거칠기맵과 편향맵을 이용한 지형 렌더링 기법

이은석[O] 조인우 신병석

인하대학교 컴퓨터·정보 공학과

{elflee77, joins}@inha.edu, bsshin@inha.ac.kr

A Terrain Rendering Method using Roughness Map and Bias Map

Eun-Seok Lee[O] In-Woo Jo Byeong-Seok Shin

Dept. of Computer Science and Information Engineering, Inha University

요 약

최근의 지형시각화 연구에서는 대용량 데이터를 실시간에 처리하기 위하여 여러 가지 상세단계조절 기법을 사용한다. 하지만 상세단계조절을 통한 메쉬 간략화 과정에서 발생하는 기하오차 때문에 연속된 프레임에서 기하 파핑 현상이 일어난다. 본 논문에서는 거칠기맵과 편향맵을 이용하여 기하파핑 현상을 효과적으로 줄일 수 있는 방법을 제안한다. 거칠기맵과 편향맵은 지형 메쉬를 구성하는 정점이 적은 기하오차를 가지는 위치로 이동시켜주는 역할을 한다. 거칠기맵과 편향맵은 텍스쳐로 저장되기 때문에 GPU에서 사용하기 적합하다. 또한 편향맵을 이용한 정점 이동 연산은 GPU에서 수행되므로 병렬처리를 통한 빠른 시각화가 가능하다.

Abstract

In recent researches, several LOD techniques are used for real-time visualization of large sized terrain data. However, during mesh simplification, geometry popping may occur in consecutive frames, because of the geometric error. We propose an efficient method for reducing the geometry popping using roughness map and bias map. A roughness map and a bias map are used to move vertices of the terrain mesh to appropriate position where they minimize the geometry errors. A roughness map and a bias map are represented as a texture suitable for GPU processing. Moving vertices using bias map is processed on the GPU, so the high-speed visualization can be possible.

키워드: 지형 렌더링, 실시간 렌더링, 연속상세단계, 변위 매핑

Keywords: terrain rendering, realtime rendering, LOD, displacement mapping

투고일: 2010-10-25 / 심사일: 2차 2010-12-08 / 게재확정일: 2010-12-13

1. 서론

고도 데이터를 통한 지형 시각화는 컴퓨터 게임과 비행 시뮬레이션 등의 다양한 응용프로그램들에서 사용된다. 일반적으로 이러한 응용프로그램들은 CPU에서 실시간에 처리할 수 없는 대용량의 지형 데이터를 높은 프레임율로 렌더링해야 한다. 그러나 대용량의 지형 데이터들은 다량의 다각형들로 표현되므로 이를 시각화하려면 많은 양의 메모리가 필요하며 실시간에 처리하기 어렵다. 이러한 문제점을 해결하기 위해서 많은 연속상세단계(continuous level-of-detail) 기법과 외부 메모리 처리기법(out-of-core)들이 연구되어 왔다.

데이터를 외부 메모리에서 처리하면 많은 시간이 소요된다. 따라서 외부 메모리 처리를 통한 실시간 시각화를 위해서는 기존에 주 메모리(in-core)에서 처리되던 연속상세단계 조절 기법들에 비해 간단한 연산이 필요하다. 대표적인 외부 메모리 처리기법인 기하 클립맵(geometry clipmap)[1]에서도 연산량을 줄이기 위해 단순한 거리기반의 상세단계조절 기법을 이용한다.

하지만 거리기반의 상세단계조절기법들[2,3,4]이나 기하오차의 허용범위를 확대하는 방법[3]들은 시점이 변경되었을 때 기하파핑 현상을 일으킬 수 있다. 본 논문에서는 이러한 기하오차를 줄여 기하파핑 현상을 줄일 수 있는 거칠기맵(roughness map)과 편향맵(bias map)을 제안한다. 이 텍스쳐들은 순차적으로 전처리과정에서 생성되며 편향맵은 렌더링과정에서 DEM 데이터 대신 사용된다. 편향맵은 GPU에서 참조하기 용이하도록 텍스쳐형태로 저장 되며 간략화된 지형 메쉬를 구성하는 정점들이 기하파핑을 최소화하는 위치로 이동하도록 하는 벡터정보를 저장하고 있다. 렌더링 과정에서는 GPU 메모리에 업로드된 편향맵의 벡터값을 이용하여 정점을 이동시킨다. 따라서 기존의 정규격자를 이용한 방법들[1,4,5,6,7,8]보다 기하파핑이 줄어든다.

2장에선 지형 데이터를 시각화 하는데 사용된 기존의 연속 상세단계기법들에 대해 소개하고 3장에서는 거칠기맵과 편향맵에 대한 방법을 상세히 기술한다. 4장에서는 기존의 방법과 비교한 결과를 보이고 5장에서 결론을 맺는다.

2. 관련연구

기존 상세단계조절 기법은 주로 CPU에서 계층 구조를 사용하여 수행되었다. 대표적인 예로 사진트리를 이용한 방법과 이진트리를 이용한 방법이 있다. Lindstrom이 제안한 사진트리 기반 방법은 지형 데이터를 4개의 균일한 영역으로 재귀적으로 분할함으로써 연속상세단계선택과 시각 절두체 선별을 수행할 수 있게 하였다[5]. Röttger는 사진트리기법에서 하향식방식을 사용하여 간결한 크랙 제거와 기하 모핑을 할 수 있는 방법을 제안하였다[6]. Duchaineau는 지형 시각화를 위해 삼각형들의 이진트리를 이용하는 ROAM(real-time optimally adaptive meshes)을 제안했다[7]. 이 방법들은 계층구조를 사용하여 시점에 따른 최적의 메쉬를 선택할 수 있게 한다. 하지만 계층구조는 포인터연산과 재귀연산 때문에 CPU에서만 처리할 수 있고 고속 병렬처리가 가능한 GPU에서는 활용할 수 없다.

GPU가 사용되면서 연속상세단계를 이용한 지형 렌더링 방법들은 기존의 방법보다 더 큰 데이터들을 빠르게 시각화할 수 있게 되었다. 그러나 주 메모리에서 비디오 메모리로 매 프레임 새로 생성된 메쉬 데이터를 업로드 하는데 병목이 발생하였다. 이러한 병목을 줄여주기 위하여 Ulrich는 전처리과정에서 지형의 기하정보를 미리 생성하여 사진트리형태의 묶음으로 비디오메모리에 업로드하여 캐쉬(cache)형태로 사용하는 묶음 상세단계(chunked LOD) 기법을 제안하였다[8]. 이는 CPU에서 비디오메모리로 적재하는 과정의 병목을 효율적으로 줄여주었다. 하지만 기존의 연속상세단계 기법들에 비하여 묶음 상세단계 기법은 전처리 단계에서 미리 기하구조를 만들어 놓기 때문에 시점이 변화하는 경우 정확한 상세단계를 선택하기 힘들다.

최근에는 GPU를 이용한 다양한 지형 시각화 기법들이 소개되고 있다. Dachsbacher이 제안한 기하 이미지 와핑(warping)은 중요한 영역의 기하정보 주파수를 높여 준다[9]. 이때 중요한 영역을 결정하기 위해서 중요성 맵(importance map)을 사용하는데[10], 중요성은 시각절두체 선별과 시점과의 거리 그리고 표면의 방향 및 차폐 여부에 따라 결정된다. Losasso가 제안한 기하 클립맵[1]은 대용량 텍스쳐의 밉맵(mip-map)을 피라미드 구조로 생성하여 처리하는 클립맵[2]을 대용량 지형 데이터에 적용하여 외부 메모리에서 효과적으로 처리 할 수 있도록 하였다. Livny는 이를 개선하여 시야 평면에서 정규격자를 지형평면으로 투영시켜서 지형 메쉬를 생성하는 PGM(persistant grid mapping)을 제안하였다[3]. 또한 Livny는 지형 메쉬를 생성할 때 상세단계 차이로 인한 크랙을 미리 비디오 메모리에 정의해 놓은 패치를 이용하여 GPU에서 제거하는 방법을 제안하였다[4]. 하지만 이 방법들은 거리기반의 상세단계 선택을 하기 때문에 지형의 거칠기를 반영할 수 없었으며, 이로 인해 관측조건이 바뀔 때 파핑이 발생한다.

3. 거칠기맵/편향맵을 이용한 지형 렌더링

기존의 연속상세단계 기법들에서는 지형 메쉬를 간략화하는 과정에서 기하오차가 발생하고 이로 인해 기하파핑이 발생한다. 여기서는 기하파핑 현상을 효과적으로 줄이기위해

거칠기맵과 편향맵을 이용한 정점 이동기법을 제안한다. 그림 1은 제안하는 방법의 처리절차를 보여준다.

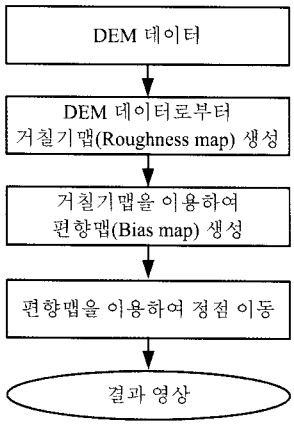


그림1 : 본 논문에서 제안하는 방법의 처리절차

거칠기맵은 지형의 거칠기를 저장한 데이터이고 편향맵은 정점이 이동할 방향과 해당 샘플의 높이값을 저장한 데이터이다. 이 데이터들은 모두 전처리 과정에서 GPU가 쉽게 처리할 수 있도록 텍스쳐 형태로 만들어진다. 렌더링 과정에서는 GPU의 렌더링 파이프라인에 편향맵이 업로드 되어 지형 격자를 구성하는 정점들을 이동하는데 참조된다.

3.1 거칠기맵 (Roughness Map)

지형의 거칠기 계산은 영상처리에서 널리 사용하는 가우시안 라플라시안 필터(Laplacian of Gaussian filter : LoG)[11]를 사용했다.

라플라시안 필터(Laplacian filter)는 한 픽셀과 주변 픽셀과의 색상차를 이용하여 윤곽선 추출을 하는데 사용이 된다. 그림2의 왼쪽은 수직/수평 성분만 고려한 라플라시안 마스크(식1 참조)이고 오른쪽은 대각선 성분까지 고려한 라플라시안 마스크이다(식2 참조). 이 라플라시안 마스크에 색상값 대신 높이값을 적용하면 높이 차이가 클수록 큰 값을 갖는 거칠기맵을 생성할 수 있다 (그림3 참조).

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

그림2 : 4방향과 8방향에 대한 라플라시안 마스크

$$\nabla^2 f = [f(x+1,y)+f(x,y+1) + f(x-1,y)+f(x,y-1) - 4f(x,y)] \quad \text{(식1)}$$

$$\nabla^2 f = [f(x+1,y)+f(x,y+1) + f(x+1,y+1)+f(x-1,y) + f(x,y-1)+f(x-1,y-1) + f(x-1,y+1)+f(x+1,y-1) - 8f(x,y)] \quad \text{(식2)}$$

가우시안 필터(Gaussian Filter)는 주변의 값들에 가우스 곡선을 이용하여 가중치를 주고 그 값들의 평균값을 구하여 블러 효과(blur effect)를 내는데 사용 된다. 이 필터의 마스크 값을 이용하면 각 샘플점에서의 거리기반 가중치를 쉽게 구할 수 있다.

가우시안 라플라시안 필터는 가우시안 형태를 취한 라플라시안 필터로 이것을 지형 데이터에 적용하면 샘플점과 비교할 주변점들의 높이값 차이(거칠기값)과 거리에 대한 가중치를 동시에 표현할 수 있다. 따라서 라플라시안 필터만 적용한 경우에는 가까운 거리에서 높이값이 변화하는 부분만 거칠기 값을 구할 수 있는 반면 가우시안 라플라시안 필터를 적용한 경우엔 넓은 범위에서 가중치를 두어 주변값들의 높이값의 변화를 보다 효과적으로 구할 수 있다 (그림4 참조). 식3은 본 논문에서 사용하는 가우시안 라플라시안 필터이다. σ는 가우시안의 표준편차이다. 본 논문에서는 마스크값의 절반을 표준편차로 사용하였다.

$$\mathrm{LoG}(x,y) = -\left[\frac{(x^2+y^2)-\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \text{(식3)}$$

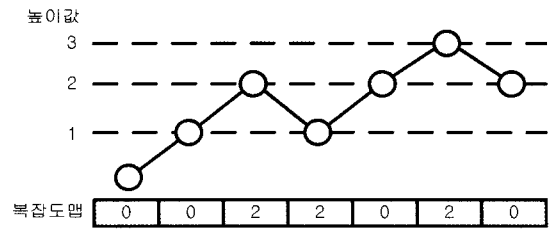


그림3 : 라플라시안 필터만 이용한 경우의 지형 거칠기

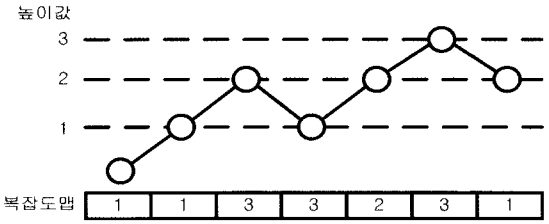


그림4 : 가우시안 라플라시안 필터를 이용한 경우의 지형 거칠기

3.2 편향맵 (Bias Map)

편향맵은 거칠기 정보를 이용하여 지형 메쉬의 각 정점들이 기하오차를 최소화하는 위치로 이동할 수 있도록 하는 벡터값을 저장하는 24비트 텍스쳐다. 편향맵의 각 샘플점에서 거칠기맵과 동일한 마스크의 영역 안에 있는 텍셀값을 모두 읽어 온다. 그리고 그림5와 같이 마스크의 중앙에서부터 0이 아닌 값을 가지는 텍셀들을 향한 이차원 벡터들을 생성한다. 이때 거칠기값을 이 벡터의 가중치로 이용한다. 가중치가 부여된 벡터들을 모두 합하여 평균을 내면 최종적으로 식4와 같이 이동할 방향벡터 V_{bias}가 생성된다. 여기서 V는 마스크 중앙에서 샘플점을 향한 벡터이고 R은 각 샘플점의 거칠기값이다. 거칠기가 0이면 지형이 평평함을 의미하기 때문에 연산에서 제외한다. 그리고 0이 아닌 값을 갖는 텍셀을 이용하여 중앙점을 기준으로 벡터를 생성한다. 그림5에서 이 벡터들은 각각 7, 3, 2의 거칠기를 갖는데 이 값들은 벡터의 가중치로 계산하여 최종적으로 정점을 이동할 방향 벡터를 구한다. 이렇게 하면 거칠기가 낮은곳보다 높은곳으로 정점이 더 편향되므로 기하 오차율을 줄일 수 있다. 이 벡터의 X성분과 Y성분은 각각 편향맵의 R,G 필드에 저장한다. 벡터를 적용시킨 샘플 위치에 있는 높이값을 DEM데이터로부터 읽어 B채널에 저장하면 편향맵이 완성된다.

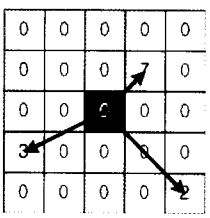


그림5 : 5x5 사이즈의 마스크를 갖는 편향맵의 연산과정

$$V_{bias} = \frac{\sum(V(x,y) \times R(x,y))}{\sum R(x,y)} \qquad (식4)$$

편향맵은 지형 메쉬의 정점에 매핑되는 텍셀만 사용하기 때문에 모든 텍셀에 대하여 계산할 필요가 없다. 따라서 렌더링할 지형 메쉬의 해상도와 같은 해상도의 텍스쳐로 생성한다.

3.3 제약 조건

제안하는 방법은 정점 쉐이더를 이용하여 정규격자로 이루어진 지형 메쉬의 각 정점에서 편향맵의 텍셀에 저장된 벡터값을 이용하여 정점들을 이동시킨 후 최종 영상을 얻는다. 이 때 이동 연산이 병렬로 수행이 되기 때문에 빠른 렌더링이 가능하다.

그러나 편향맵이 생성되는 과정에서 정점이 겹치거나 지형이 꼬이는 기하 뒤틀림 현상이 발생 할 수 있다. 따라서 편향맵에서는 각 정점들이 이동할 때 서로의 영역을 침범하지 않도록 해야 한다. 본 논문에서는 거칠기맵과 편향맵의 마스크 크기 $M(l)$을 식5 와 같이 계산한다. 여기서 l은 지형 메쉬의 상세단계 레벨로 원본 데이터를 0으로 하며 간략화 할수록 값이 커진다. 이 식과 같이 마스크 크기를 지정할 경우 한 정점의 최대 이동범위와 다른 정점의 최대 이동범위와 겹치는 점이 생기지 않기 때문에 정점과 겹치거나 꼬이지 않는다.

$$M(x) = 2^l + 1 \qquad (식5)$$

4. 실험결과

실험은 Intel(R) Core™2Duo CPU E8400 3.00GHz에 DDR2 4GB의 주 메모리를 갖는 시스템에서 수행하였다. 그래픽 장치는 512MB의 비디오메모리를 갖는 Radeon™ HD4850을 사용하였다. DirectX 10 라이브러리를 사용하였고 뷰포트 크기는 1024x768로 했다. 데이터셋은 1025x1025 해상도의 8비트 Grand Canyon DEM 데이터를 사용하였다. 상세단계를 이용한 렌더링 기법에서 발생하는 기하파핑을 측정하기 위하여 계층적으로 생성한 편향맵과 DEM 데이터의 밉맵으로 각각 지형 메쉬를 생성하여 상세단계별 기하파핑 정도를 비교하였다.

그림6은 밉맵와 편향맵을 원본영상과 비교를 한 결과를 보여 준다. 상단의 그림은 1025x1025 크기의 원본 데이터이며 하단의 그림들은 각각 129x129의 밉맵(좌측)과 편향맵(우측)을 적용한 결과영상이다. 밉맵이 적용된 메쉬를 렌더링한 영상에 비하여 편향맵이 적용된 메쉬를 렌더링한 영상에 특징점이 잘 유지되는 것을 볼 수 있다. 거칠기 값이 높은 지점은 지형이 울퉁불퉁하거나 급격히 지형의 경사도가 변하는 지점이기 때문에 기하파핑이 크게 발생한다. 편향맵은 각 정점을 거칠기 값이 낮은 지점에서 높은 지점으로 옮겨주는 역할을 하기 때문에 지형의 특징점을 잘 유지시키며, 이는 기하파핑을 효과적으로 줄여준다.

표1: 밉맵과 편향맵의 해상도별 상세단계 기법의 화질비교

간략화 한 지형 데이터	비교할 지형 데이터	밉맵을 이용한 방법의 오차 픽셀 수	편향맵을 이용한 방법의 오차 픽셀 수	밉맵을 이용한 방법의 오차 비율	편향맵을 이용한 방법의 오차 비율
513×513	1025×1025	591656	551203	77%	72%
257×257	1025×1025	676631	642022	82%	78%
	513×513	655865	384650	79%	46%
129×129	1025×1025	714072	703617	87%	85%
	513×513	709335	503677	86%	61%
	257×257	692069	472715	84%	57%
65×65	1025×1025	746398	745004	91%	90%
	513×513	746124	596032	91%	72%
	257×257	745864	587142	91%	71%
	129×129	741058	565882	90%	68%

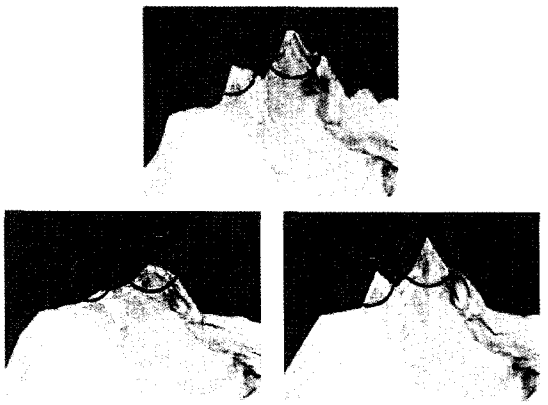


그림6 : 129×129 크기의 DEM 밉맵(좌측)과 같은 크기의 편향맵(우측)을 사용한 영상 비교

기하파핑 정도를 측정하기 위하여 각각의 데이터를 상세단계별로 렌더링한 결과영상을 화소 단위로 비교하여 표1에서 오차를 측정하였다. 데이터는 편향맵을 사용한 방법과 밉맵을 사용한 방법을 따로 측정하였다. 이것으로 본 논문에서 제안한 편향맵을 사용하여 상세단계선택을 하였을 경우와 기존의 밉맵을 사용한 경우에[1,2,3,4] 발생한 오차의 정도를 비교할 수 있다. 오차의 측정기준은 두 렌더링 결과영상에서 서로 다른 색상을 가지고 있는 픽셀의 숫자이다. 오차 비율은 결과 영상의 총 픽셀수와 오차 픽셀수의 비율이다.

표에서 보는 것과 같이 밉맵을 이용했을 경우보다 편향맵을 사용했을 경우 가까운 상세단계일수록 데이터들간의 오차율이 현저히 낮은 것을 확인할 수 있다. 이는 밉맵을 이용한 상세단계조절 기법에서[1,2,3,4] 편향맵을 적용하면 밉맵을 사용했을 때보다 기하파핑 현상이 현저히 감소됨을 의미한다.

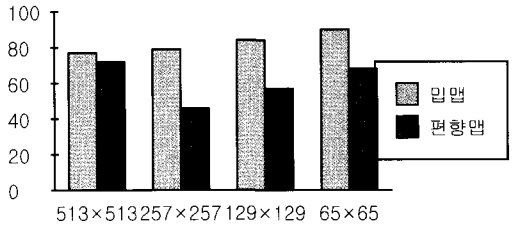


그림7 : 밉맵과 편향맵을 사용한 렌더링 방법들의 상세단계조절에 따른 화소단위의 오차율 비교

그림 7은 고정된 시점에서 순차적으로 상세단계 레벨이 커질 경우 발생하는 오차율을 보여준다. 밉맵을 이용한 방법보다 편향맵을 이용한 방법이 오차율이 평균적으로 낮은 것을 알 수 있다. 그림8과 그림9는 비교 실험에서 사용된 결과 영상이다. 와이어 프레임 영상인 그림10과 그림11을 보면 편향맵을 사용하였을 때 지형의 기하파핑을 줄이기 위해 정점이 적절한 위치로 이동된 것을 확인할 수 있다.

표2에서는 밉맵 데이터의 용량과 편향맵의 용량을 비교하였다. 밉맵 데이터는 8비트의 이미지 파일이며 편향맵은 벡터를 저장하는 24비트의 이미지 파일이기 때문에 평균적으로 3배 더 크다. 특히 해상도가 높을수록 차이가 크므로 외부메모리 처리 기법에서는 적은 해상도로 이미지를 나누어 관리하는 것이 유리하다.

표3은 편향맵과 같은 해상도의 밉맵을 이용한 경우와 편향맵을 사용하여 시각화 하였을 경우의 렌더링 속도를 측정한 결과이다. GPU를 사용하여 렌더링 하였고 속도 차이는 무시할만 하다. 이것은 편향맵을 적용해도 기존 방법에 비해 속도가 느리지 않으면서 지형의 중요한 특징들을 잘 표현해 준다는 것을 의미한다.

표2: 데이터 크기에 따른 용량 비교(BMP 기준)

데이터 크기	밉맵	편향맵	차이
513×513	220KB	771KB	3.5배
257×257	63KB	193KB	3.06배
129×129	18KB	48.9KB	2.72배
65×65	6KB	12.4KB	2.06배

표3: 데이터 크기에 따른 프레임률 비교

데이터 크기	밉맵	편향맵	속도차(%)
513×513	344	342	0.5%
257×257	1147	1132	1.3%
129×129	2745	2701	1.6%
65×65	3382	3328	1.5%

5. 결론

지형 메쉬의 간략화 과정에서 발생하는 기하오차를 줄이기 위해서 거칠기맵과 편향맵을 이용한 렌더링 방법을 제안하였다. 거칠기맵은 DEM데이터로부터 지형의 거칠기를 가우시안 라플라시안 필터로 구하여 저장한다. 편향맵은 거칠기맵의 거칠기값이 높은 지점으로 지형격자의 정점들을 움직이는데 사용한다. 따라서 편향맵에는 각 정점을 이동할 벡터를 저장한다. 전처리 단계에서 생성되는 거칠기맵과 편향맵은 GPU에서 처리하기 적합하게 텍스쳐로 저장이 되며 이는 실시간 렌더링시 GPU가속효과를 높일 수 있다. 실험결과 편향맵을 사용한 경우와 DEM데이터 원본을 그대로 사용하였을 경우 속도차이는 없었다.

감사의 글

이 논문은 2011년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(2011-0003842)

참고 문헌

[1] F. Losasso, and H. Hoppe, "Geometry Clipmaps: Terrain Rendering Using Nested Regular Grids.", Proceedings of the 2004 SIGGRAPH Conference, Vol. 23, pp.769-776, 2004

[2] C. Tanner, C. Migdal, and M. Jones, "The clipmap: A Virtual Mipmap.", Proceedings of the 25th annual conference on Computer graphics and interactive techniques, pp.151-158, 1998

[3] Y. Livny, N. Sokolovsky, T. Grinshpoun, and J. El-Sana, "A GPU Persistent Grid Mapping For Terrain Rendering", Visual Comput 24, 139-153, 2008

[4] L. Yotam, K. Zvi and El. Jihad, "Seamless Patches for GPU-Based Terrain Rendering", WSCG, 2007

[5] P. Lindstrom, D. Koller, W. Ribarsky, L. Hodges, N. Faust, and G. Turner, "Real-time, Continuous Level-of-Detail Rendering of Height Fields", Proceedings of ACM SIGGRAPH'96, Addison Wesley pp.109-118, 1996.

[6] S. Röttger, W. Heidrich, P. Slusallek, and H. Seidel, "Real-Time Generation of Continuous Levels of Detail for Height Fields", Proceedings of 6th International Conference in Central Europeon Computer Graphics and Visualization, pp.315-322, 1998.

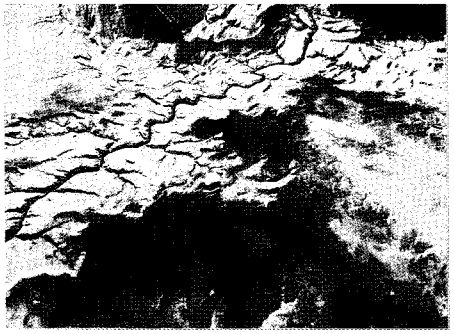
[7] M. Duchaineau, M. Wolinsky, D. Sigeti, M. Miller, C. Aldrich, and M. Mineev-Weinstein, "ROAMing terrain: real-time optimally adapting meshes", In: Proceedings of Visualization''97, pp.81-88 ,1997.

[8] T. Ulrich, "Rendering massive terrains using chunked level of detail control", In: Proceedings of ACM SIGGRAPH'02, Addison Wesley, 2002.

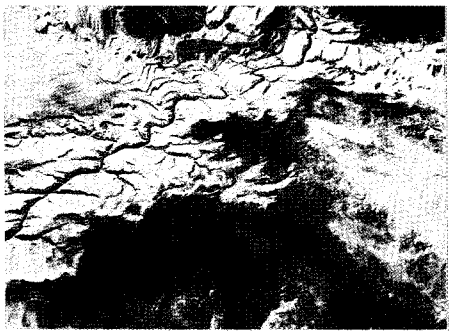
[9] C. Dachsbacher, and M. Stamminger, "Rendering Procedural Terrain by Geometry Image Warping", Proceedings of Eurographics Symposium on Rendering, pp. 103-110, 2004

[10] J. Peter-Pike, P. Sloan, D. M. Weinstein, and J. D. Brederson, "Importance Driven Texture Coordinate Optimization", Computer Graphics Forum, Volume 17, Number 3, pp. 97-104, 1998

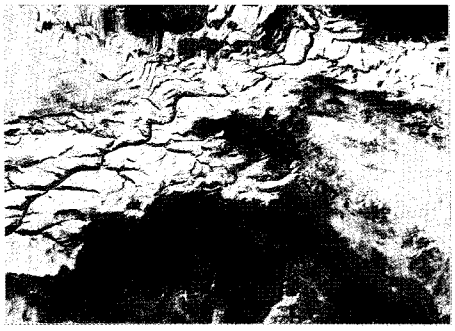
[11] D. Marr and H. Hildreth, "Theory of edge detection," Proc. Roy.Soc. London, vol. B207, pp. 187-217, 1980.



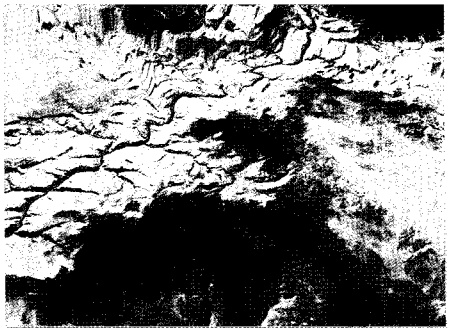
(a) 65×65



(b) 129×129

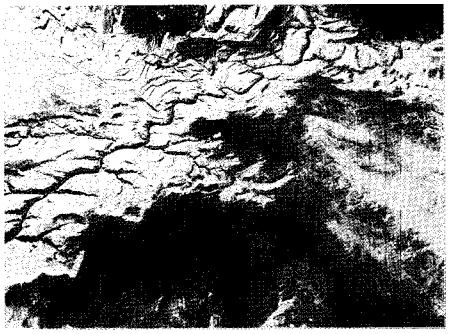


(c) 257×257

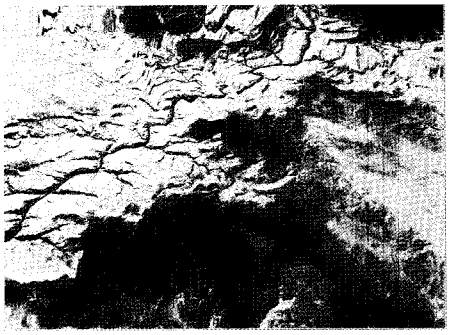


(d) 513×513

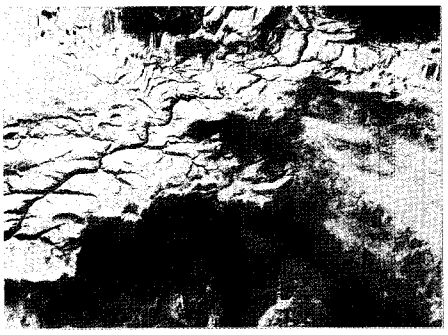
그림8 : 밉맵으로 렌더링한 영상



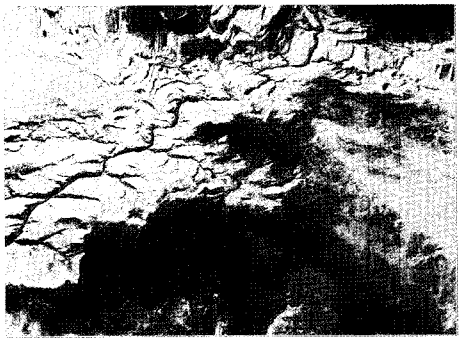
(a) 65×65



(b) 129×129

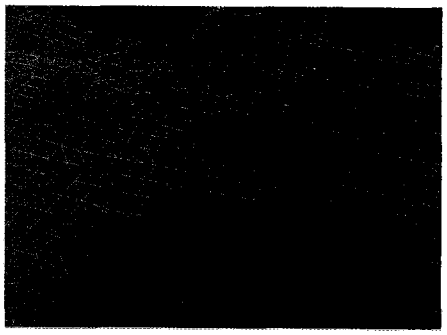


(c) 257×257



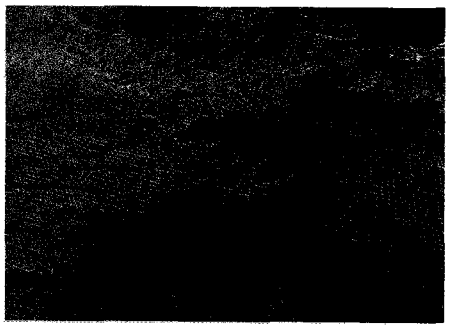
(d) 513×513

그림9 : 편향맵으로 렌더링한 영상

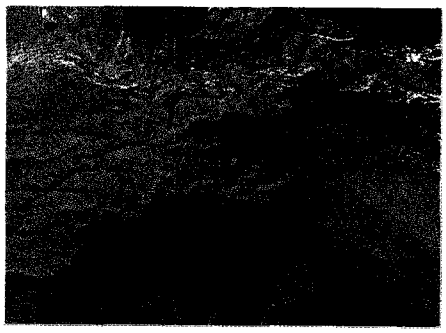


(a) 65×65

(b) 129×129

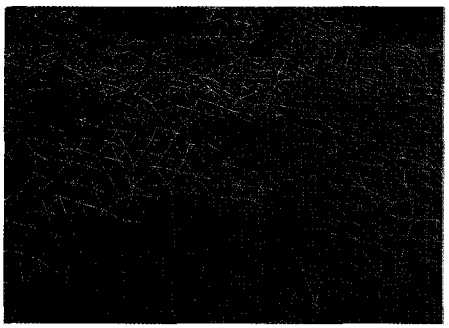


(c) 257×257



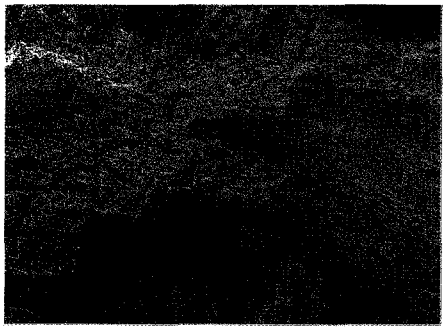
(d) 513×513

그림10 : 밉맵으로 렌더링한 영상의 와이어프레임

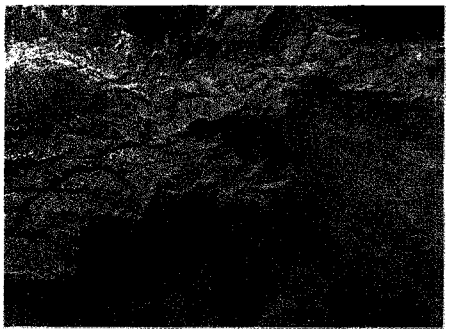


(a) 65×65

(b) 129×129



(c) 257×257



(d) 513×513

그림11: 편향맵으로 렌더링한 영상의 와이어프레임

〈저 자 소 개〉

이은석(Eun-Seok Lee)

- 2008년 2월 인하대학교 컴퓨터공학부 (학사)
- 2010년 8월 인하대학교 컴퓨터정보공학과(석사)
- 2011년 2월 ~ 현재 인하대학교 컴퓨터정보공학과(박사)
- 관심분야 : 지형 렌더링, 차세대 컴퓨팅, 상세단계선별

조인우 (In Woo Jo)

- 2010년 2월 인하대학교 컴퓨터공학부 (학사)
- 2010년-현재 인하대학교 컴퓨터공학부 (석사)
- 관심분야 : 지형 렌더링, 상세단계선별

신병석 (Byeong-Seok Shin)

- 1990년 2월 서울대학교 컴퓨터공학과 (학사)
- 1992년 2월 서울대학교 컴퓨터공학과 (석사)
- 1997년 2월 서울대학교 컴퓨터공학과 (박사)
- 2000년-현재 인하대학교 컴퓨터공학부 교수
- 관심분야 : 볼륨그래픽스, 차세대 컴퓨팅, 실시간렌더링