GPU 기반의 효율적인 거대 분자의 실시간 렌더링 기법

이준[O†] 박성준[†] 김지인[††]
[†] 건국대학교 컴퓨터정보통신학과, [††] 건국대학교 인터넷미디어공학과
{junlee[O], hcipsj, jnkm }@konkuk.ac.kr

An Efficient Real-Time Rendering of Large Molecular Models based on GPU

Jun Lee[O†], Sungjun Park[†], Jee-In Kim[††]
[†] Department of Computer Science & Engineering/[††] Department of Internet & Media,
Konkuk University

요 약

정보생물학 분야에 있어서 분자 구조를 3차원으로 렌더링하여 보여주는 것은 매우 중요한 작업이다. 특히 분자의 표면 렌더링은 분자의 3차원 구조 분석 등에 중요하게 사용된다. 그러나 분자 표면 렌더링을 수행하기 위해서는 많은 양의 폴리곤이 필요하게 된다. 특히 대장균 바이러스와 같은 분자량이 많은 거대 분자를 자연스럽게 렌더링 하기 위해서는 고가의 그래픽 전용 워크스테이션을 사용해야 한다. 본 논문에서는 저렴한 일반 PC 급 시스템에서도 거대 분자를 무리 없이 렌더링 할 수 있는 효율적인 알고리즘을 제안 하였다. 제안하는 알고리즘은 높은 속도와 좋은 화질을 유지할 수 있는 Hybrid Point & Polygon 렌더링 기법이다. 이 알고리즘은 계층적인 자료구조인 옥트리(Octree)를 사용하였으며 최적의 성능을 내기 위하여 GPU가 작업을 처리 한다. 제안된 알고리즘의 성능 평가는 일반 PC급에서 수행되었으며 특히 그래픽 카드 2개를 병렬로 연결하여 높은 성능을 낼 수 있는 SLI(Scalable Link Interface) 환경에서 평가를 수행 하였다.

1. 서론

분자 표면 렌더링(Molecular Surface Rendering)[1]은 분자의 반지름 정보를 이용하여 분자 구조의 전체적인 3차원 모습을 만들어 내는 것을 말한다. 이러한 분자 표면 렌더링은 분자 구조의 분석, 단백질 간의 상호작용 분석, 신약 물질 설계 등에 중요하게 사용되는 분자 모델링의 한 연구 분야이다.

표면 렌더링을 사용하여 분자 구조를 3차원으로 보여줄 경우, 분자량이 증가할수록 각 분자를 구성하고 있는 폴리곤(Polygon)의 개수가 급격히 늘어나기 때문에 거대 분자 구조의 실시간 관측이 매우 어렵다. 이러한 문제점을 해결하기 위해, 보통은 좀더 빠른 렌더링 속도를 위해 고가의 그래픽 전용 워크스테이션을 구입하여 실험을 하고 있다. 또 다른 방법으로는 많은 양의 폴리곤 수를 줄이거나, 혹은 폴리곤 대신 점으로 렌더링 하는 방법을 사용하기도 한다.

본 논문에서는 PC를 사용하여 GPU(Graphic Processing Unit)를 장착한 일반 그래픽 카드를 사용하고, 많은 양의 폴리곤 수를 급격히 줄일 수 있는 알고리즘을 개발하여, 거대 분자를 표면 렌더링 하는데 있어서 최적화된 렌더링 기법을 소개한다.

본 논문에서 제안하는 방법은 계층적 자료구조인 옥트리(Octree)를 사용 하여 분자의 표면 폴리곤들을 구성 한 후에 이들을 사용자의 시점에 따라 상대적으로 중요한 부분은 폴리곤으로 렌더링 하고, 가장자리나 뒤에 위치하는 부분의 다수의 폴리곤들은 점으로 렌더링 하게 된다. 이렇게 함으로써 제안하는 알고리즘은 렌더링하는 분자의 좋은 화질 유지 및 속도 향상이라는 두 가지 장점을 가지고 있다.

본 논문에서는 제안하는 알고리즘의 효율성을 검증하기 위해 본 연구팀에서 개발한 가상현실 기반의 분자 모델링 시스템인 VRMMS [2,3]와 기존 렌더링 방식을 사용한 시스템과 렌더링 속도를 비교 분석 하였다. 실험 데이터는 대장균 바이러스, ATPase 등 실제 시뮬레이션 실험에 사용 되는 거대 단백질들을 사용하였다.

2. 관련연구

분자 표면 렌더링을 생성하는 알고리즘들은[4, 5] 분자량이 증가 할수록 분자를 구성하고 있는 폴리곤 수가 급격히 증가 하기 때문에 거대 분자의 실시간 렌더링이 어려운 단점이 있다. 이러한 문제점을 해결하기 위해 VMD[6]는 그래픽 전용 워크스테이션에서 GPU가속을 이용한 렌더링 방식을 제안 하였다. 하지만 일반 사양의 그래픽 카드에서는 이러한 성능이 나오지 않는 단점이 있다.

렌더링시 폴리곤의 개체 수를 줄이기 위한 방법으로 Qsplat [7]은 폴리곤 대신 점으로 렌더링 하는 방법을 제안하였다. 이 방법은 대화식 수준의 프레임 갱신 속도를 얻었으나 폴리곤 렌더링에 비해 화질이 떨어지는 단점이 있다. 최근에는 속도의 개선을 위해 GPU기반

3.3 옥트리 렌더링

렌더링 단계에서는 옥트리를 탐색 하며 렌더링을 수행하게 된다. 그림3처럼 옥트리는 단말 노드까지 순회를 하면서 해당 노드의 바운딩 박스의 크기가 실제 사용자의 관측 화면에서 일정 Threshold 값 과 비교한다. Threshold 값은 사용자의 관측 시점에서 점 또는 폴리곤으로 렌더링 하기 위한 기준이 되는 값이다. Threshold값은 분자 전체 크기의 비율에 맞게 결정 된다. 바운딩 박스의 크기가 Threshold값 보다 작게 되면 이 노드에서 포함하고 있는 폴리곤들을 하나의 점으로 렌더링 한다. 주로 이 경우에는 해당 노드가 화면의 뒤에 위치하거나 가장 자리에 있게 되는 경우이므로 화질에 중요한 영향을 주지 않게 된다. 반대로 Threshold 값 보다 크게 되면 이 노드가 포함 하고 있는 폴리곤들을 바로 렌더링한다. 이 노드는 실제로 사용자의 시점에 매우 가까이 있는 폴리곤들로써 이들을 점을 사용한 스플래팅으로 처리시 화질이 떨어지게 되므로 폴리곤으로 렌더링 함으로써 높은 화질을 유지 할 수 있다. 그림 4는 실제 시스템에서 Threshold값을 판별하여 점 및 폴리곤으로 렌더링한 장면이다.

3.4 GPU 가속

렌더링시 속도 개선을 하기 위해 GPU가 사용 된다. GPU작업은 GLSL(OpenGL Shading Language)을 사용하였으며 버텍스 쉐이더(Vertex Shader)에서 작업을 주로 처리한다. 렌더링에서 점으로 렌더링 하는 경우에는 이 정보를 버텍스 쉐이더에서 카메라의 위치에 따른 계산을 한 알맞은 크기의 정점을 처리 한다. 반면 폴리곤 렌더링 과정에서는 해당 폴리곤의 개수만큼 배열 형태로 CPU를 거치지 않고 GPU로 바로 DMA전송을 통하여 처리한다. 다음은 GPU에서 바운딩 박스를 점으로 렌더링 하게 될 때 렌더링 하게 되는 점의 크기를 결정하기 위해 사용 되는 수식이다[8].

$$Size_{node} = r \cdot \frac{n}{z_{eye}} \cdot \frac{h}{t-b}$$

$Size_{node}$ 는 점으로 렌더링 하게되는 노드의 크기를 의미하며, Z_{eye} 는 사용자 시점으로부터 해당 노드의 거리, r은 옥트리 노드의 크기이며 n, t, b, h 는 그림 3에 나와 있는 투시 투영에 대한 인자들이다. 이렇게 점으로 렌더링시 점의 크기를 설정해 줘야 하는 이유는 기존의 폴리곤 대신에 점으로 렌더링시에 발생하는 홀 현상 및 계단 현상을 최소화 할 수 있기 때문이다. 이러한 점의 크기에 대한 계산을 GPU의 버텍스 쉐이더에서 처리함으로써 CPU의 부하를 줄일 수 있다.

4. 실험

본 실험에서는 실제 분자 시뮬레이션 구조에서 사용되는 거대 바이러스 물질들의 표면 렌더링 시간을 측정 하였다. 제안한 방법의 성능 개선을 비교 하기 위해 기존의 일반적인 렌더링 방식을 지원 하는 VMD 시스템과 성능 평가를 비교 하였다. 실험을 한 시스템은 다음의 표 1과 같이 2개의 시스템에서 적용 하였다. 시스템1은 고가의 그래픽 카드를 장착하여 테스트 하였으며 시스템 2는 SLI 가속을 지원 하는 병렬 그래픽 카드를 장착한 시스템을 적용 하였는데, 이는 시스템 1에서 실험한 그래픽 카드의 가격에 비해 3배 정도 가격이 싼 일반 사양의 그래픽 카드 2개를 병렬 가속을 이용한 실험을 통하여 향후 그래픽 카드의 병렬 지원을 통한 가격대 성능비가 우수한 시스템을 구축할 수 있다.

표 1 실험에 사용된 시스템

Polygons	시스템 1	시스템2
CPU	Intel Pentium 4 3.2Ghz	AMD 애슬론 3500+
RAM	2GB	2GB
VGA	nVidia Quadro 3400	nVidia Geforce 6600GT X2 (SLI)
OS	Windows XP	Windows XP

실험에 사용된 데이터는 PDB Bank[10]에서 다운로드 한 거대 단백질 분자를 사용하였다. 실험을 4개의 단백질로 한정한 이유는 이보다 작은 수의 분자량을 가진 단백질을 테스트 할 경우에는 기존의 방식으로도 높은 대화식 프레임 속도를 가질 수 있기 때문이다. 실험은 각 단백질의 파일에 대해서 VMD 시스템과 우리가 구현한 VRMMS에서 분자 표면 렌더링 알고리즘을 사용하여 렌더링 하였으며, FPS 및 실제 렌더링 시간을 측정 하였다. 다음의 그림5는 실험에 사용된 단백질 중 1A0D라는 이름을 가지는 대장균 세포에 대한 실제 렌더링 결과 이다.

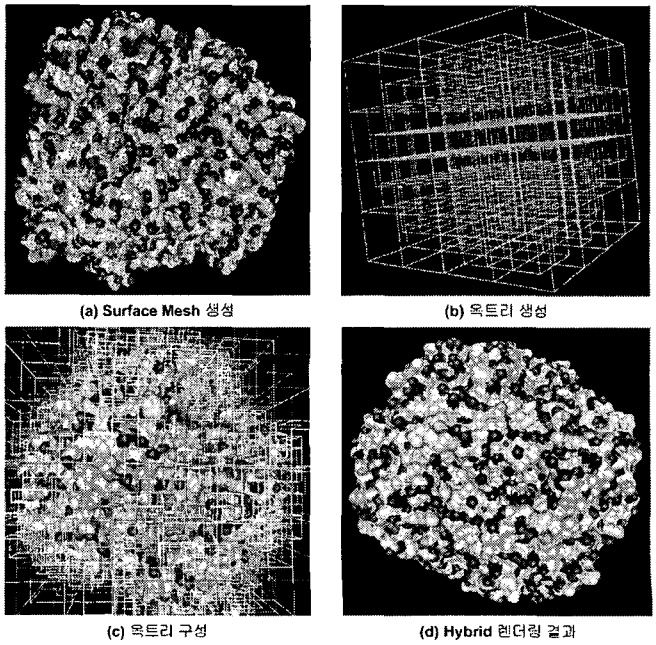

그림 5 대장균 단백질 렌더링 결과 (70만 폴리곤)

에서 점으로 렌더링 하는 연구[8]가 진행되고 있다.

또 다른 방법으로 폴리곤과 점을 혼합하여 렌더링[9]하는 연구가 진행 되고 있다. 이 방법은 좋은 화질과 빠른 속도를 얻을 수 있는 장점이 있다. 그러나 이 방법은 3D Scanner로부터 얻어진 Geometry Image를 이용하여 렌더링을 수행하기 때문에 3차원 분자구조의 정보에 의해 폴리곤을 생성 하는 분자 표면 렌더링에 적용 하기에는 적합하지 않다. 또한 폴리곤들을 점으로 렌더링 할 경우 분자들의 시각적인 정보가 섞이는 문제 등을 고려해야 한다.

본 논문에서는 이러한 문제를 해결하기 위해 옥트리를 사용한 전처리를 통하여 분자 구조의 폴리곤들을 점 및 폴리곤으로 렌더링 하며, 시각적으로도 분자 구조의 특징에 최적화된 방법을 제안 하였다. 본 논문에서 제안한 렌더링 알고리즘을 사용한다면 대규모 병렬 처리 시스템을 통하여 수행하는 거대 바이러스 물질의 닥킹 시뮬레이션 등에 적용이 가능하다. 특히 3차원 가상현실 시스템 기반에서 거대 바이러스 물질의 관측 및 설정을 통한 시뮬레이션에 유용하게 사용될 수 있다.

3. GPU 기반의 Hybrid Point & Polygon 렌더링

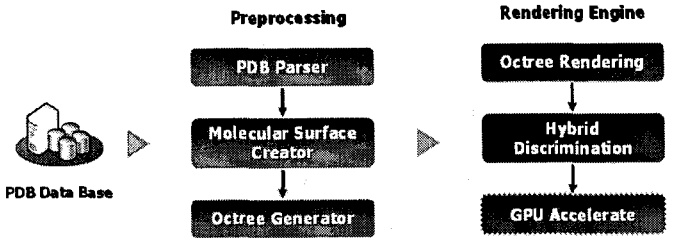

그림 1 VRMMS 시스템 구성도

3.1 시스템 구성

그림1은 제안하는 알고리즘을 구현한 시스템의 구성도이다. 시스템은 크게 전처리 과정에 대한 구성요소와 렌더링 엔진에서 사용하는 구성요소로 이루어져 있다. 전처리 과정에서는 먼저 렌더링을 원하는 분자 구조 파일(PDB, Protein Data Base)을 로딩 하게 된다. 로딩된 분자 구조를 처리하는 작업은 PDB Parser에서 담당을 하게 된다. PDB Parser를 통하여 생성된 3차원 분자 구조 정보는 Molecular Surface Creator를 통하여 분자 표면 생성 알고리즘을 이용하여[4] 분자 표면에 대한 메시를 생성한다. 생성된 메시 정보는 이후 Octree Generator에서 옥트리를 생성하고 옥트리의 각 노드에 분자 표면의 각 폴리곤 정보들을 구성하게 된다. 옥트리를 사용 하는 이유는 사용자의 시점에 따른 분자 구조들의 계층적 구성의 조작을 통한 효율적인 렌더링이 가능 하기 때문이다.

렌더링 과정에서는 Octree Rendering을 통하여 생성된 옥트리를 빠른 속도로 순회하게 되며 각 노드에 대한 후면 절두체 선별 및 시각 절두체 선별등의 처리 작업을 하게 된다. 이후 Hybrid Discriminiation 과정에서 해당 노드를 Threshold 값과 비교하여 점 또는 폴리곤으로 렌더링을 결정 하게 된다. 이후 GPU Accelerate를 통하여 GPU에 최적화된 작업을 처리해준다.

3.2 옥트리 생성

생성된 분자 표면의 메쉬 정보는 옥트리에 의해 각 폴리곤들이 옥트리의 노드에 들어가게 된다. 이 때 트리의 분할 단계를 적절히 조절 해야 한다. 이러한 이유는 첫째, 너무 세분화된 분할을 하면, 작은 폴리곤들을 렌더링하므로, 오히려 속도가 떨어지는 현상이 발생하기 때문이다.

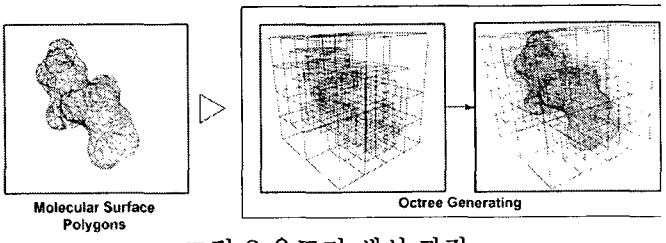

그림 2 옥트리 생성 과정

둘째, 분자 구조의 특성상 폴리곤들은 분자의 색 및 분자구조의 지형적인 특징을 담고 있기 때문에 폴리곤들의 개수를 알맞게 조정해야 한다. 본 시스템에서는 옥트리의 분할 단계를 분자구조를 구성하는 하나의 원자 크기로 분할 될 때까지 분할 단계를 조정한다.

옥트리는 다음과 같이 생성된다. 초기 분자 표면을 포함하는 바운딩 박스의 길이와 원자 1개가 차지하는 평균 공간을 계산하여 옥트리의 분할 단계를 조절한다. 이후 각 축의 중간값을 기준으로 옥트리를 분할 하며, 폴리곤이 포함되지 않은 공간은 효과적으로 제거된다. 그림 2는 분자 표면 메시 데이터가 옥트리로 구성되는 그림이다.

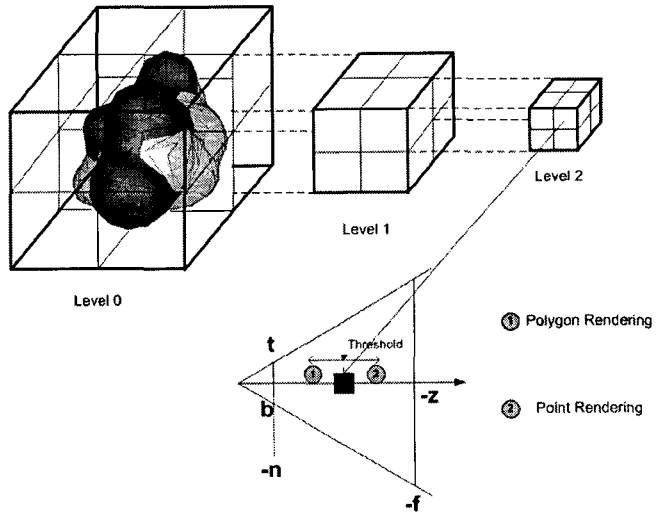

그림 3 옥트리 노드의 렌더링 판별 과정

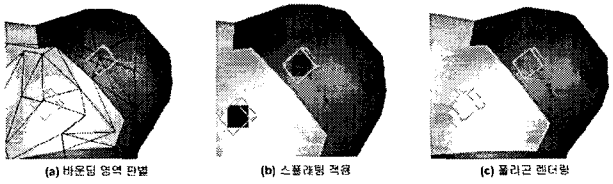

그림 4 Hybrid Point & Polygon 렌더링 결과

우리가 제안하는 방법의 렌더링 향상 정도를 측정 하기 위해서 일반적인 렌더링 방식을 지원 하는 VMD를 비교 대상으로 선정 하였다. 이러한 이유는 VMD가 다양한 분자 표면 렌더링 알고리즘을 지원하고 크로스 플랫폼을 지원하기 때문이다. 성능 측정은 크게 FPS(Frame Per Second) 대한 비교 측정을 하였다. 표2는 4개의 거대 분자에 대한 VMD와 Hybrid Polygon&Point 렌더링 방식을 적용한 VRMMS시스템의 성능측정 결과이다.

표 2 실험 데이터에 대한 VMD와 VRMMS 프레임 결과

PDB CODE	Name	Polygons	시스템 1		시스템2	
			VMD	VRMMS	VMD	VRMMS
1AD0	Ballicus	714,653	10.9	28.7	12.8	32.5
1CX2	Musculus	1,027,024	7.5	23.7	9.2	25.3
1YCE	Na+ ATPase	1,447,440	5.1	18.7	6.4	19.1
1MT5	Hydrolyase	4,000,967	1.2	3.4	2.6	4.4

Frame Per Second

표2에서 알 수 있듯이 기존의 렌더링 방식을 지원하는 VMD 시스템에 비해 우리가 제안한 알고리즘을 지원하는 VRMMS 시스템의 평균 FPS 속도가 2~3배 정도 빠른 것을 알 수 있다.

그림 6은 이 표2의 측정 결과 데이터를 그래프로 나타내었다. 6.(a)는 폴리곤수의 증가에 따른 FPS의 측정 결과를 나타낸 그래프이다. 그래프는 분자의 폴리곤수의 증가에 따른 FPS의 측정 결과를 나타내고 있으며 6.(a)는 표2의 시스템1에서의 측정 결과를 6.(b)는 시스템 2에서의 측정 결과를 나타낸 그림이다. 이 그림에서 알 수 있듯이 기존의 렌더링 방식에 비해서 2배 이상의 속도 향상을 가져 오는 것을 알 수 있다. 또한 고가의 그래픽 카드를 적용한 시스템1 보다 일반적인 그래픽 카드를 SLI를 통한 병렬 환경을 구축한 시스템의 성능이 높게 나오는 것을 알 수 있다.

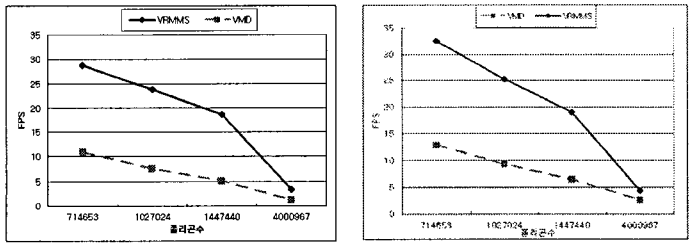

그림 6 실험 측정 결과 그래프

5. 결론 및 향후 연구

본 논문에서는 거대 분자 구조의 표면 렌더링이 기본적으로 가지고 있는 부하문제를 GPU기반의 Hybrid Point & Polygon 기법을 적용하여 해결 하였다. 사용자의 시점에서 화면의 뒷부분이나 가장자리에 위치한 부분들에 많은 폴리곤들이 중첩이 될 때 이를 하나의 점으로 렌더링함으로써 많은 속도 향상을 가져 올 수 있었다. 또한 화면의 앞부분에 위치하는 경우에는 해당 폴리곤들을 바로 렌더링함으로써 선명한 화질을 유지 할 수 있었다.

향후 연구에서는 GPU기반의 포인트 렌더링의 최적화를 통한 렌더링 속도 향상을 개발 하겠다. 특히 병렬 그래픽 카드를 지원하는 SLI환경에서의 GPU가속을 지원함으로써 범용적인 PC환경에서도 워크스테이션 급의 성능을 낼 수 있는 시스템을 구축 하겠다. 또한 보다 시각적으로 자연스러운 분자 렌더링을 위하여 분자 모델에 Adaptive한 Level of Detail 렌더링 알고리즘을 개발하겠다. 이러한 과정을 통하여 실제 대규모 병렬처리 시스템을 사용하여 시뮬레이션 하는 거대 바이러스 물질의 닥킹 실험등을 통한 신약 설계 작업에서, 사용자가 3차원 가상환경으로 거대 분자 모델을 관측 및 조작에 유용하게 사용될 수 있다.

참고문헌

[1] F.M Richards, " Areas, volumes, packing and protein structures," in Annual Review of Biophysics and Bioengineering, 6, pp. 151-176, 1977.

[2] Jee-In Kim, Sungjun Park, Jun Lee, Youngjin Choi, Seunho Jung, " Development of a Gesture-Based Molecular Visualization Tool Based on Virtual Reality for Molecular Docking" , Bulletin of the Korean Chemical Society, Vol.25, No.10, pp 1571-1574, 2004.

[3] Sungjun Park, Jun Lee, Jee-In Kim, " A Molecular Modeling System Based on Dynamic Gestures" , LNCS 3480, pp.886- 895, 2005.

[4] Amitabh Varshney, Frederick P. Brooks, Jr. William V. Wright, " Linearly Scable Computation of Smooth Molecular Surfaces" , IEEE Computer Graphics and Applications, Vol. 14, No.5, pp 19 - 25 Sept 1994.

[5] C.H. Lee and A. Varshney, " Representing Thermal Vibrations and Uncertainty in Molecular Surfaces" , SPIE Conference on Visualization and Data Analysis 2002, San Jose, CA Jan, 2002.

[6] VMD http://www.ks.uiuc.edu/Research/vmd/

[7] S.Rusinkiewicz, M.Levoy, " QSplat: A Multiresolution Point Rendering System for Large Meshes" , Proceedings of SIGGRAPH' 2000, pp 343-352, 2000.

[8] M. Bostch and L.Kobbelt, " High-quality point-based rendering on modern GPUs" , Proc. 11th Pacific Conference on Computer Graphics and Applications , Canmore, Canada, October 2003.

[9] Junfeng Ji, Sheng Li, Xuehui Liu, Enhua Wu, " P-Quadtrees:A Point and Polygon Hybrid Multi-resolution Rendering Approach" , Proceedings of the Computer Graphics International(CGI' 04), pp. 382-385, 2004.

[10] PDB Bank http://www.rcsb.org/