한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 24, No. 3, P.21~31

자연재난 데이터 실감 가시화 시스템

김종용° 정석철 이계원 조준영 김동욱 박상훈*

동국대학교 멀티미디어학과

{khj8499, kkokko, gyeweon, kor9306, kimdongwook, mshpark}@dongguk.edu

Visualization System for Natural Disaster Data

Jongyong Kim° Seokcheol Jeong Gyeweon Lee
Joonyoung Cho Dongwook Kim Sanghun Park*

Department of Multimedia, Dongguk University

요 약

태풍, 해일, 홍수, 범람 등에 관련된 자연재난 데이터를 빠르고 효과적으로 가시화하여 재난 · 재해 상황에서 정확한 의사결정을 할 수 있도록 지원하는 시스템을 소개한다. 재난정보를 포함하는 데이터는 적게는 수백 MB에서 많게는 수십, 수백 GB로 구성되어 있으므로 개인이 지닌 컴퓨터로는 처리할 수 없다. 그렇기 때문에 본 시스템은 클라이언트-서버 기반의 시스템을 제공하여 고성능 서버에서 가시화 결과를 생성하고 클라이언트에서는 결과를 받아 출력하는 형태로 구현되었다. 서버는 클라이언트의 요청을 처리하고 내장된 고성능 클러스터로 렌더링된 결과를 클라이언트로 전송한다. 클라이언트는 원하는 기간을 지정하여 가시화된 결과를 이미지, 동영상, 3D 그래픽 모델 중 원하는 형태로 서버로부터 제공받아 표출할 수 있으며 사용자 친화적인 GUI와 효과적으로 가시화 결과를 볼 수 있는 다양한 기능을 사용자에게 제공한다.

Abstract

We introduces a system that enables fast and effective visualization of natural disaster data such as typhoons, tsunamis, floods, and flooding to help make informed decisions in disaster situations. Data containing disaster information consists of a few hundred megabytes to many tens and hundreds of gigabytes, which can not be handled by a PC. This system was implemented in the form of a client-server based service to generate and output results from high-performance servers. The server in a built-in, high-performance cluster handles client requests and sends the result of visualization to the client. Clients can receive the results in any form of images, videos, or 3D graphic model by specifying a desired time frame, effectively viewing the results with a user-friendly GUI.

키워드: 과학적 가시화, 자연재난 데이터, 병렬 · 분산 컴퓨팅, 클라이언트-서버 컴퓨팅
Keywords: Scientific visualization, Natural disaster data, Parallel · distributed computing, Client-Server computing

1. 서론

태풍, 해일, 홍수, 범람과 같은 자연재난과 기타 관련된 지구환경(earth environment) 데이터는 매우 복잡한 구조를 지니며 다양한 데이터가 축적되어 크기가 매우 방대해지고 있다. 따라서 최신 과학적 가시화(scientific visualization) 기법의 활용과 병렬 · 분산 처리로 방대한 데이터를 빠르게 가시화하고 핵심적인 정보를 효과적으로 제공함으로써 의사결정에 중대한 도움을 주는 시스템의 필요가 증대되고 있다 [1][2]. 최근 정부와 관계부처는 재난 통합관제 시스템을 구축하고 있으며 국내 기업들도 재난/재해 시스템을 출시하고 있다. 국내 기업에서는 GIS(Geographic Information System) 기반의 데이터를 사용하여 지리 정보를 가시화하고 실감나는 시뮬레이션을 위해 3차원 그래픽 모델을 배치하여 정보를 분석하는 시스템이 출시되어 있다. 하지만 아직 초기단계이기 때문에 고가의 외산 소프트웨어를 사용하는 실정이다. 대표적인

*corresponding author:Sanghun Park/Dongguk University(mshpark@dongguk.edu)

Received : 2018.06.22./ Review completed : 1st 2018.06.29. / Accepted : 2018.07.04.
DOI : 10.15701/kcgs.2018.24.3.21
ISSN : 1975-7883(Print)/2383-529X(Online)

외산 소프트웨어가 ArcGIS [3]로 도시 및 지역 계획과 다양한 데이터를 분석할 수 있는 시스템이다.

본 논문은 선행 연구 [4]를 기반으로 각각의 디바이스마다 사용되던 기능들을 보완하고 하나의 통합 가시화 프로그램으로 만드는데 중점을 두고있다. 또한 사용하는 자연재난 데이터에 맞는 렌더링 기법인 마칭큐브(marching cube)와 광선투사(raycasting) 가시화 기법을 추가하였고 클라이언트-서버 기반의 사용 가능한 시스템을 제작하였다. 클라이언트-서버 기반으로 동작되는 시스템은 분산된 데이터를 다수의 PC 클러스터를 이용하여 병렬 · 분산 가시화하도록 구현되었다. 이것은 고성능 클러스터에서 가시화 작업을 수행하고, 사용자는 가시화 결과를 단말기로 전송받아 디스플레이하기 때문에 고가의 장비뿐만 아니라 고성능 GPU가 탑재 되어 있지 않은 저가의 범용 PC나 휴대용 디바이스에서도 높은 품질의 가시화 결과를 대화식으로(interactively) 확인할 수 있다. 또한 통신 프로토콜을 통해 다수의 일반 개발자들이 가시화 서버를 이용하여 서비스를 활용할 수 있는 환경을 구축하였으며, 일반 사용자들은 게임 엔진 기반의 클라이언트 환경에서 가시화를 수행할 수 있다.

본 시스템에서 개발된 가시화 클라이언트는 고성능 클러스터 서버와 연동하도록 구현되었다. 또한 사용자는 가시화 클라이언트의 직관적인 GUI와 부가 기능을 사용하여 서비스를 효과적으로 제공받을 수 있도록 하였다. 더욱 효과적으로 가시화 할 수 있도록 클라이언트의 기능을 추가하고 고성능 클러스터에 원격 접속하여 원하는 가시화 서비스를 요청할 수 있도록 시스템을 개발하였다. 이를 위해 소프트웨어공학 기반의 프레임워크(framework) 방법론을 본 연구에 적용하여 개별 모듈의 독립성이 유지될 수 있는 구조로 전체 시스템을 설계하였다 [5].

최종적으로 사용자들은 클라이언트 프로그램을 이용해 서버에 저장된 특정 지역과 시간(기간)을 설정할 수 있고, 해당 지역과 시간에 대응하는 가시화 가능하도록 준비된 데이터를 선택하고, 가시화를 요청함으로써 기대하는 영상 정보를 확인할 수 있다. 실제 자연재난 상황에서 긴급조치 사항들을 결정해야 하는 결정권자들의 경우, 서버에 저장된 다양한 관측 및 예측 데이터들의 선택적 가시화 결과를 이용해 빠르고 정확한 의사결정을 할 수 있을 것이고 자연재해로 인한 피해를 최소화 할 수 있을 것이다.

고성능 컴퓨터에는 지형 고도정보 데이터(DEM : Digital Elevation Model), 항공촬영 영상 데이터, 기상 관련 실측 데이터, 수문 및 해양 모델 시뮬레이션 데이터, 태풍 예측모델 시뮬레이션 데이터, 인공위성 GPS 기반 데이터 등이 저장되며, 사용자의 권한에 따라 데이터 액세스 정도가 결정될 것이다.

2. 시스템 구조

그림 1은 본 논문에서 소개하는 시스템의 전체 구성도이다. 본 시스템은 클라이언트-서버 기반으로 구현되었다. 고성능 컴퓨터에는 자연재해 데이터를 저장하는 스토리지와 데이터를 가시화할 병렬 클러스터가 있다. 또한 클라이언트의 요청에 응답하고, 스

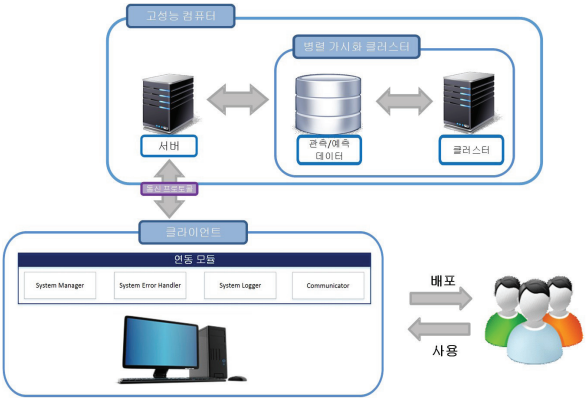


Figure 1: System architecture overview

토리지 검색, 사용자 요청 파악 등의 기능이 포함된 서버로 구성되어있다. 서버는 C/C++ 기반의 소켓 통신을 기반으로 개발되었고, 멀티 스레드를 사용하여 여러 사용자가 접속해도 정상적으로 구동되도록 하였다 [6]. 클라이언트는 서버와 연동하여 고품질의 가시화 결과를 내장된 기능을 활용하여 사용자에게 표출한다 (표 1 참조).

Table 1: System organization

Model	Function
Client	- Users can use the client to view their desired visualization results - Unity3d-based clients
Server	- Connect client and visualization clusters - Perform functions such as data exchange and search
Visualization Cluster	- Storage connectivity with earth environment data - Development of CPU + GPU visualization algorithms based on multi-core and many-cor for high quality and high-speed rendering

2.1 고성능 병렬 가시화 클러스터

그림 1의 관측/예측 데이터는 WRF(Weather Research and Forecasting) 모델의 NetCDF(Network Common Data Format)으로 저장되어있다 [7]. NetCDF는 미국 UCAR(University Corporation for Atmospheric Research)에서 제안된 모델이다. UCAR에서는 NetCDF 파일 쉽게 읽고 사용할 수 있도록 OpenAPI를 제공하기 때문에 사용하기가 용이하다. 관측/예측 데이터를 가시화하기 위해 그래픽 오픈 소스 라이브러리인 VTK(The Visualization ToolKit)를 사용했다 [8]. VTK는 프로그래머에게 데이터에 과학

적 가시화 기법들을 쉽게 적용할 수 있도록 많은 기능을 지원해준다. VTK 라이브러리를 통해 NetCDF파일의 내용을 가시화하기 위해서는 NetCDF파일의 데이터를 VTK 라이브러리가 읽을 수 있는 형태의 데이터(.vtk)로 변환해야한다. 따라서 클러스터 내에 병렬 가시화 프로그램과 별개로 데이터 변환 역할을 해내는 프로그램을 추가하여 존재하는 모든 NetCDF 파일을 읽어 vtk 데이터 파일로 변환한 후 저장한다. 서버는 백그라운드(background)에서 실행되고 있으며 클라이언트의 작업 요청시 병렬 가시화 프로그램(클러스터)을 호출한다. 서버는 클라이언트로부터 전달받은 파라미터(parameter)를 가시화 프로그램에 전달하기 위해 텍스트 파일(.txt)로 파라미터들을 저장하고 가시화 프로그램은 텍스트 파일을 읽어 작업을 수행한다. 표 2는 저장되는 파라미터로 클라이언트에서 요청한 가시화 기간을 특정 짓기 위해 시작 날과 마지막 날을 저장한다. 또한 이미지, 동영상, 3D 그래픽 모델 중 어느 형태로 가시화하여 클라이언트로 전송할지를 저장하는 부분이 있다.

Table 2: Parameters for data access

Contents	Example
Start date of data to be visualized	2008010100
End date of data to be visualized	2008010120
Type of input data	10
Types of output results	100

클라이언트와 서버의 통신은 정확한 데이터 전달을 위해 TCP/IP 소켓 통신을 사용하였다. 기본적으로 클라이언트는 대용량 데이터의 렌더링된 결과를 얻기 위해서 보고자 하는 데이터가 존재하는지를 확인해야 한다. 그러기 위해서는 우선 클라이언트에서 서버로 데이터가 존재하는지 확인하는 절차가 필수적이다. 서버는 클라이언트로부터 데이터가 있는지 확인 요청을 받으면 자료가 저장되어 있는 영역을 탐색해 데이터의 유무를 클라이언트로 전송하여야 한다. 그 후에 클라이언트는 데이터의 유무를 전송 받고 없다면 다른 데이터를 요청하거나, 데이터가 존재한다면 렌더링 결과를 서버에 요청해야 한다. 서버는 요청 받은 데이터의 형식과 렌더링 결과의 형태, 범위 등의 정보을 전달 받고 병렬 가시화 프로그램을 실행해 결과를 클라이언트로 전송한다.

2.2 클라이언트

클라이언트는 Unity3d 기반으로 제작되었다 [9]. Unity3d는 게임 제작 뿐만 아니라 실내외 건축 디자인과 애니메이션 등 비게임 분야에도 진출하여 다양한 분야에서 사용되고 있으며 여러 디바이스로 빌드가 가능하므로 빌드 별로 수정을 가하면 다양한 디바이스에서도 사용 가능하다. 막강한 그래픽 기능과 C# 기반의 스크립트를 사용한 심도있는 엔진 사용으로 프로그램의 개발의 시간을 단축하였으며 가시화 기능과 품질을 향상 시킬수 있게 되었다.

3. 대용량 시간가변 데이터를 위한 병렬 · 분산 가시화

본 장에서는 가시화 클러스터에서 사용하는 데이터 가시화 알고리즘에 대해 서술한다. 과학적 가시화 알고리즘은 데이터에 저장된 복셀 타입에 따라 나누었다. 본 시스템에서는 복셀의 값이 스칼라이므로 컬러 맵핑(mapping) [10]과 볼륨 렌더링 기법인 마칭큐브, 광선 투사법을 사용하였다 [11]. 병렬 · 분산 가시화 클러스터는 한 개의 마스터 프로세스(master process), n개의 슬레이브 프로세스(slave process)로 구성되었다 [12]. 마스터 프로세스는 시간가변 데이터를 처리할 수 있도록 데이터를 분할작업을 수행하고 슬레이브 프로세스들에게 분할된 작업을 할당한다. 슬레이브 프로세스는 분할된 영역에 대해 가시화 작업을 수행 후 결과를 마스터 프로세스에게 전달한다. 마스터 프로세스는 전달받은 결과들을 취합 후 최종적인 결과를 생성한다. 프로세스의 병렬처리를 위해 MPI (message passing interface)를 사용했다 [13].

알고리즘 1은 마스터 노드의 병렬/분산 처리 의사 코드를 나타낸 것이다. 가시화 작업의 요청이 있을 경우, 마스터 노드는 데이터를 분할하고 분할된 영역을 순서대로 슬레이브 노드들에게 할당한다. 마스터 노드는 작업 완료 여부를 슬레이브 노드에게 질의하여 휴식 상태의 슬레이브 노드가 없도록 한다. 만약 휴식 상태의 슬레이브 노드가 있을 경우 다시 영역을 할당한다. 이 작업을 총 시간 간격 n 만큼 수행한다. 슬레이브 노드는 마스터 노드에게 할당받은 영역에 대해서 가시화 알고리즘을 수행하고 마스터 노드에게 전달한다. 전송이 끝나면 휴식상태로 전환되고, 다시 작업이 할당될 때까지 대기하는 과정을 반복한다. 모든 작업이 끝났다는 메세지를 마스터 노드로부터 전달받으면 슬레이브의 작업은 종료된다.

Algorithm 1 Master pseudo code

```
1:  n ← Number of processes
2:  time ← Total time interval
3:  for time t do
4:      Split area by the number of slave nodes
5:      m ← Number of divided areas
6:      for task m do
7:          After inserting it into the work queue, it is passed to the
        slave nodes in sequence
8:      end for
9:      while m ≠ 0 do
10:         Store the results received from slave nodes in memory
11:         m = m − 1
12:     end while
13:     After collecting the received results, save the visualization
    result
14: end for
15: for process n do
16:     Notify slave nodes that the operation is finished
17: end for
18: Generate final visualization video
```

슬레이브 노드는 알고리즘 2의 의사코드를 수행한다. 마스터 노드에게 할당받은 영역에 대해서 가시화 알고리즘을 수행하고 마스터 노드에게 최종적인 결과를 전달한다. 전송이 끝나게 되면 휴식상태로 전환되고, 다음 작업이 할당될 때까지 대기하는 과정을 반복한다. 모든 작업이 끝났다는 메세지를 마스터 노드부터 전달받으면 슬레이브 노드의 프로세스는 종료된다.

Algorithm 2 Slave pseudo code

```
1: Saving information from visualization description files
2: Wait for job order
3: msg ← Command received from master node
4: while msg != Work complete do
5:     Perform visualization algorithm after extracting data
6:     Send results to the master node
7: end while
```

그림 2은 가시화 연산에서 마스터 프로세스가 데이터를 분할하는 방법을 보여준다. 마스터 프로세스가 1개, 슬레이브 프로세스가 4개라고 가정하면 마스터 프로세스는 2D 그리드 (grid)를 슬레이브 프로세스에게 균등하게 분할한다.

Figure 2: Data partition by master process

3.1 태풍 가시화

태풍을 직관적으로 표출하기 위해 마칭큐브 알고리즘을 사용하였다. 마칭큐브 알고리즘은 데이터의 모든 복셀을 순회하면서 삼각형 메쉬구조로 형성되는 폴리곤을 만든다. 주어진 임계값을 각 복셀의 값과 비교하여 임계값이 복셀의 값보다 크면 1로, 작으면 0으로 라벨링(labeling)하는 작업을 거친 후, 복셀의 정점 8개에 대해 8비트 인덱스를 생성한다. 계산된 인덱스는 미리 정의된 테이블 패턴에 맞게 삼각형 메쉬를 정의한다. 모든 경우의 수를 고려할 때, 최초 $2^8 = 256$가지 패턴은 중복을 제외하면 15가지로 줄어든다. 각 삼각형 에지에 대하여 선형 보간법(linear interpolation)으로 정점의 위치와 법선 벡터를 구할 수 있다. 그림 3은 마칭큐브 결과를 보여준다. 왼쪽 그림은 초기 결과 화면이고, 우측 그림은 삼각형 메쉬구조를 볼 수 있게 선으로 표현한 것이다.

마칭큐브 알고리즘은 다른 알고리즘에 비해 연산 속도가 상대적으로 느리기에 GPU를 사용하여 연산속도를 향상하고자 하였고, GPGPU (General-Purpose computing on GPU) [14] 라이브

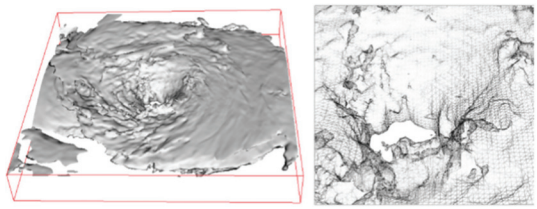


Figure 3: Typhoon data rendering results with marching cube algorithm

러리 중 NVIDIA의 CUDA (Compute Unified Device Architecture) [15]를 사용했다. CUDA는 CPU를 의미하는 호스트(host)와 GPU에 해당하는 디바이스(device)로 구성되어 있다. 전역 메모리(global memory)는 호스트와 디바이스 사이에서 읽기, 쓰기가 가능하고, 전역 메모리보다 속도가 빠르지만 저장 공간은 작은 공유 메모리(shared memory)는 디바이스만 읽고 쓸 수 있다. CUDA 디바이스는 수많은 연산을 동시에 수행할 수 있는 병렬성을 가지고 있기 때문에 마칭 큐브에서 복셀을 순회하고 계산하는 부분을 디바이스에서 처리하도록 구현할 수 있다. 그리고 데이터를 불러오는 것처럼 병렬성이 없는 작업들은 호스트에서 수행되도록 한다. 디바이스에서 실행되는 함수인 커널(kernel)을 호출하기 위해 그리드와 블록(block)의 크기를 적절하게 설정하는 것이 중요하다. 만약 데이터의 크기가 $256 \times 256 \times 256$ 이라면 그리드의 크기를 $32,768 \times 2$, 블록의 크기를 256×1로 하면 전체 스레드는 16,777,216개가 생성되어 동시에 수행된다. 이때 생성된 전체 스레드 개수와 복셀의 수는 같다. 이로 인해 기존 CPU에서 복셀을 순회하면서 계산하는 부분을 디바이스 커널에서 동시에 수행할 수 있으므로 연산 속도가 향상된다.

3.2 광선투사 가시화

광선투사는 3차원 볼륨 데이터 집합으로부터 직접 2D 영상을 계산하는 알고리즘이다 [16]. 이미지 평면의 픽셀을 통과한 광선을 따라 볼륨 데이터를 동일간격으로 선택된 샘플링 위치에서 컬러와 알파 값을 계산하는 기법이다. 전처리 과정 (pre-processing)으로 컬러와 알파 값을 함수로 정의하는데, 주로 가우시안 (gaussian) 함수를 많이 사용한다. 그 이유는 복셀의 평균 값과 표준편차를구하여 전체적인 복셀의 값이 밀집된 영역을 두드러지게 표출하기 위해서이다.

태풍 데이터의 변수 중 기압(pressure) 값을 가시화하였다. 그림 4은 시간에 따른 기압의 변화를 볼륨 렌더링한 결과이다. 기압의 최소값은 -585.77, 최대값은 1870.04 그리고 평균값은 293.84 이므로 변환 함수를 최소값에는 검은색(0.0, 0.0, 0.0), 불투명도 0.0, 평균값에는 녹색(0.0, 1.0, 0.0), 불투명도 0.3 그리고 최대값에는 빨간색(1.0, 0.0,0.0), 불투명도 0.9로 맵핑하였다. 따라서 파란색부분은 기압이 낮은 부분이며 빨간색에 가까울수록 기압이 높은 지역임을 확인할 수 있다.

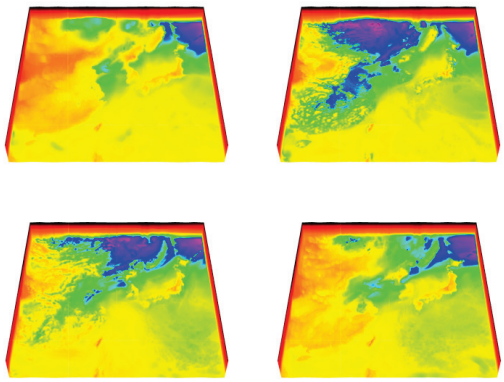


Figure 4: Raycasting of pressure values in typhoon data

4. 가시화 클라이언트 구현

클라이언트에서 사용되는 데이터는 크게 2가지로 나뉜다.

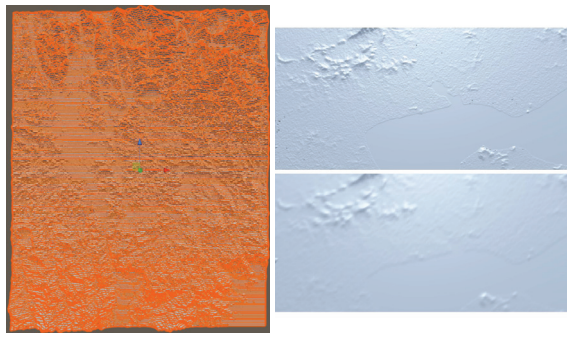
1. 수문 모델 시뮬레이션 데이터 : 임진강 유역의 강수량 데이터를 시뮬레이션한 결과로 38개의 소유역(subbasin)과 하도(channel) 별로 저장

2. 태풍 데이터 : 태풍의 풍속, 수분, 기압 등의 데이터를 포함

수문 데이터는 임진강 유역의 총 25개의 하도에 대한 하도 수치값 추적 결과와 38개의 소유역에 대한 유출량 결과 데이터이다. 각 수문 데이터는 2007년 7월 2일 00시부터 2007년 9월 22일 24시까지의 시간별 수치 값을 가지고 있다. 태풍 데이터는 태풍 장미에 대한 정보로 NetCDF 포맷의 파일로 수십가지의 태풍 정보를 포함하고 있다. 임진강 유역과 태풍 정보를 표현하기 위해 3단계의 기본 도메인(domain) 영역으로 나누었다. 첫번째 도메인은 아시아 영역으로 서버로부터 전송받은 태풍 모델 렌더링 결과를 동영상, 3D 그래픽 모델로 표현한다. 두번째 도메인은 한반도 전체와 일본, 중국 일부를 포함하는 영역이다. 세번째는 임진강의 수문 데이터를 가시화 하는 영역으로 하천 범람 등을 표출한다.

4.1 지형 가시화

임진강 유역의 38개의 소유역은 경기도와 함경남도 일부를 포함하는 한반도 절반 정도를 차지한다. 이 영역을 높은 품질의 지형으로 표출하기 위해 임진강 유역을 포함하는 가장 큰 영역의 지형 데이터를 가시화하였다. 표현된 영역은 위도(N) 37 경도(E) 126에서 위도 40 경도 128까지의 영역으로 위도 경도 기반의 지형을 hgt 데이터 포맷을 사용해 표현하였다. hgt는 NASA와 NGA(National Geospatial-Intelligence Agency)의 SRTM(Shuttle Radar Topography Mission)에서 얻은 표면의 3D 사진과 고도 데이터가 포함된 것이다. 위도 37 $\sim$ 40 경도126 $\sim$ 128까지의 지형은 총 6개의 hgt 파일로 되어있었으며 분석한 결과 각각 가로 3601, 세로 3601 사이즈로 구성되어 있다. 각 좌표마다 높이 정보를 unsigned short 2byte 크기로 저장하고 있음을 알 수 있었다. 또한 리눅스에서 기반의 데이터이므로 윈도우와의 배열 순서가 반대여서 순서를 바꿔줘야 한다. 이 데이터를 Unity3d 엔진에 적용하기 위해서는 오브젝트당 최대로 지원하는 정점 포인트 65000개 이하로 적용해야한다. 이를 위해서 65000 이하로 정점을 분산하여 메쉬(mesh)화를 하였다. 그림 5(a)을 보면 주황색 선으로 표시되어있는 모델은 6개의 hgt 파일 중 하나이며 원본 메쉬를 65000개의 정점을 가진 메쉬로 나눈 것으로 총 212개의 메쉬가 생성되었다 [17]. 이렇게 만들어진 각각의 메쉬는 원본 상태의 크기가 2.0 $\sim$ 2.3GB 크기로 6개의 데이터가 다 모이면 8GB를 초과하여 단일 PC에서 실행하기에는 상당한 부담이 생기게 된다. 이러한 문제를 없애기 위해서 읽어 들이는 데이터에 1씩 주던 interval을 5로 변경하여 해상도와 데이터 크기를 1/25로 줄이게 되었다. 그림 5(b)는 interval 1과 interval 5로 만든 메쉬를 확대한 것으로 해상도에서 차이가 나지만 엔진에서 실시간으로 실행 가능하게 되었다.



(a) A set of subdivided terrain meshes (b) Terrain detail comparison (top: interval 1, bottom: interval 5)

Figure 5: Creating terrains of the Imjin river basin

텍스처의 부재의 문제를 해결하기 위해서 Google Earth를 이용하기로 하였다 [18]. Google Earth는 전세계의 대부분의 텍스처를 가지고 있다. Google Earth Pro를 사용하면 위도, 경도를 기반으로한 탐색이 가능하므로, 지형 데이터에 해당하는 위도 경도를 입력하고 해당하는 텍스처를 다운로드 받아 후처리를 하여 사용하기로 하였다.

지형 데이터들은 서해와 경기도, 북한 부분을 포함하고 있어서 임진강 전역을 표현 가능하다. 하지만 이 상태에서는 소유역 부분만 표현이 가능하므로, 표현 가능한 지역을 한반도 전체와 동아시아와 태평양 부분을 표현해야 한다. 그러기 위해 Google Earth에서 한반도 지역과, 아시아와 태평양 부분의 텍스처를 받아야 한다. Google Earth의 텍스처는 구 형태인 지구에 매핑(mapping)되어 있기 때문에 큰 범위의 텍스처일수록 경계면으로 가면 갈수록 왜곡이 심해질 수밖에 없는 구조이다. 그렇기 때문에 이미지를

겹치게 되면 위도 경도로 이미지를 얻는다 하더라도 이미지끼리 정확하게 매칭이 되지 않는 한계점이 존재하게 된다. 그림 6은 모든 지형 데이터들이 통합된 그림으로 포함된 것이다. 지형 데이터와 그에 해당하는 텍스처, 한반도 주변 텍스처, 아시아와 태평양 지역의 텍스처, 임진강 소유역 메쉬로 3개의 도메인으로 나뉘어져 있다.

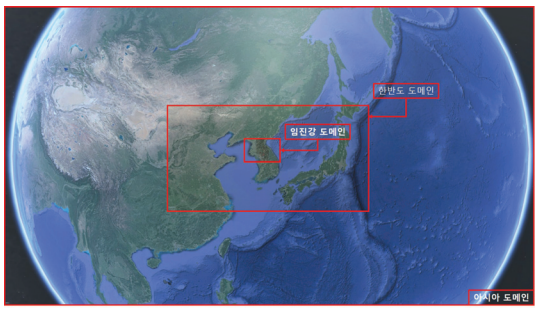


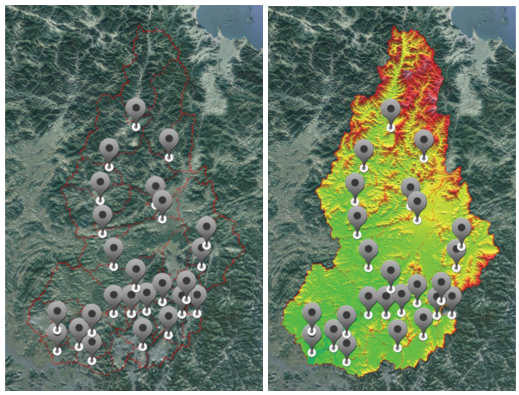
Figure 6: Terrain data integration

4.2 소유역 가시화

임진강 유역은 38개의 소유역과 25개의 하도로 구성되었다. 각각의 소유역에 해당하는 수문 데이터를 가시화 하기 위해서는 각각의 소유역의 영역을 하나의 메쉬로 만들어야한다. 이를 각 패치(patch)단위로 구성하여 가시화하였다. 패치는 다음과 같이 구현하였다. 제공받은 ArcGIS 기반의 소유역 외곽을 저장하고 있는 파일을 클라이언트에서 사용할 수 있도록 변환해주었다. ArcGIS 형식의 좌표를 전처리 과정을 통해서 위도, 경도로 변환하여 CSV(Comma Seperation Value) 형식으로 변환하였다. 다음으로 클라이언트에서 CSV 파일을 읽어 각 소유역의 외곽 정보를 내부적으로 배열에 저장하고 Ear clipping 알고리즘을 적용하여 소유역을 하나의 메쉬로 만들어준다 [19]. 이렇게 생성된 메쉬는 고도 정보가 있지 않는 평평한 형태이다.

고품질의 소유역 가시화를 위해 4.1에서 생성한 지형 데이터의 고도 정보를 패치에 적용할 필요가 있다. 개선된 패치 생성 알고리즘을 사용하여 소유역 정보에 맵핑되는 지형 데이터가 적용되게 하였다. 각 소유역의 패치가 포함하는 지형 데이터의 정보를 추출한다. 추출된 데이터들에 대하여 전체를 순회하면서 삼각형 폴리곤을 생성하여 지형의 고도 정보가 포함된 패치를 제작하였다. 그리고 임진강 소유역의 패치를 구분하기 위해 앞에서 구했던 소유역의 외곽 정보를 Unity3d의 LineRenderer 객체를 사용하여 경계선을 표현했다. 또한 임진강 지역은 25개의 하도(channel)로 구성되었다. 하도의 위치 값은 관련 기관에서 제공받은 정보를 이용하여 임진강 지형 데이터에 맵핑하였다. 임진강 지형에 하도를 매핑하고 2D 텍스처 마커로 하도를 나타내었다.

임진강 유역의 패치에 해당하는 부분에 각 지역의 높낮이를 보다 명확하게 볼 수 있도록 등고(contour) 텍스처를 구현하였다. 일반 텍스처를 입혔을 때보다 한눈에 어느 지역이 낮고 높은지 확연하게 볼 수 있으며 이를 통해 강수량에 따른 저지대의 피해를 더욱 쉽게 예상해 볼 수 있다 지형 데이터에 저장되어 있는 정보들을 모두 읽어 가장 최대 높이 값(189.7)과 최소 높이 값(-6.6)을 추출하였다. 이를 통해 모든 높이 값들을 0.0 ~ 1.0 사이의 값으로 정규화 한 후, 각 값에서 1.0에 해당하는 값은 흰색, 0.5에 해당하는 값은 빨강색, 0.25에 해당하는 값은 노란색, 0.05에 해당하는 값은 초록색, 0.0에 해당하는 값은 파랑색으로 할당하였고 사이의 값들은 보간하여 구현하였다. 그림 7은 소유역 가시화의 결과이다.



(a) patch, channel, boundary visualization result

(b) contour texture

Figure 7: subwatershed visualization result

4.3 GUI 구현

사용자가 원하는 날짜를 선택하고 데이터 타입을 선택할 수 있도록 하기 위한 데이터 전송 GUI와 각종 기능들을 실행하기 위한 GUI를 구현하였다. 클라이언트에서 사용자가 원하는 결과를 보기위해서는 다음과 같은 절차를 걸쳐야 한다.

1. 클라이언트 날짜 GUI에서 사용자는 원하는 데이터의 날짜(기간)을 선택하고 Send 버튼을 눌러 서버로 날짜를 전송(그림 8(a) 참조)

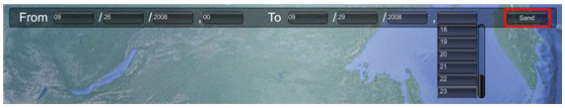
2. 서버는 클라이언트에서 전송 받은 날짜 범위가 데이터 안에 존재하는지 검색후 존재하는 데이터 날짜를 클라이언트로 전송

3. 클라이언트 Menu GUI에서 Data List 버튼을 누르면 가능한 데이터 타입, 렌더링 타입과 날짜가 활성화(그림 8(b) 참조)

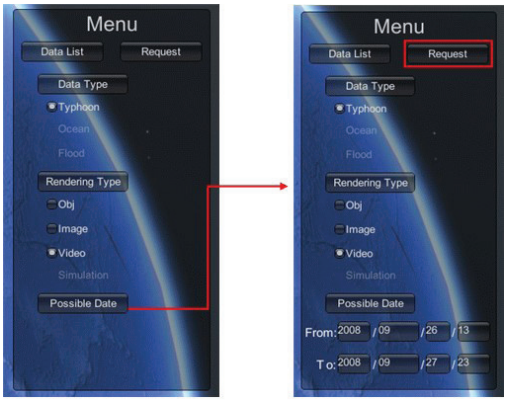
4. 가능한 기간을 재선택 후 Request 버튼을 눌러 최종적으로 렌더링된 결과를 서버에 요청하고 결과를 전송 받기까지 기다림(그림 8(b) 참조)

5. 전송이 완료되면 렌더링 타입에 맞게 가시화

클라이언트가 원하는 날짜 정보를 전송하게 되면 서버로부터 해당 날짜사이의 데이터 생성이 가능한 날짜가 다시 전송된다. 또한, 날짜 사이에 생성이 가능한 렌더링 타입과 데이터 타입이 활성화 된다. Menu GUI의 경우 Data List 버튼을 클릭함에 따라 펼쳐보기와 숨겨보기의 기능으로 구현하였다. 모든 데이터 항목을 선택 한 후, Request 버튼을 클릭하면 클라이언트는 서버로 원하는 데이터 렌더링과 타입에 대한 정보를 전송하게 된다. Data Type은 태풍, 해양, 홍수와 같은 데이터의 종류를 나타내고 Rendering Type은 어떠한 렌더링 형태의 데이터를 받아볼 것인지 선택한다. Possible Date는 서버로부터 전송된 데이터 생성 가능 날짜사이에서 사용자가 원하는 날짜를 재설정 할 수 있다.



(a) Menus for date request frome server



(b) selecting candidate data stored in server

Figure 8: Client GUI for setting time range

클라이언트의 GUI는 각 도메인 위치에 있을때마다 활성화 되는 GUI가 다르다. 서버로 데이터를 요청하는 GUI는 모든 도메인에서 실행가능하지만 등고선을 보는 contour GUI와 카메라의 위치를 자유롭게 설정할 수 있는 simulator GUI의 경우 임진강 도메인에서만 실행이 가능하도록 하였다. 모든 GUI는 Unity3d에서 지원되는 GUI 기능을 이용하여 스크립트를 통해 직접 구현하였다. 각 GUI 박스들은 마우스 오버 이벤트가 발생하면 투명도를 높혀 잘 보일 수 있도록 구현하였다.

5. 통합 가시화

그림 9은 클라이언트의 처음 시작화면이다. 이전에 소개했던 클라이언트의 3가지의 도메인의 지형과 임진강 유역의 소유역의 패치와 하도를 표현하는 마커 등과 사용자를 위한 GUI가 구현되어 있다. 서버와 통신할 통신 모듈, 시스템을 컨트롤하는 모듈과 병렬 가시화 클러스터에서 렌더링된 이미지와 3D 그래픽 모델을 플레이하는 모듈 등이 포함되어 있다.

Figure 9: Initial scene for Imjin River visualization

5.1 임진강 수문 데이터 가시화

본 장에서는 임진강 유역의 채널에 대한 수문 데이터 가시화를 소개한다. 직관적인 가시화를 위해 시간 간격별 수문 수치 값을 2D 선 그래프로 나타내었다. 사용된 데이터는 총 25개의 하도에 대한 하도 수치 값 추적 결과와 38개의 소유역에 대한 강수 유출량 결과 데이터이다. 그림 10는 소유역과 하도의 가시화 결과 이미지로 그래프는 하나의 임진강 소유역 데이터에 대한 가시화 결과를 나타낸다. 사용자의 이벤트에 따라 해당하는 시간대의 수치 값을 확인하거나, 소유역 혹은 하도 포인트를 클릭하거나 'S' 키를 입력하면 자동으로 시간대별로 수치 값을 확인할 수 있도록 설계되었다. 기본적으로 상단에는 수문 데이터의 파일 이름, x 축은 시간, y축은 수치 값 그리고 최대 수치 값을 보여준다.

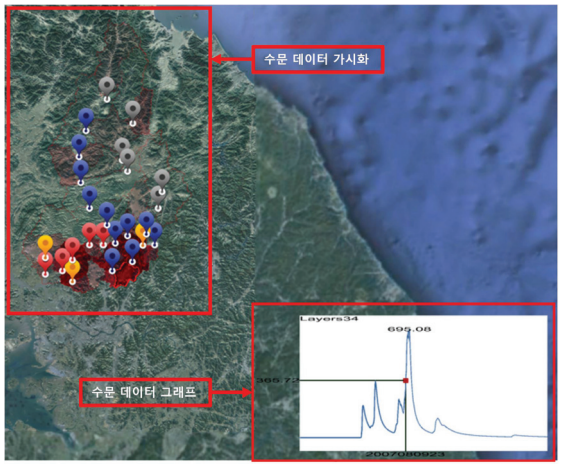


Figure 10: Hydrological data visualization of Imjin River

하도 추적결과는 포인트 색의 변화로, 소유역에 대한 유출량 결과 데이터는 패치의 색의 변화로 표현하였다. 수문 데이터 값이 점차 높은 수치를 나타냄에 따라 짙은 빨간색으로 적용하여 시간의 변화에 따른 하도와 소유역의 데이터를 가시화하였다(그림 10의 수문데이터 가시화 영역 참조). 사용자는 시뮬레이션 결과를 통해 유역의 수문 값을 확인할 수 있고, 어떤 유역이 어느 시점에 위험한지를 직관적으로 확인할 수 있다.

5.2 태풍 데이터 가시화

그림 1의 병렬 가시화 클러스터의 렌더링 결과는 이미지 혹은 3D 그래픽 모델 2가지이다. 클라이언트에서 렌더링 타입으로 Video를 선택하고 가시화를 요청하였을 경우 연속된 이미지를 클라이언트로 전송하게 된다. 그러므로 클라이언트는 전송 받은 렌더링 결과를 영상으로 플레이 할 수 있는 동영상 플레이어 기능이 필수적으로 필요하다. 통신으로 전달 받은 이미지는 바이너리 형태의 데이터로 이 데이터를 Unity3d의 클래스 중 하나인 Texture2D 객체 배열로 읽어 들여서 저장한다. Unity3d는 메인 스레드가 아닌 외부 스레드를 사용하는 경우는 Unity3d 클래스를 사용할 수 없으므로 통신 스레드에서 이미지 파일을 디스크에 저장한 뒤 메인 스레드에서 다시 읽어오는 작업을 해야한다. 디스크에서 읽어온 이미지를 메인 스레드에서 Texture2D 객체로 생성하고 Plane 오브젝트에 동영상 프레임 속도만큼 texture를 변경해 주면서 동영상을 플레이한다. 그림 11은 Texture2D로 만든 동영상 플레이어로 각 프레임마다 이미지를 캡처한 것이다.

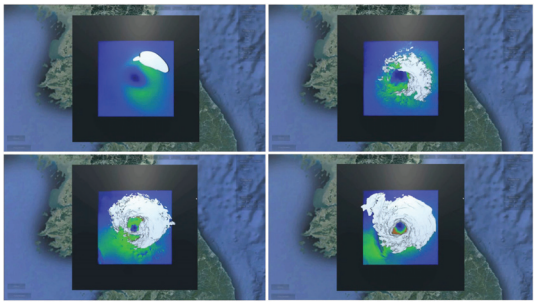


Figure 11: Time-varaying typhoon data visualization

그림 12은 렌더링 결과를 3D 그래픽 모델로 요청한 경우의 결과이다. 가시화 클러스터는 마칭큐브를 사용하여 스칼라 데이터에서 3D 모델을 폴리곤화하고 이를 서버를 통해 클라이언트로 전송하게 된다. 클라이언트는 전송 받은 3D 그래픽 모델을 로드하여 이미지와 비슷한 과정을 거친 후 정해진 프레임마다 플레이하여 애니메이션 효과를 낸다.

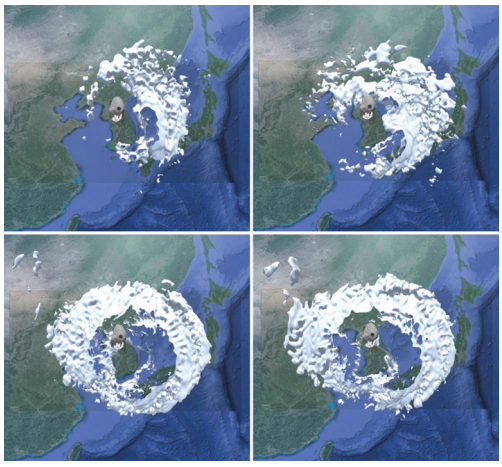


Figure 12: Visualization of the typhoon model

6. 실험 결과

이번 장에서는 시스템에서 사용한 데이터에 대해 가시화 알고리즘을 적용한 실험 결과를 통해 본 논문에서 제안하는 시스템의 특성과 성능을 설명한다. 제안된 병렬 가시화 클러스터는 C++ 언어로 구현되었고, 병렬처리를 위해 MPI, CUDA, 그리고 렌더링을 위해 OpenGL 과 VTK가 사용되었다. 실험은 표 3에 명시된 환경에서 수행되었으며 병렬/분산 가시화 실험(6.1)에서는 마스터 노드로 사용되었고, GPU 병렬 프로그래밍(6.2)에서는 단독으로 사용되었다.

Table 3: Experimental environment

CPU	Intel® Core™ i7 3.40 GHz
Memory	16.0 GB
VGA	NVIDIA GeForce GTX 770

6.1 병렬/분산 가시화 실험 결과

병렬/분산 가시화 수행시간을 실험하였다. 알고리즘은 마칭큐브, 광선 추적법 두 가지를 사용하였으며, 알고리즘에 대해 가시화를 수행하는 시간을 측정하였다. 총 시간 간격이 34개인 태풍 장미 데이터의 온도에 대해 실험을 실시하였다. 보안 문제로 인한 실험 환경 접근의 제약으로 고성능 컴퓨터의 클러스터를 사용하지 못하고 보유하고있는 4개의 노드를 사용하였다. 각 노드의 성능은 표 4와 같다.

표 5와 같이 병렬적으로 하나의 슬레이브 노드가 한 시간 간격의 데이터를 읽고 작업을 처리하는 시간은 평균 1.48초가 소요되었으며, 작업을 분산 및 취합하는 마스터 노드의 수행 시간은 약 3.28초가 소요되었다. 표 6는 전체 시간간격에 대한 가시화 작업 처리 시간을 실험한 결과이다. 가시화 알고리즘은 마칭큐브, 광

Table 4: Slave node environment

CPU	Intel® Core™ 2 Duo E8500 3.16GHz
Memory	6.0 GB
VGA	NVIDIA GeForce 9600GT

선 추적법을 사용한다. 슬레이브 노드가 하나일 때는 전체 작업 시간은 20.54초, 두 개일 때는 18.01초 그리고 네 개일 때는 13.74초가 소요된다.

Table 5: Master, slave node task processing time

Node	**Master node**	**Slave node**
Time(sec)	3.28	1.48

Table 6: Job processing time based on number of nodes

Node	**1**	**2**	**3**	**4**
Time(sec)	20.54	18.01	15.37	13.74

6.2 GPU 병렬 프로그래밍 실험 결과

표 7는 시간간격 하나에 해당하는 볼륨 데이터에 대한 마칭큐브 실험을 수행했을 때, CPU와 GPU의 연산 속도를 비교한 것이다. CPU는 순차적인 방법으로 모든 복셀을 순회하면서 다각형을 생성하였고, GPU는 각 복셀에 대한 연산을 하나의 스레드 커널 함수로 정의하여 동시에 수행되도록 설계하였다. 실험에 사용된 데이터 해상도는 $300 \times 300 \times 40$, 크기는 약 13.7 MB 이다. GPU 그리드를 14,062개, 블록을 256개 그리고 각 블록당 스레드를 256개 사용하였다. 결국 동시에 3,599,872개의 스레드가 실행하도록 설계하고 실험을 진행하였다. 그 결과 임계값이 200일때는 GPU가 CPU에 비해 2배정도, 120일때는 3배정도, 그리고 40일때는 5배이상 속도가 향상되었다.

Table 7: Time performance of marching cube (ms)

threshold	**CPU**	**GPU**
Iso 40	312	62.61
Iso 120	218	81.25
Iso 200	116	52.88

표 8는 광선 투사법을 GPU 병렬 프로그래밍으로 구현한 결과를 나타낸다. 화면의 해상도에 따라 실험을 진행하였으며, 해상도가 512×512일때 그리드를 16×16, 블록을 32×32로 분할하였다. 해상도의 한 픽셀마다 스레드가 수행되도록 하였으며 스레드의 커널 함수는 하나의 광선을 출발시켜 색과 불투명도를 계산하는 연산을 수행한다. 볼륨 데이터 및 변환 함수를 GPU 텍스처 메모리에 저장해서 데이터에 빠르게 접근할 수 있도록 설계하였다. 그 결과 화면 해상도가 커질수록 초당 프레임수 (fps)는 줄어들지만, 30fps보다 훨씬 수치가 크므로 실시간 렌더링이 가능하다. 1024×1024의 해상도의 경우 61.02fps의 속도를 얻을 수 있었다.

Table 8: Raycasting rendering speed (fps)

Resolution	**FPS**
256^2	74.30
512^2	68.56
1024^2	61.02

7. 결론 및 향후 연구

본 논문은 기존의 과학적 가시화 기법을 통해 시간가변 데이터를 정확히 분석하고 가시화함으로써, 자연재난 상황에서 긴급조치 사항들을 결정해야하는 분야에서 자연재난 데이터를 효과적으로 활용할 수 있는 서버-클라이언트 시스템을 개발하였다. 개발된 시스템의 병렬 가시화 클러스터는 저장되어 있는 태풍, 해일, 홍수, 범람과 같은 재난재해 데이터와 관측/예측된 데이터를 효과적으로 가시화하기 위해 필수적으로 요구되는 병렬/분산 렌더링을 수행한다. 그리고 GPU 병렬 프로그래밍을 통해 가시화 알고리즘의 수행 속도를 향상하였으며 데이터의 매개변수 등을 읽어들여 동영상과 3D 그래픽 모델 파일로 가시화 결과를 생성하는 작업을 수행한다. 이렇게 생성된 결과물은 서버를 통해 가시화 클라이언트로 전송되게 되고 이로인해 저가의 범용 PC나 휴대용 시스템 상에서도 높은 품질의 렌더링 결과를 확인할 수 있게 되었다. 사용자는 가시화 클라이언트를 통해 원하는 기간을 지정하여 가시화된 결과를 대화식으로 전달받을 수 있으며 임진강 유역의 사실적인 지형 표현과 소유역, 하도의 시뮬레이션을 통해 침수 지역에 대한 대피나 피해를 예상할 수 있게 되었다.

향후 연구로 다양한 가시화 알고리즘을 적용하는 시스템으로 확장하여 직관적인 가시화 영상을 표출하고자 한다. 그리고 시스템의 여유 메모리, 가용 CPU, GPU 자원을 고려한 프로세서별 최적의 부하균형을 만족하는 병렬/분산 알고리즘을 설계하여 가시화 서버의 성능을 향상시키고자 한다. 또한 개발된 클라이언트의 임진강 유역의 지형의 더욱 사실적인 가시화를 위한 맞춤형 쉐이더를 추가적으로 적용하고 기존에 적용되어 있는 텍스처의 해상도를 높이기 위해 Google Earth의 위도 경도의 범위를 더욱 축소시켜 고해상도의 이미지를 추출하고 적용할 것이다. 또한 사용자 편의성을 위해 GUI의 기능을 향상시키고자 한다.

감사의 글

본 연구는 한국과학기술정보연구원(KISTI)의 위탁연구 과제로 수행한 것임. 이 논문은 2017년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2016R1D1A1B03931641).

References

[1] J. Kruger and R. Westermann, "Acceleration techniques for gpu-based volume rendering," in *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*. IEEE Computer Society, 2003, p. 38.

[2] B. Wylie, C. Pavlakos, V. Lewis, and K. Moreland, "Scalable rendering on pc clusters," *IEEE Computer Graphics and Applications*, vol. 21, no. 4, pp. 62–69, 2001.

[3] W. Wong and J. Lee, *Statistical analysis of geographic information with ArcView GIS and ArcGIS*. Wiley, 2005.

[4] 정석철, 정서원, 김종용, and 박상훈, "지구환경 데이터를 위한 멀티플랫폼 가시화 시스템," *컴퓨터그래픽스학회논문지*, vol. 21, no. 3, pp. 36–45, 2015.

[5] J. Bayer, O. Flege, P. Knauber, R. Laqua, D. Muthig, K. Schmid, T. Widen, and J.-M. DeBaud, "Pulse: A methodology to develop software product lines," in *Proceedings of the 1999 symposium on Software reusability*. ACM, 1999, pp. 122–131.

[6] S. L. Olivier, A. K. Porterfield, K. B. Wheeler, and J. F. Prins, "Scheduling task parallelism on multi-socket multicore systems," in *Proceedings of the 1st International Workshop on Runtime and Operating Systems for Supercomputers*. ACM, 2011, pp. 49–56.

[7] NetCDF, http://www.unidata.ucar.edu/, 2015.

[8] VTK, http://www.vtk.org/, 2016.

[9] Unity3d, https://unity3d.com/, 2018.

[10] C. Ware, "Color sequences for univariate maps: Theory, experiments and principles," *IEEE Computer Graphics and Applications*, vol. 8, no. 5, pp. 41–49, 1988.

[11] K. Nishihashi, T. Higaki, K. Okabe, B. Raytchev, T. Tamaki, and K. Kaneda, "Volume rendering using grid computing for large-scale volume data," *International Journal of CAD/CAM*, vol. 9, no. 1, pp. 111–120, 2009.

[12] X. Zhou, R. T. Collins, T. Kanade, and P. Metes, "A master-slave system to acquire biometric imagery of humans at distance," in *First ACM SIGMM international workshop on Video surveillance*. ACM, 2003, pp. 113–120.

[13] W. Gropp, R. Thakur, and E. Lusk, *Using MPI-2: Advanced features of the message passing interface*. MIT press, 1999.

[14] S. Lee, S.-J. Min, and R. Eigenmann, "Openmp to gpgpu: a compiler framework for automatic translation and optimization," *ACM Sigplan Notices*, vol. 44, no. 4, pp. 101–110, 2009.

[15] CUDA, https://developer.nvidia.com/cuda-zone/, 2015.

[16] M. Levoy, "Efficient ray tracing of volume data," *ACM Transactions on Graphics (TOG)*, vol. 9, no. 3, pp. 245–261, 1990.

[17] H. Edelsbrunner, *Geometry and topology for mesh generation*. Cambridge University Press, 2001, vol. 7.

[18] N. Gorelick, "Google earth engine," in *AGU Fall Meeting Abstracts*, 2012.

[19] D. Eberly, "Triangulation by ear clipping," *Geometric Tools*, 2008.

〈저자소개〉

김 종 용
- 2014 청운대학교 컴퓨터학과 학사
- 2016 동국대학교 멀티미디어학과 석사
- 2016 ~ 현재 동국대학교 멀티미디어학과 박사과정
- 관심분야 : 혼합현실, 가상현실, 컴퓨터 그래픽스

정 석 철
- 2014 동국대학교 멀티미디어공학과 학사
- 2016 동국대학교 멀티미디어학과 석사
- 관심분야: 컴퓨터 그래픽스, 병렬/분산 가시화

이 계 원
- 2018 동국대학교 멀티미디어공학과 학사
- 관심분야: 영상처리, 컴퓨터 그래픽스, 네트워크

조 준 영
- 2012년 ~ 현재 동국대학교 멀티미디어공학과 학사과정
- 관심분야: 정보보안, 객체지향 프로그래밍

김 동 욱
- 2017 동국대학교 멀티미디어공학과 학사

2017 ~ 현재 동국대학교 멀티미디어공학과 석사 과정

관심분야 : 딥러닝, 컴퓨터 비전, 컴퓨터 그래픽스

박 상 훈
- 1993 서강대학교 수학과 학사
- 1995 서강대학교 컴퓨터학과 석사
- 2000 서강대학교 컴퓨터학과 박사
- 2002 ~ 2005 대구카톨릭대학교 컴퓨터정보통신공학부조교수
- 2001 University of California, Davis 방문 연구원
- 2005 ~ 현재 동국대학교 멀티미디어학과 교수
- 관심분야: 실시간 렌더링, 사실적 렌더링, 과학적 가시화, 고성능 컴퓨팅 등