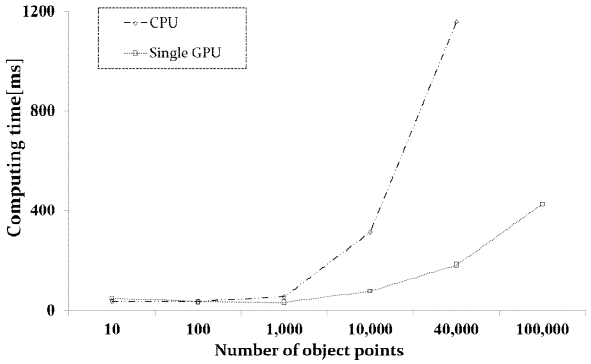

그림 12. CPU와 단일 GPU 기반의 CGH 연산 성능
(포인트 10~100,000개)

CPU와 GPU 기반의 CGH 연산 시간을 비교했을 때, 포인트 개수가 약 100개 이상이 되었을 때 GPU 기반의 연산이 효율적인 것을 알 수 있다. 이러한 결과는 GPU를 기반으로 연산이 수행되는 경우 포인트 데이터를 GPU의 메모리로 이동하는 과정에 따른 오버헤드로 판단할 수 있다.

포인트 개수가 10,000개 이상이 되면 CPU와 GPU 기반 CGH 연산 속도는 급격한 차이가 발생되며, 10,000개 일 때 약 4배, 40,000개 이상일 때부터 약 6.3배 차이가 발생된다.

단일 GPU와 다중 GPU에 대해서도 포인트 개수를 10개부터 1,000,000개까지 변화시켜가며 CGH 생성 시간을 비교해 보았다. 그림 13과 같이 포인트 개수가 10,000개 일 때까지는 단일 GPU 기반의 CGH 생성 성능이 약간 더 우세하였으며, 10,000개 이상이 되었을 때부터 2개의 GPU를 이용한 다중 GPU 기반의 CGH 연산 성능이 더 우수함을 알 수 있다.

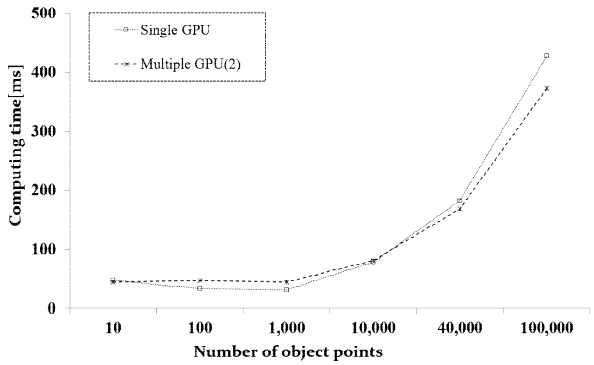

그림 13. 단일 GPU와 다중 GPU 기반의 CGH 연산 성능
(포인트 10~100,000개)

다중 GPU를 기반으로 연산이 이루어질 때에는 각 GPU에서 수행된 연산의 결과를 전역 메모리에서 병합하는 과정에서 오버헤드가 발생하기 때문에 포인트의 개수가 일정량 이상이 되기 전까지는 오히려 단일 GPU를 사용할 때보다 연산속도가 저하되는 것을 확인할 수 있다.

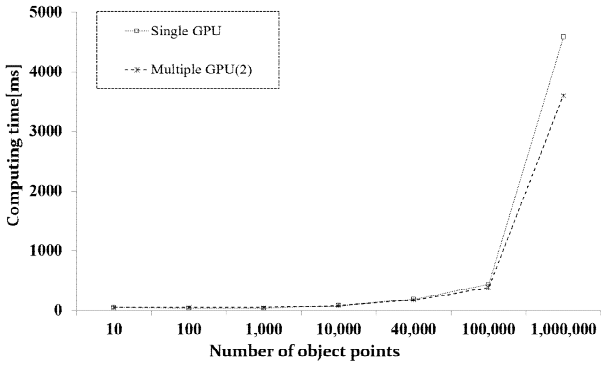

그림 14. 단일 GPU와 다중 GPU 기반의 CGH 연산 성능
(포인트 10~1,000,000개)

그림 14는 단일 GPU 기반의 연산 속도와 두 개의 GPU를 사용했을 때의 연산 속도를 비교하고 있으며, 포인트의 개수가 약 100,000개 정도일 때까지는 다중 GPU 기반의 병렬 처리에 의한 CGH 생성 연산이 단일 GPU 기반과 비교했을 때 그 차이가 매우 미미하지만, 포인트의 개수가 100,000개 이상이 되면서부터 다중 GPU를 이용한 연산속도가 더 우수해지며, 포인트 개수가 1,000,000 인 경우, 두 개의 GPU를 사용하여 병렬처리를 수행하는 것이 단일 GPU를 사용했을 때에 비해 약 22% 향상된 성능을 보였다. 이는 다중 GPU를 기반으로 CGH 연산을 수행하기 위해서는 각각의 GPU 별로 메모리의 할당이 필요하고, 각각의 GPU에서 수행된 연산의 결과를 병합하는 과정 때문에 발생되는 오버헤드로 인한 문제이다. 따라서 다중 GPU 기반 시스템에서 보다 효율적인 홀로그램을 생성하기 위해서는 포인트의 개수에 따라 CPU, 단일 GPU, 또는 다중 GPU를 선택적으로 사용하여 연산이 이루어져야 하며, 다중 GPU 기반인 경우에는 각 GPU에서 수행된 연산의 결과를 통합하는 과정에서 메모리 접근에 대한 동시성 문제를 최소화시킬 수 있는 소프트웨어 설계가 요구된다.

REFERENCES

[1] Peter Wai Ming Tsang, Ting-Chung Poon, “Review on the State-of-the-Art Technologies for Acquisition and Display of Digital Holograms,” *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS*, Vol.12, No.3, pp.886-901. 2016.

[2] 임용준 김진웅, “가상, 증강 및 혼합현실을 위한 디지털 홀로그래피 기술 연구 동향,” 정보통신기술진

홍센터, *주간기술동향-AR, VR, MR*, pp. 1-13, Dec. 2018.
[3] Tomoyoshi S., Tomoyoshi I., Nobuyuki M., Yasuyuki I., and Naoki, T. 2010, "Fast calculation of computer-generated-hologram on AMD HD5000 series GPU and OpenCL," *OSA OPTICS EXPRESS.* Vol. 18, No. 10, pp. 9955-9960, May. 2010.
[4] Tomoyoshi S., Nobuyuki M., Takashige S., Satoru H., Shinobu T., Tomoyoshi I., "Special-purpose computer for holography HORN-3 with PLD technology," *Computer Physics Communications*, Vol. 130, Issues 1 - 2, pp. 75-82, Jul. 2000.
[5] T. Yatagai, "Stereoscopic approach to 3-D display using computer-generated holograms," *Applied Optics*, Vol. 15, pp. 2722-2729, 1976.
[6] M. Yamaguchi, H. Hoshino, T. Honda and N. Ohyama, "phase-added stereogram: calculation of hologram using computer graphic technique," in *Practical holographic VII, S. A. Benton,* ed., Proc. SPIE 1914, pp. 25-33, 1993.
[7] Yongchao Liu*, Douglas L Maskell and Bertil Schmidt, CUDASW++: optimizing Smith-Waterman sequence database searches for CUDA-enabled graphics processing units, *BMC Research Notes*, BioMed Central, May 2009,
[8] Tomoyoshi S., Tomoyoshi I., Nobuyuki M., Yasuyuki I., and Naoki T., "Fast calculation of computer-generated-hologram on AMD HD5000 series GPU and OpenCL" *OPTICS EXPRESS 9955*, Vol. 18, No. 10, pp. 9955-9960, 2010.
[9] 이성욱, 변기범, 김기수, 홍지만, "GPGPU를 이용한 영상 품질 측정 프로그램의 가속화 연구," *스마트미디어저널*, 제5권, 제4호, 69-74쪽, 2016년 12월
[10] 최홍준, 김철홍, "범용 그래픽 처리 장치의 메모리 설계를 위한 그래픽 처리 장치의 메모리 특성 분석," *스마트미디어저널*, 제3권, 제1호, 33-38쪽, 2014년 3월
[11] 손동오, 심규연, 김철홍, "작업 처리 단위 변화에 따른 GPU 성능과 메모리 접근 시간의 관계 분석," *스마트미디어저널*, 제4권, 제4호, 56-63쪽, 2015년 12월

저 자 소 개

국중진(정회원)

2005년 광운대학교 컴퓨터공학과 학사 졸업.
2007년 광운대학교 컴퓨터공학과 석사 졸업.
2012년 숭실대학교 컴퓨터학과 박사 졸업.
2017~현재 상명대학교 정보보안공학과 조교수

<주관심분야 : 임베디드시스템, 운영체제, 센서네트워크, 디지털포렌식, 사물인터넷>