네트웍 환경에서의 대규모 지형 데이터 전송 및 렌더링

김대성°, 한정현
성균관대학교 정보통신공학부
{poohhere°,han}@ece.skku.ac.kr

Transmission and Rendering of Massive Terrain Data in Network Environment

Daesung Kim°, JungHyun Han
School of Information and Communications Engineering, Sungkyunkwan Univ.

요 약

본 논문에서는 대규모 지형 데이터를 이용한 네트웍 환경에서의 지형 탐색을 위한 다중 해상도 기법과 prefetching 기법을 제안한다. 지형 렌더링에 널리 사용되는 직각이등변 삼각형 메쉬 형태의 DEM 데이터를 정삼각형 메쉬 데이터로 재구성한 뒤, 이를 다중 해상도로 구조화하여, 네트웍 환경에서의 주요 문제점인 대역폭과 지연 문제를 보완 하였다. 본 기법은 3차원 지형 데이터를 이용한 온라인 게임 등에 응용될 수 있다.

1. 서 론

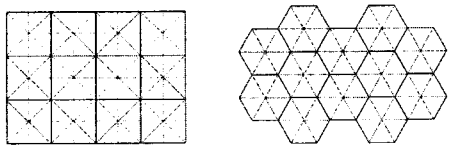
지형정보시스템(GIS), 비행 시뮬레이터, 게임 등의 분야에서 실시간 지형 렌더링이 필수적이다. 하지만, 일반적인 지형 데이터는 그 방대한 크기로 인해 실시간 렌더링이 쉽지 않다. 이를 해결하기 위해 LOD(Levels of Detail) 기법[1-6]이 다수 제안되었으나, 클라이언트-서버 환경에서의 지형데이터 전송에 대해서는 주목할만한 연구 성과가 존재하지 않는다.

본 논문은 대규모 3차원 지형 데이터를 이용한 온라인 게임에서의 LOD 데이터 전송 및 렌더링 기법을 제안한다. 즉, 방대한 크기의 지형데이터는 서버에 저장하고, 클라이언트 쪽 사용자의 뷰잉 및 이동 정보를 이용하여 필요한 데이터만을 전송한다. 동시에, 이러한 클라이언트-서버 환경에서 야기되는 대역폭 제한과 전송 지연이라는 문제를 해결하는 기법을 제시한다.

2. 지형 데이터의 LOD 구조

2.1 정삼각형 메쉬 데이터

지형 데이터 표현 기법으로는, 실제 지형 고도를 2차원 격자 형태로 샘플링한 DEM(Digital Elevation Model)이 널리 사용된다. DEM을 메쉬로 표현하면, <그림 1>의 (a)와 같은 이등변삼각형 메쉬가 된다. GIS나 비행 시뮬레이터 등의 분야는 정확한 DEM 데이터를 요구하지만, 게임에서는 그렇지 않다. 본 논문에서는 주어진 DEM 지형데이터를 정삼각형 메쉬로 리샘플링한다. 한 꼭지점을 공유하는 6개의 정삼각형을 모으면, <그림 1>의 (b)와 같은 육각형 맵을 구성할 수 있다.

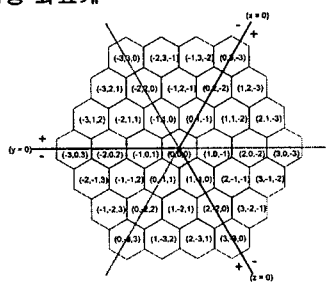


(a) 사각형 지형 데이터 (b)육각형 지형 데이터

<그림 1> 지형 데이터 재구성

이와 같은 육각형 지형 데이터를 이용하면 이른바 AOI(Area Of Interest)[7,8] 관리가 용이해진다. 즉, 현재 시점(viewpoint)를 중심으로 하는 일정 크기의 원이 AOI를 형성하게 되는 바, 이는 사각형 AOI에 비해 면적이 적다. 따라서, 서버에서 전송되는 데이터 크기를 줄일 수 있게 되고, 결국 네트웍의 대역폭을 줄이게 된다.

2.2 전체 지형 좌표계

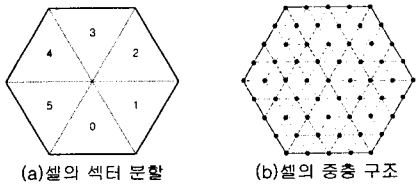


<그림 2> 지형 좌표계

전체 지형 데이터는 [9]에서 언급되었던 좌표계를 사용한다. 이는 <그림 2>에 도시된 바와 같이, 육각형 데이터를 표현하는 데 적합하여, 시점의 이동방향을 +x, -x, +y, -y, +z, -z의 여섯 방향으로 관리를 할 수 있도록 한다.

2.3 셀 구조

2.1절에서 기술한 육각형 영역 하나를 셀(cell)이라 부른다. 전체 지형 데이터는 이러한 셀들의 집합이다. 한편, 실제 셀은 단순한 6개 정삼각형의 집합이 아니라, 다중해상도를 가진 중층 구조로 이루어져 있다.



(a)셀의 섹터 분할 (b)셀의 중층 구조

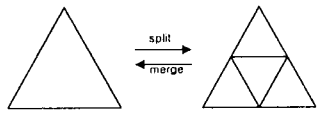
<그림 3> 셀 구조

즉, <그림 3>에 도시된 바와 같이, 섹터라 불리는 각각의

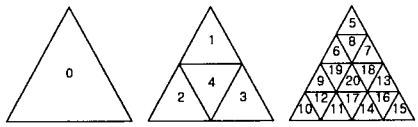
삼각형은 4개의 삼각형이 결합되어 만들어진 것이다. <그림 3>의 (b)는 3층 구조의 다중해상도 셀을 보여준다.

2.4 메쉬 단순화

본 기법에서의 메쉬 단순화는 <그림 4>에 도시된 바와 같이 [6]에서 사용되었던 삼각형 단위의 분할과 병합 방법을 적용하였다.



<그림 4> 삼각형의 분할과 병합

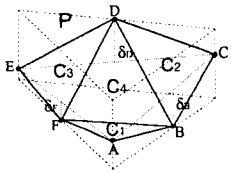


<그림 5> 삼각형의 인덱스 구조

이와 같은 단순화 기법을 통해 형성된 다중해상도 메쉬는 <그림 5>의 삼각형 인덱스 기법으로 이용해 저장된다. 삼각형은 부모 삼각형의 순서대로 분할이 되며 분할되는 자식 삼각형의 인덱스는 반시계 방향으로 순서가 정해지며 가운데 삼각형이 마지막이 된다. 이 경우, 부모의 인덱스를 P라고 했을 때 자식 삼각형의 인덱스 C_n은 $4 \times P + n$ ($n = 1, 2, 3, 4$) 으로 정의된다.

2.5 Object Space Error

앞서 언급된 바와 같이 본 시스템은 클라이언트-서버 환경에서 작동된다. 따라서, 일반적인 지형렌더링 기법과 같이 Screen Space Error값을 이용한 LOD 생성이 불가능하다. 따라서 본 논문에서는 Screen Space Error 대신 Object Space Error값을 이용하여 다중해상도를 구성하고 그에 따른 서버와 클라이언트간의 전송을 처리하도록 하였다.



<그림 6> Object space error

<그림 6>은 4개의 삼각형 C_1(A,B,F), C_2(B,C,D), C_3(D,E,F), C_4(B,D,F)가 P(A,C,E)로 병합되는 과정에서 발생하는 Object Space Error를 보여준다. 예를 들어, 꼭지점 D의 에러 δ_B는 $\delta_B = \left| \frac{A+C}{2} - B \right|$ 와 같이 표현된다.

각 자식(child) 삼각형의 에러값을 δ_{C1}, δ_{C2}, δ_{C3}, δ_{C4} 라고 하고 세 선분 꼭지점 에러를 δ_1, δ_2, δ_3 (δ_B, δ_D, δ_F에 해당)라고 했을 때, 부모 삼각형 P의 Object Space Error δ_P는

$$\delta_P = \begin{cases} 0 & : \text{if leaf node} \\ \max(\delta_1, \delta_2, \delta_3, \delta_{C1}, \delta_{C2}, \delta_{C3}, \delta_{C4}) & : \text{otherwise} \end{cases}$$

와 같이 표현된다. 이렇게 에러값을 정의하면, 부모 삼각형의 에러 값은 자식 삼각형의 에러 값보다 같거나 크게 된다. 서버측에서는 이렇게 구성된 에러 값들에 따라 다중해상도 삼각형들을 정렬하여 저장한다.

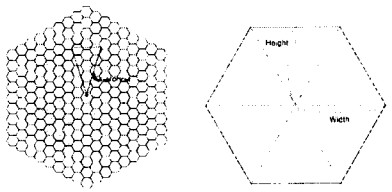
3. 지형 데이터의 LOD 전송

3.1 데이터 전송

클라이언트는 자신의 위치 정보를 서버에 알려주고 서버는 그에 해당하는 지형 데이터 정보를 클라이언트에게 전송한다. 이때 서버에서 가지고 있는 지형데이터는 2절에서 기술된 바와 같이 Object Space Error값에 의해 정렬이 되어 있고 이 에러 값이 큰 순서대로 전송된다. 따라서 클라이언트는 네트웍의 데이터 전송량이 허용하는 LOD 데이터를 가지게 된다.

3.2 Active Area

Active Area는 클라이언트에서 현재 메모리에 로딩되어 있는 영역을 나타낸다.



(a)Active Area (b)셀의 넓이, 높이의 정의

<그림 7> Active Area와 Cell의 크기

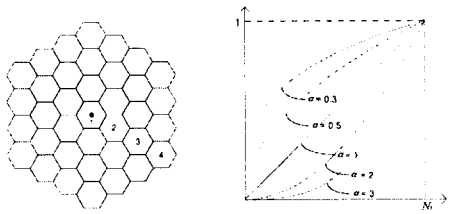
<그림 7>의 (a)에서 색칠된 부분이 Active Area를 나타내며 이는 FOV(Field Of View)의 회전반경을 포함하는 영역이다. Active Area의 크기는 다음과 같은 식으로 정의 된다.

$$F_{vf} < \left(1 + \frac{3}{2} A_s\right) \times W_c$$

여기에서 F_{vf}는 View Frustum에서의 Far의 길이 이며 A_s는 Active Area의 크기, W_c는 셀의 Width를 나타낸다. 이때 위 식을 만족하는 A_s의 최소 정수값이 Active Area의 크기가 되며 <그림 7>의 (a)의 경우 Active Area의 크기는 7이 된다.

3.3 에러값 보정

2.5절에서 언급된 Object Space Error값은 카메라와의 거리를 무시한 에러 값이다. 즉 Screen Space Error에 대한 보정이 필요하게 된다. 이를 수행하기 위해서는 $\delta_{new} = \delta_{old} / R$ 식에 의해서 에러값이 다시 계산되어져야 한다. 여기에서 R 은 시점(viewpoint)와 지형데이터간 거리값이다.



(a)Active Area 레벨 (b)에러 보정 그래프

<그림 8>에러 보정

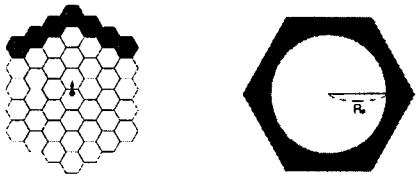
하지만 본 시스템은 클라이언트-서버 환경에서 수행되므로 클라이언트와 지형 데이터 간의 정확한 거리를 파악하는 것은 불가능하다. 그래서 본 논문에서는 셀간의 거리값에 의한 근사치를 이용하여 거리에 따른 에러값 보정을 수

행하는데 다음과 같은 식을 적용한다.

$$R_l = \frac{1}{N_l^{\alpha}} \times L_l^{\alpha} \quad (l = 1,2,\ldots N_l)$$

여기에서 R_l은 에러 보정에 사용할 결과 값이며 N_l은 Active Area의 크기 값, L_l은 Active Area에서의 레벨 값을 나타낸다. Active Area의 레벨 값은 <그림 8>의 (a)와 같이 중심부터 그를 둘러싸는 셀들의 개수이다. 그리고 α 는 상수 값인데 이 α 값에 따른 위 식의 그래프가 <그림 8>의 (b)에 도시하였다. 이 α 값은 클라이언트의 속성에 따라 결정이 된다. 예를 들면 Flight Simulator와 같이 지형과 멀리 떨어진 채 이동하는 경우는 α 값을 작게 하여 셀간의 거리에 의한 에러 차이를 줄인다. 한편, Walkthrough System과 같이 지형 바로 위에서 이동하는 경우에는 α 값을 크게 함으로써 셀간의 거리에 의한 에러 차이를 크게 하여 가까운 곳의 지형을 더 상세히 볼 수 있도록 한다. 이렇게 계산되어진 R_l에 따라 보정되는 에러값은 $\delta_{new} = \delta_{old} \times R_l$ 와 같이 계산이 되어진다.

3.4 Prefetching



(a)Prefetching (b)Prefetching영역의 정의

<그림 9> Prefetching

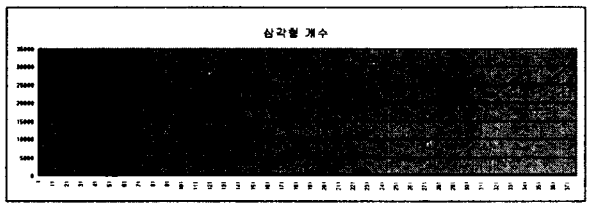
<그림 9>의 (a)는 Prefetching을 수행 예를 보여준다. 는 카메라가 위쪽으로 이동을 하고 있을 때 미리 이동할 위치에 대한 데이터, 즉 위쪽 색칠된 부분의 데이터를 받아오도록 하는 것이다. Prefetching 영역은 계산의 단순화를 위해 <그림 9>의 (b)와 같이 원으로 정의하며 그 원의 반지름 R_p는 다음과 같은 식으로 정의된다.

$$R_p = C_h - \left(\frac{N}{V_n} + S_p\right) \times V_c$$

여기에서 C_h는 셀의 높이, V_n은 네트웍 속도, N은 Prefetching 수행 시 전송되어야 할 데이터의 크기, S_p는 서버에서의 Prefetching 수행시간, V_c는 카메라의 이동 속도를 나타낸다. 이때 네트웍의 상황에 따라 Prefetching 데이터 전송이 완료되지 못 할 경우가 발생할 수 있는데 이를 위해 Prefetching 에러율을 계산하여 Prefetching 영역을 재조정하도록 한다. Prefetching 에러율은 $\varepsilon_p = \frac{N_t - N_c}{N_t}$ 와 같이 정의된다. 여기에서 N_t는 Prefetching 시 전송되어야 할 데이터의 크기를, N_c는 전송된 데이터의 크기를 나타낸다. 이때 이 에러값 ε_p 이 임계치인 τ_p 보다 클 경우 Prefetching 영역을 나타내는 기존 R_{po}를 $R_{pn} = R_{po} \times (1 + \varepsilon_p)$ 식에 적용시켜 새로운 영역인 R_{pn}으로 변경하여 네트웍 상황에 대응하도록 한다.

4. 결 과

약 10Km X 10Km넓이에서 leaf 노드의 한변의 길이가 10m인 지형 데이터를 이용하였으며 이를 위한 데이터베이스에는 삼각형 정보를 가지고 있는 약 100만개의 record가 생성되었다. 그리고 100Mbps의 로컬 네트웍 환경에서 테스트를 수행하였다. 이때 클라이언트가 60Km/h정도의 속도로 이동시의 결과를 <그림 10>에 도시하였는데 X축은 시간, Y축은 삼각형 개수를 나타낸다. 이 그림을 보면 클라이언트는 전체 100만 여개의 삼각형 정보 중에서 20000개에서 30000개정도의 삼각형 정보만을 유지하면서 지형탐색을 수행할 수 있었다.



<그림 10> 수행 결과

참 고 문 헌

[1] J. Falby, M. Zyda, D. Pratt, L. Mackey: NPSNET: Hierarchical data structures for realtime 3-dimensional visual simulation. Computers & Graphics, Vol. 17, No. 1, pp. 65. 1993

[2] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, and G. Turner."Real-Time, Continuous Level of Detail Rendering of Height Fields." *Proceedings of SIGGRAPH 96*, 109118. Aug. 1996

[3] M. A. Duchaineau, M.Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, andM. B.Mineev-Weinstein. "ROAMing Terrain: Real-time Optimally Adapting Meshes." *IEEE Visualization '97*, 8188. Nov. 1997.

[4] M. Reddy, Y. G.. Leclerc, L. Iverson, Nat Bletter, and Kiril Vidimce "Modeling the Digital Earth in VRML" In Proc. Of SPIE The International Society for Optical Engineering. Volume 3905, 113-121. 1999.

[5] M. Reddy, Y. G.. Leclerc, L. Iverson, Nat Bletter "TerraVision II : Visualizing Massive Terrain Database in VRML." IEEE Computer Graphics and Applications. 19(2):30-38. 1999

[6] H. Hakl, "Diamond LOD: CLOD for Height Fields." 2001.

[7] D. Schmalstieg and M. Gerautz, "Demand-Driven Geometry Transmision for Distributed Virtual Environment for Distributed Virtual Environments."In Proc. Of Eurographics'96, 15(3):421-432, 1996

[8] J. Chim, M. Green, R. Lau, H. V. Leong, and A. Si. "On Caching and Prefetching of Virtual Objects in Distributed Virtual Environments", In Proc. Of the sixth ACM international conference on Multimedia, 1998, 171-180, Sept. 1998.

[9] Innchyn Her. "Geometric Transformations on Hexagonal Grid", IEEE Transaction on Image Processing, vol. 4. No. 9. Sept. 1995.