깜빡임 현상을 최소화하는 쿼드트리 기반의 CLOD기법

최이규[O] 신병석
인하대학교 전자계산공학과
g2021356@inhavision.inha.ac.kr bsshin@inha.ac.kr

A Quadtree-Based CLOD Method that Minimize Flickering Effect

Ei-Kyu Choi[0] Byeong-Seok Shin
Department of Computer Science & Engineering, Inha University Korea

요 약

컴퓨터 게임, 지리정보시스템(GIS), 가상현실 분야 등에서 환경표현의 기반이 되는 지형의 렌더링 기술은 매우 중요하다. 하지만 지형정보는 일반적으로 방대한 양의 데이터로 구성되어 처리가 쉽지 않다. 지형렌더링을 수행하기 위한 여러가지 기법들 중에 쿼드트리 구조를 기반으로한 연속적 상세단계기법이 있다. 본 논문에서는 쿼드트리 기반의 연속적 상세단계에서 시각 절두체 내에서 렌더링되는 삼각형의 개수를 조절하여 프레임율을 일정하게 유지하는 기법을 제안한다. 이 방법은 인접 프레임간의 일관성을 이용하여 최근 프레임에 가중치를 줌으로써 삼각형의 개수가 급격하게 변하는 것을 막는다.

1. 서론

컴퓨터 게임, GIS분야에서 지형의 표현은 매우 중요한 요소이다. 또한 전통적인 응용분야인 가상현실, 비행시뮬레이션 등에서 사실감을 더욱 극대화하기 위해서는 효과적인 지형렌더링이 필수적이다.

지형정보는 일반적으로 방대한 양의 데이터를 가지게 되므로 하드웨어의 속도가 빨라져도 실시간 처리가 쉽지 않다. 따라서 방대한 지형 데이터를 실시간 처리하기 위해 다양한 형태의 자료구조와 간략화기법들이 고안되었다[1-3].

지형 데이터를 간략화하기 위한 대표적인 자료구조로 ROAM(Real-time Optimally Adaptive Meshes)으로 불리는 이진삼각형트리 구조[4]와 쿼드트리 구조[5-8]가 있다. 그 중 쿼드트리는 시각 절두체 선별(view frustum culling)과 연속적 상세단계(CLOD) 구현이 용이한 구조로 널리 사용되고 있다. 연속적 상세단계를 이용하면 지형과 시점사이의 거리와 관측방향에 따라 지형의 상세단계를 결정하여 실시간 지형 영상을 얻을 수 있다. 이때 시각 절두체 내에서 그려지는 삼각형의 개수가 실시간으로 처리 가능한 정도를 넘어서 까지 증가하면 프레임율이 떨어지게 되므로 적절한 화질과 프레임율을 유지하도록 하기 위해서는 렌더링되는 삼각형의 개수를 일정하게 유지하도록 하는 방법이 필요하다. 이러한 방법을 적용할 때 삼각형의 개수가 연속된 프레임에서 급격하게 변하는 현상이 반복되는데 이로 인해 깜빡임(flickering)이 발생한다.

본 논문에서는 쿼드트리 기반의 연속적인 상세단계 기법을 적용한 지형 렌더링을 할 때, 시각 절두체 내에서 렌더링되는 삼각형 개수를 일정하게 유지하는 과정에서 생길 수 있는 영상의 깜빡임을 감소시키는 방법을 제안한다. 이 방법은 지형의 형태에 관계없이 깜빡임 없는 일정한 프레임율 유지를 위해 인접프레임간의 시간일관성을 이용하여 최근 프레임에 가중치를 주는 방법으로 삼각형의 개수가 급격하게 변하는 것을 막는다.

2절에서 쿼드트리 기반의 CLOD기법에 대해 설명하고, 3 절에서는 프레임율 유지를 위한 기존방법과 제안하는 방법을 비교하여 설명한다. 4절에서 기존 방법에서 결과에 영향을 주는 인자에 대한 분석을 하며 제안하는 방법의 결과를 기술하고 결론을 맺는다.

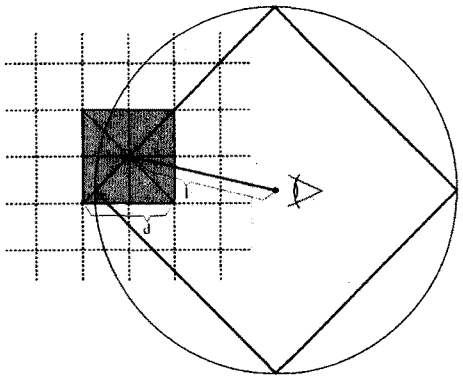
2. 쿼드트리 기반 CLOD 방법

쿼드트리는 2차원 공간을 4분할하여 트리를 구성하는 자료구조로서 지형을 효율적으로 분할하여 구성할 수 있다. 쿼드트리를 이용한 연속적 상세단계 방법은 Lindstrom이 처음으로 제안했다[2]. 이 방법은 삼각형의 분할과 합병을 결정하는데 상향식(bottom-up) 접근방법을 이용한다. 그후 Rottger는 하향식(top-down) 접근방법을 이용한 알고리듬을 제안했다[3]. 이 방법은 전처리를 통해 계산된 error metric을 이용하여 정점을 제거함으로써 추가비용 없이 geo-morphing을 할 수 있다. 이 방법은 구조적 특성상 연속적 상세단계의 구현과 시각 절두체 선별의 구현이 용이하다는 이점이 있다. 쿼드트리로 구성된 지형은 고도필드의 크기에 상관없이 이미지의 품질에 따라 그래픽 파이프라인으로 넘어가는 삼각형의 개수가 결정된다. 따라서 하드웨어의 성능에 대응하도록 이미지 품질을 조절하는 것이 가능하다.

쿼드트리를 이용한 CLOD기법으로 고도필드를 렌더링하는 순서는 다음과 같다.

(1) 고도필드 데이터를 탐색하며 재귀적으로 모든 단말 노드의 error metric을 계산한다.
(2) 계산된 단말의 error metric을 상위 노드로 전파한다.
(3) 계산된 error metric의 집합을 가지고 각 노드에서 상세도를 결정하는 함수로 현재 노드가 자식 노드를 가지게 되는지를 결정한다.
(4) 생성된 쿼드트리를 재귀적으로 탐색하면서 삼각형 집합을 생성한다.
(5) 생성된 삼각형 집합을 렌더링한다.

3의 단계는 하향식 단계로 재귀적으로 이루어지며 쿼드트리를 생성한다. 4의 단계에서 쿼드트리의 단말노드에 도달했을 때, 전체 혹은 부분적인 삼각형 팬이 그려진다.



[그림 1] 관측자의 거리와 노드 크기와의 관계 [5]

쿼드트리의 각 하위 노드에서 분할 여부를 결정할 때 [그림 1]에서 보는 바와 같이 시점과 노드간의 거리가 증가함에 따라 노드의 크기가 커지고 전체 해상도는 감소해야 하며, 이 조건은 식(1)을 만족하도록 한다.

$$\frac{l}{d} < C \quad \text{- 식(1)}$$

l : 노드와 시점간 거리, d : 노드 블록의 간선 길이

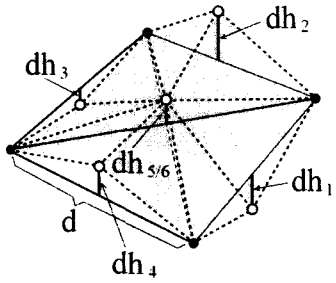
여기서 C는 지형의 최소 해상도를 나타내는 값으로서 전체 지형을 표현하는 삼각형 개수를 결정한다. C를 크게 하면 프레임당 많은 삼각형을 처리해야 하고 C를 작게 설정하면 처리할 삼각형이 적어진다.

상세단계를 표현하기 위한 또다른 매개변수로서 error metric이 있다. error metric으로는 식(2)에서와 같이 노드에 인접한 고도값들 가운데 그 차이의 절대값이 가장 큰 값을 이용한다.

$$d2 = \frac{1}{d} max| dh_i| \quad (i = 1..6) \text{ - 식(2)}$$

dh_i : 노드의 i 번째 간선에서 높이의 차

[그림 2]에서 보는바와 같이 $d2$ 값은 주변의 네 정점이 이루는 여섯 개의 간선에서 높이의 차이를 구한 후 그 가운데 최대값을 노드의 길이로 나눈 값이다. 이 값이 해당 노드의 error metric이 된다. 이 값은 모든 단말 노드에서 계산되어 상위 노드로 전파된다.



[그림 2] error metric의 측정

결과적으로 식(1)과 식(2)를 적용하여 각 노드의 상세도를 결정하는 식을 식(3)과 같이 정의할 수 있다.

$$f = \frac{l}{d \times C \times max(c \times d2, 1)} \quad \text{- 식(3)}$$

l : 노드의 중심과 시점간의 거리
d : 노드 간선의 길이
C : 최소 전역 해상도
c : 원하는 삼각형 개수로 도달하는 정도
$d2$: 노드에 인접한 고도값들의 차이 가운데 가장 큰 값

임의 노드에 대해서 결정 변수 f의 값을 계산했을 때 그 값이 1보다 작다면 그 노드는 다음 단계로 분할된다. c는 원하는 해상도까지 도달하는 정도를 나타낸다. 생성된 쿼드트리와 고도필드를 참조하여 삼각형의 집합을 생성함으로써 지형을 렌더링할 수 있다.

3. 깜빡임 현상을 최소화하는 CLOD방법

3.1 기존의 프레임율 유지 방법

연속적 상세단계기법에서 시점과 지형과의 거리에 따라 상세단계가 변한다. 시점이 지형에 가까워짐에 따라 상세단계는 높아지고 이에 따라 화면에 렌더링 되는 삼각형의 개수도 증가한다. 반대로 시점에서 멀어지면 삼각형의 개수는 감소한다. 그런데 이처럼 삼각형의 개수를 동적으로 조절하는 과정에서 시각 절두체 내의 삼각형의 개수가 하드웨어에서 처리할 수 있는 이상으로 증가하는 경우가 생긴다. 이때 한 프레임을 처리하기 위한 시간이 증가하게 되어 일정한 프레임율을 유지할 수 없게 된다. 기존의 CLOD를 이용하면서 프레임율을 유지하려면 삼각형의 개수가 최대가 되는 시점을 그래픽 하드웨어가 처리 가능한 정도로 맞추는 방법이 필요하다.

Rottger는 식(3)의 조건을 만족하도록 상세단계를 조절하도록 했다. 한 프레임에서 쿼드트리를 생성할 때, 노드의 분할여부를 결정하는 인자는 식(3)의 C이다. 이 값은 원래 상수로서, 시점과 노드간의 거리에 따라 해당 노드가 어느 단계까지 분할될지를 결정한다.

만약 C값을 변경할 수 있다면 해당 프레임에서 허용가능한 삼각형의 개수를 조절할 수 있다. C값을 크게 하면 식(3)의 조건에 따라 프레임 당 삼각형의 개수는 증가한다. 따라서 인접 프레임간 삼각형의 증가분을 피드백해서 다음 프레임의 C값에 적용하는 방법을 사용하면 삼각형의 개수를 일정하게 유지할 수 있다.

C값을 조절하는 조건으로, 이전 프레임에서 렌더링된 삼각형 개수와 현재 프레임의 삼각형 개수와의 차이를 이용한다. 그 값은 식(4)와 같이 표현할 수 있다.

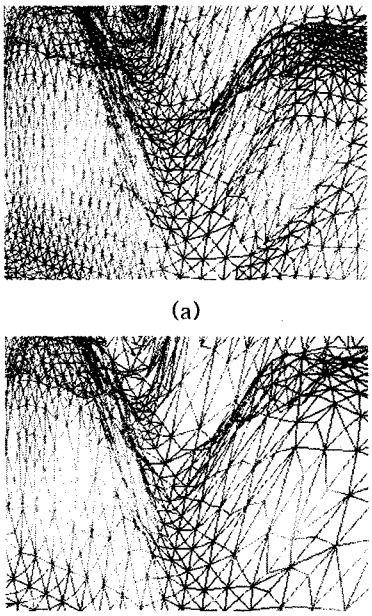
$$C_{next} = C_{prev} + \Delta n / a \quad \text{- 식(4)}$$

Δn : 프레임간 렌더링된 삼각형 개수 차,
$\Delta n = N_{curr} - N_{prev}$, N은 특정프레임에서의 삼각형개수
a : Δn을 C의 범위로 정규화 하는 상수, $a \neq 0$

식(4)에서 a는 상수 값으로 Δn을 C값의 범위로 변환하도록 한다. 이와 함께 식 (3)에서 c는 고정된 프레임율을 유지하도록 하는 상수이며 c 값을 변경하여 삼각형 개수 변화의 정도를 조절할 수 있다. 즉 c값이 작을 경우 삼각형의 수를 정밀하게 조절하지만 원하는 수에 도달하기 까지 시간이 오래 걸리며, 클 경우 원하는 단계까지 빨리 도달하게 되지만 일정 값에 도달하지 못하고 진동하는 문제가 있다.

프레임율을 일정하게 유지하기 위해 이전 C값을 피드백하는 기존 방법의 문제점은, 이전 프레임과 다음 프레임이 급격하게 변하는 굴곡이 심한 특정 지형에 시점이 고정되었을 때 깜빡임 현상이 생긴다는 것이다. 깜빡임 현상은 인

접한 두 프레임간의 삼각형 개수차이가 많이 나는 상태가 계속 반복될 때 나타난다. 깜빡임의 원인은 삼각형의 개수를 조절하는 C가 부적절하게 피드백 되기 때문이다.



(a)

(b)

[그림 3] 연속한 두 프레임사이에서 삼각형개수가 급격하게 변하는 경우, (a)의 삼각형 fan 개수 = 5256개, (b)의 삼각형 fan 개수 = 2745개

[그림 3]는 연속한 두 프레임의 삼각형 개수차이가 커서 깜빡임 현상이 나타나는 것을 보여준다. 이러한 깜빡임 현상은 식(2)의 c를 조절해서 감소시킬 수 있으나 지형의 굴곡이나 관측 위치에 따라서 삼각형 개수의 변화와 깜빡임의 정도가 다르기 때문에 적절한 값을 정하는 것이 쉽지 않다. c를 작게 하면 깜빡임 현상이 줄어들지만 원하는 삼각형개수로 도달하는 시간이 길어져 올바른 LOD효과를 내기 어렵다. 따라서 c값이 충분히 큰 악조건에서도 깜빡임 현상이 생기지 않도록 하는 방법이 필요하다.

3.2. 깜빡임 현상을 최소화하는 삼각형 개수 조절 방법

본 논문에서는 삼각형의 개수가 신속하게 안정된 범위로 수렴하도록 하면서 다양한 지형의 변화에 대해 일정한 삼각형 수를 유지하는 방법을 제안한다. 여기서는 이전 프레임의 삼각형 변화량 뿐만 아니라 C값의 변화정도를 참조하여 다음 프레임의 C값를 조절하도록 한다. 이때 이웃한 프레임간의 시간 일관성을 이용하여 현재 계산된 C값과 이전 값들에 대하여 최근 프레임의 C값에 높은 가중치를 부여하여 조정하는 방법을 이용하였다.

깜빡임은 연속된 프레임간의 삼각형 개수 변화량이 반복적으로 증감하면서 나타나는 것이기 때문에, 현재 삼각형의 개수가 증가했다면 다음 프레임에서는 개수가 감소할 것으로 예측할 수 있다. 따라서 현재 Δn과 이전 Δn은 부호가 반대일 가능성이 높다. 이 점에 착안해서 이전 몇 프레임에서 삼각형 수를 결정하는데 사용된 C값들을 평균하여 다음 프레임의 C값을 정하는 데 이용한다면 연속된 프레임간의 진동을 상쇄하는 효과를 얻을 수 있다. 이때 참조할 프레임의 개수는 짝수가 되어야 한다. 여기에 지역성을 적용하여 현재 프레임에서 가까운 최근의 C값에 높은 가중치를 적용하여 삼각형의 개수가 신속하게 안정된 범위로 수렴하도록 한다. 이것을 표현한 식은 아래와 같다.

$$C = \sum_{i=0}^{r} C_{n-i} w_{r-i} \qquad \text{- 식(5)}$$

$$w_i > w_{i-1} \ (i = 0...r), \quad \sum_{i=0}^{r} w_i = 1$$

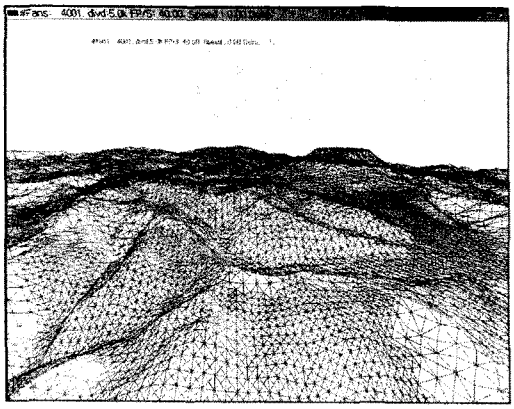
n = 현재 프레임.

이전 프레임의 C값과 Δn을 저장하기 위해서는 참조할 프레임 수 만큼의 기억공간과 계산이 필요하므로 적절한 값을 정해야 한다. 실험에 의하면 r=4일 때 가장 좋은 결과가 나왔다. 이렇게 하면 현재 프레임에서 삼각형 개수가 크게 변경된 경우에도 연속한 프레임에서의 계산값을 참조하여 필요한 값을 적용하게 된다. 이것으로 원하는 삼각형수로 신속하게 수렴하면서 깜빡임을 효과적으로 줄일 수 있다.

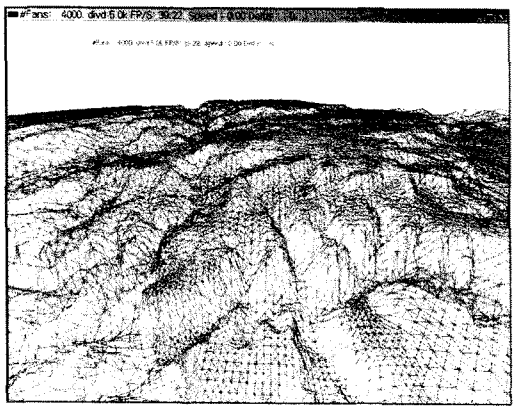
실제로 기존의 방법에서 c=1일 경우 본 논문에서 제안하는 방법과 결과에서 큰 차이가 없으나 기존방법에서 크게 깜빡임이 일어나는 c>3의 조건에서 깜빡임이 크게 감소하는 결과를 보인다.

4. 실험 및 결과

[그림 4]는 pugetsound지형(a)과 grandcanyon지형(b) 고도필드를 렌더링한 영상이다. (a)지형은 전반적으로 완만하며 (b)지형은 굴곡이 심한 것을 알 수 있다.

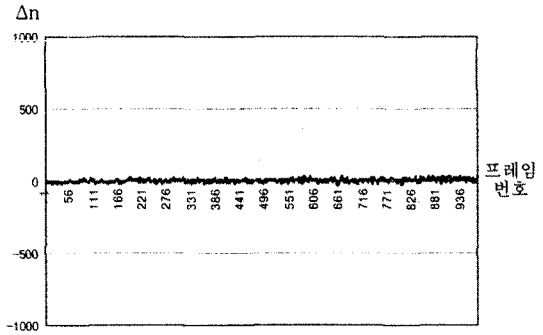


(a)pugetsound지형

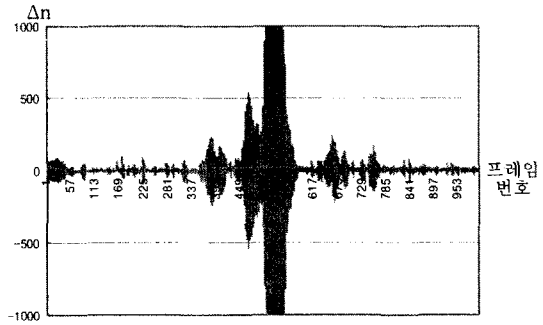


(b) grandcanyon지형

[그림 4] 실험에 사용된 지형을 렌더링한 영상



(a)

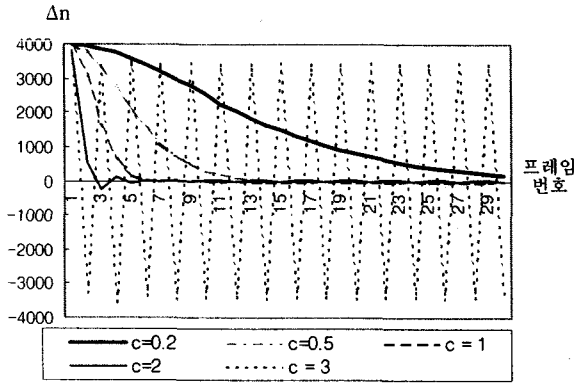


(b)

[그림 5] 기존 방법에서pugetsound지형(a)과 grandcanyon 지형(b)의 Δn의 변화

[그림 5]은 기존의 프레임율 유지 방법을 적용했을 때, 두 지형의 1000프레임동안 Δn의 변화를 나타낸 그래프이다. 여기서 조건은 c=5.0으로 했다. 동일한 조건에서 렌더링을 하였지만 [그림 4]의 (a)지형은 전반적으로 깜빡임이 없이 안정된 영상을 보였다. 반면 (b)지형은 삼각형의 개수차가 심하게 나타나는 것을 볼 수 있다. 이 실험에서 일정한 프레임율을 유지하기 위한 각 인자의 조건은 지형에 따라 다르다는 것을 알 수 있다.

[그림 6]은 식(3)의 인자인 c의 값의 변화에 따라 원하는 삼각형 수로 수렴하는 정도를 비교한 것이다. 그림에서 y 축은 이전 프레임과 삼각형개수의 차이(Δn)이고 x축은 프레임을 나타낸다. 이 실험은 grandcanyon 지형의 특정위치에서 시점을 고정한 후 수치를 대입한 것이다.

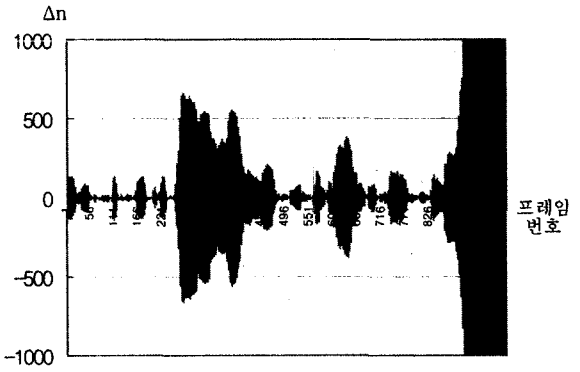


[그림 6] c의 값에 따른 Δn의 변화

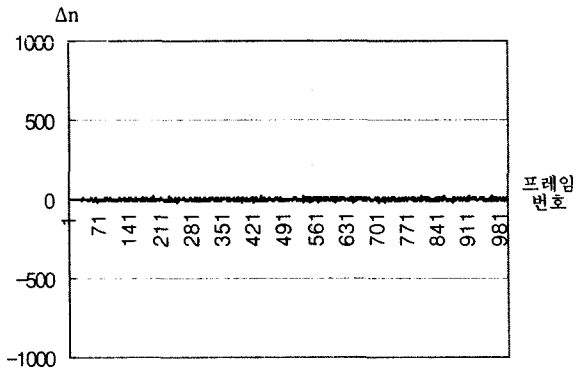
c 의 값이 증가함에 따라 Δn 이 0으로 수렴하는 속도가 빨라지는 것을 알 수 있으나 일정 값 이상에서 진동이 발생하는 것을 알 수 있으며 이 진동이 바로 화면의 깜빡임을 나타낸다. 반대로 c의 값을 감소시키면 깜빡임 현상이 없지만 대신 변화단계에 오랜 시간이 걸린다. 이 실험에서는 c=1 일 때가 가장 최적의 결과를 나타냈다. c=2일 때 약한 진동이후 안정화 되었으며 c=3 이상이 되면 진동이 발생했다.

다음은 c가 비교적 큰 값으로 진동이 많이 일어나는 조건에서, 기존의 방법과 본 논문의 방법을 비교한 것이다. [그림 6]의 실험에서 진동이 발생했던 값보다 좀더 높은 c=5의 값과, a=5000 으로 설정했다. 이 값은 grandcanyon지형에서 기존 방법으로는 매우 심한 진동이 발생한다.

[그림 7]과 [그림 8]은 1000프레임 동안 동일한 지형을 이동하며 렌더링했을 때 인접프레임과의 삼각형 차이를 나타낸 그래프이다. [그림 7]에서 본 논문의 방법을 적용하기 전에는 특정 위치에 대해서 큰 폭의 진동이 있으며 전체적으로 진동이 나타남을 알 수 있다. 그에 비해 [그림 8]는 전체적으로 소폭의 진동만이 보이며 전체적으로 평활함을 알 수 있다. 이 실험의 결과는 본 논문의 방법을 적용함으로써 전체적으로 안정된 프레임율로 실시간 렌더링영상을 얻을 수 있음을 보여준다. 이 방법은 렌더링되는 매 프레임마다 적용되며 추가 비용은 무시할 만하다.



[그림 8] c=5.0 일때 Δn의 변화



[그림 9] c=5.0 일 때, 본 논문의 방법을 적용한 후 Δn의 변화

5. 결론

게임, 가상현실등의 분야에서 지형데이터의 표현은 중요한 요소이므로 지형데이터의 빠르고 현실감있는 처리기법의 연구는 필수적이라고 할 수 있다. 이러한 지형데이터를 표현하기 위한 다양한 기법중에 쿼드트리를 이용한 연속적 상세단계 기법을 이용해 지형의 렌더링 영상을 얻었다. 렌더링 영상이 자연스럽게 보이도록 하기 위해서는 프레임율을 을 일정하게 유지하도록 하는 것이 중요한데 기존의 방법으로는 여러 다양한 지형에 대해 적용하는 것이 쉽지 않았

고 특정 지형에서 영상이 깜빡이는 문제가 발생했다. 본 논문에서는 일정한 프레임 율을 유지하기 위한 방법으로 시각절두체 내의 삼각형 개수를 유지하는 방법을 구현하는 과정에서 생길 수 있는 영상의 깜빡임 문제를 해결하기 위해 기존 연구를 분석하고 좀더 개선된 결과를 위한 방법을 제안하였다. 본 논문에서 제안한 방법으로 실험한 결과를 볼 때 깜빡임이 생기는 지형조건에 대해서 수렴속도를 보장하면서 안정된 지형영상을 얻는 것이 가능함을 알 수 있다.

* 본 연구는 대학 IT센터 육성 지원사업의 연구결과로 수행되었음

참고문헌

[1] B. Chen, J. Swan, E. Kuo, A. Kaufman, "LOD-Sprite Technique for Accelerated Terrain Rendering," Proceedings of IEEE Visualization 99, pp. 291-298, 1999

[2] H. Hoppe, "Smooth View-Dependent Level-of-Detail Control and Its Application to Terrain Rendering," Proceedings of IEEE Visualization 98, pp. 35-42, 1998

[3] P. Lindstrom, V. Pascucci, "Terrain Simplification Simplified : A General Framework for View-Dependent Out-of-Core Visualization," IEEE Transactions on Visualization and Computer Graphics, Vol. 8 pp. 239-254, 2002

[4] M. Duchaineau, M. Wolinsky, Davie E. Sigeti, Mark C. Miller, C. Aldrich and Mark B. Mineev-Weinsstein, "ROAMing Terrain: Real-time Optimally Adapting Meshes," SIGGRAPH, pp. 81-88, 1997

[5] P. Lindstrom, D. Koller, W. Ribarsky, Larry F. Hodges, N. faust and G. A. turner, "Real-Time, Continuous Level of Detail Rendering of Height Fields," SIGGRAPH, pp. 109-118, 1996

[6] S. Rottger, W. Heidrich, P. Slasallek, and H. Seidel, " Real-time generation of Continuous Levels of Detail for Height Fields," 6th International Conf. in Central Europe on Computer Graphics and Visualization, pp. 315-322, 1998

[7] Thatcher Ulrich, "Continuous LOD Terrain Meshing Using Adaptive Quadtrees," Gamasutra, 2000, http://www.gamasutra.com/features/20000228/ulrich_01.htm

[8] Renato Pajarola, "Large Scale Terrain Visualization Using the Restricted Quadtree Triangulation," IEEE Visualization 98, pp. 19, 1998