http://dx.doi.org/10.3745/KIPSTA.2012.19A.4.169

그래픽 하드웨어를 이용한 분자용 보로노이 다이어그램 계산

이 정 은[†] · 백 낙 훈[††] · 김 구 진[†††]

요 약

본 논문에서는 주어진 단백질 분자에 대해 3차원 보로노이 다이어그램을 계산하는 알고리즘을 제안한다. 분자는 반경이 서로 다른 구의 집합으로 표현되며, 각 구의 반경은 원자의 반데르바스 (van der Waals) 반경에 대응한다. 보로노이 다이어그램은 3차원 공간을 복셀 (voxel)의 집합으로 분할한 뒤, 보로노이 다이어그램을 포함하는 복셀을 보수적으로 추출함으로써 구성된다. 분자의 계층적 성질을 이용하여 BVH(bounding volume hierarchy)를 구성하고, CUDA 프로그래밍을 통하여 그래픽 하드웨어 가속을 활용함으로써 계산 시간 효율성을 높인다. 공간이 최대 2^{24}개의 복셀로 분할될 경우, 단일 코어 CPU로 구현하는 알고리즘에 비해 계산 속도가 323배 가량 향상 되었다.

키워드 : 보로노이 다이어그램, GPU, 단백질 분자

Voronoi Diagram Computation for a Molecule Using Graphics Hardware

Jung Eun Lee[†] · Nakhoon Baek[††] · Ku-Jin Kim[†††]

ABSTRACT

We present an algorithm that computes a 3 dimensional Voronoi diagram for a protein molecule in this paper. The molecule is represented as a set of spheres with van der Waals radii. The Voronoi diagram is constructed in the 3D space by finding the voxels containing it. For the feasibility of the computation, we represent the molecule as a BVH (bounding volume hierarchy), and our system is accelerated by modern graphics hardware with CUDA programming support. Compared to single-core CPU implementations, experimental results show 323 times faster performance in the computation time, when the space is partitioned into 2^{24} voxels.

Keywords : Voronoi Diagram, GPU, Protein Molecule

1. 서 론

본 논문에서는 단백질 분자에 대해 대화형으로 보로노이 다이어그램을 계산하기 위한 알고리즘을 제안한다. 생화학 분야에서 분자는 보통 반경이 서로 다른 3차원 구의 집합으로 표현되며, 각 구의 중심점과 반경은 분자를 구성하는 원자의 중심점과 반데르바스 반경 (Van der Waals radius)에 일대일로 대응한다[3].

단백질 구조 분야에 Richards[14]가 최초로 보로노이 다이어그램을 소개한 이후 보로노이 다이어그램 및 이와 유사한 공간 분할 기법들은 단백질 구조 분야에서 원자와 잔기 (residue)의 체적 계산, 단백질 패킹(packing)의 모델링, 그리고 단백질의 구조 분석 등을 위해 널리 사용되어 왔다[5, 10, 12]. 보로노이 다이어그램과 유사한 관련 분할 기법들로는 파워 다이어그램 (power diagram), 딜로니 사면체화 (Delaunay tetrahedralization) 등을 예로 들 수 있다. 대부분은 주어진 원자들이 점이라고 가정하거나 거리 계산의 척도로 파워 메트릭 (power metric)을 사용한다[1,2,13,16]. 이러한 가정이 주어지면, 분할 결과가 다면체의 집합으로 근사되고 계산시간의 효율성을 높일 수 있다는 장점이 있다. 반면, 분할 결과가 근사되므로 계산의 정확성은 낮아진다는 단점이 있다.

최근에 들어서 3차원 구의 집합에 대해 유클리디언 거리를 적용하여 보로노이 다이어그램을 계산하는 연구가 Kim et al.에 의해 수행되었다[8-10]. Kim et al.이 제안한 방법은 보로노이 다이어그램을 다면체로 근사하지 않고 대수적 곡면(algebraic surface)으로 계산하는 방법이다. 이들이 제안한 방법은 결과의 정확성은 높아진 반면, 단일 코어 CPU 기반의 구현으로 인해 계산 시간의 효율성은 매우 낮다는 단점을 가진다.

※ 본 논문은 저자들이 포스터로 발표한 참고문헌 [11]의 논문을 확장한 것이며, 2012년도 정부 (교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업(2012-0001755)이며 2012학년도 경북대학교 학술연구비에 의하여 연구되었음.
† 준 회 원 : 경북대학교 전자전기컴퓨터학부 석사과정
†† 종신회원 : 경북대학교 컴퓨터학부 교수
††† 정 회 원 : 경북대학교 컴퓨터학부 부교수
논문접수 : 2012년 2월 21일
수 정 일 : 1차 2012년 3월 22일
심사완료 : 2012년 3월 27일
* Corresponding Author : Ku-Jin Kim(kujinkim@yahoo.com)

본 논문에서는 3차원의 이산 공간에서 반경이 서로 다른 구의 집합에 대해 유클리디언 거리를 적용하여 보로노이 다이어그램을 계산하는 방법을 제시한다. 본 논문에서 제안하는 알고리즘은 공간을 분할하고 그래픽 하드웨어를 이용하여 병렬 처리를 수행함으로써 대화형으로 보로노이 다이어그램을 계산할 수 있다. 주어진 오차 임계값에 의해 공간을 분할하는 복셀의 갯수가 결정되므로, 사용자가 원하는 수준의 정확도를 만족시킬 수 있다. 해석적으로 또는 수치적인 방법으로 보로노이 다이어그램을 계산하는 기존의 방법을 사용하면 수치적 오차에 따라 계산 결과가 발산할 수 있다는 가능성이 있지만, 제안된 알고리즘은 수치적인 오차에 의해 발산할 가능성이 없으므로 오차에 대해 강건하다는 장점을 갖는다.

본 논문의 구성은 다음과 같다. 2절에서는 관련 연구를 살펴보고 3절에서는 GPU 기반 보로노이 다이어그램 계산 알고리즘의 개요를 보인다. 4절에서는 최소 거리를 효율적으로 계산하기 위한 분자의 표현 방식을 제시한다. 5절에서는 실험 결과를 제시하고, 6절에서 결론을 내린다.

2. 관련 연구

점, 선분, 다각형, 3차원 입체 등과 같은 기하학적 원소를 사이트(site)라 할 때, 공간에 주어진 사이트들에 대해 보로노이 다이어그램을 계산할 수 있다. 보로노이 다이어그램은 공간을 특정 성질을 갖는 영역들로 분할하며, 이때 각 영역은 주어진 사이트에 대해 가장 가까운 공간 상의 점들로 구성된다. 각 영역의 경계면은 보로노이 페이스 (Voronoi face)를 구성한다.

단백질 분자에 대한 보로노이 다이어그램은 생화학 분야에서 폴딩 (folding), 원자의 체적 계산, 분자의 인터페이스 (interface), 분자 곡면 (molecular surface) 등의 계산 등을 위해 다양한 용도로 사용된다 [1,2,8,13,16]. 생화학 분야에서는 원자를 주로 점으로 표현하거나 파워 메트릭을 사용함으로써 보로노이 다이어그램을 다면체로 근사하였다.

Kim et al.[9,10]은 반경이 서로 다른 구의 집합에 대해 유클리디언 거리를 적용하여 보로노이 다이어그램을 계산하는 방법을 제안하였다. 이들은 보로노이 다이어그램을 대수적 곡면의 식으로 유도하였으며 이때 위상적으로 발생하는 문제를 해결하기 위하여 몇 가지의 가정을 사용하였다. 예를 들어 보로노이 정점의 차수는 항상 4 이하이며 보로노이 에지는 항상 세 개의 보로노이 페이스의 교차에 위치한다는 가정을 적용하였다[9]. 이들이 제안한 방법은 수학적으로 정확한 결과를 제공하지만, 계산 시간의 효율성이 낮다는 단점을 갖는다.

컴퓨터그래픽스 분야에서는 그래픽 하드웨어의 기능을 활용하여 점의 집합 또는 선분으로 근사된 곡선에 대해 2차원 또는 3차원 보로노이 다이어그램을 계산하기 위한 연구가 수행되었다[4,6,7,15]. 2차원 보로노이 다이어그램 계산을 위해 Hoff et al.[6]은 2차원 점들에 대해 점에서 영역까지의 거리를 근사하는 원뿔을 생성하여 보로노이 다이어그램을 계산하였다.

Fisher and Gotsman[4]은 영상공간(image space)에서 2차원 점들에 대해 k차 보로노이 다이어그램을 계산하는 방법을 제안하였다. 이들은 보로노이 다이어그램을 계산하기 위해 접평면 (tangent-plane)을 계산하는 알고리즘을 사용하였다. Hsieh and Tai[7]은 점, 에지, 삼각형을 포함하는 집합에 대해 3차원 보로노이 다이어그램을 계산하는 방법을 제안하였다. 이들은 사이트로부터 가장 가까운 복셀들이 보로노이 셀 내에 포함되도록 공간을 복셀의 집합으로 분할하였다.

저자들이 아는 바로는 현재까지 반경이 서로 다른 구의 집합에 대해 보로노이 다이어그램을 계산하는 GPU 알고리즘은 없다. 기존의 보로노이 다이어그램 계산을 위한 GPU 알고리즘은 분자를 점의 집합 또는 삼각형의 집합으로 근사해야만 적용이 가능하며, 이러한 경우에는 보로노이 다이어그램 계산의 정확성을 보장할 수 없다.

3. GPU 기반의 보로노이 다이어그램 계산

본 논문에서는 주어진 구의 집합에 대한 보로노이 다이어그램을 구하고자 한다. 입력이 되는 각각의 구 s_i는 원자의 중심과 반데르바스 반경에 대응되는 구의 중심점 $\mathbf{c}_i = (x_i, y_i, z_i)$ 와 반지름 r_i ($r_i > 0$)로 표현된다.

분자에 대응하는 구의 집합이 $S = \{ s_i \mid 1 \le i \le n \}$라 하자. 질의점 (query point) $\mathbf{q} = (x, y, z)$로부터 s_i까지의 거리는 다음과 같이 계산된다.

$$\mathrm{dist}(\mathbf{q}, \mathbf{s}_i) = \mathrm{dist}(\mathbf{q}, \mathbf{c}_i) - r_i = ((x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2)^{1/2} - r_i.$$

구 내부에 속하는 질의점들에 대해서는 이 거리가 음수값이 될 수 있다. 한 개의 구 s_p에 대한 보로노이 영역 (Voronoi region)은 다음과 같이 정의된다.

$$region(s_p) = \{\mathbf{p} \mid dist(\mathbf{p}, s_p) \le dist(\mathbf{p}, s_i), \text{for all } s_i \in S, s_i \ne s_p\}$$

region(s_p)는 구 s_p에 가장 가까운 공간 영역을 나타낸다. 공간 상에서 2개 이상의 영역에 동시에 속하는 점들, 즉 $\mathrm{dist}(\mathbf{q}, s_1) = \mathrm{dist}(\mathbf{q}, s_2)$ 를 만족하는 점 $\mathbf{q}$의 집합이 각 영역의 경계(boundary)를 형성한다.

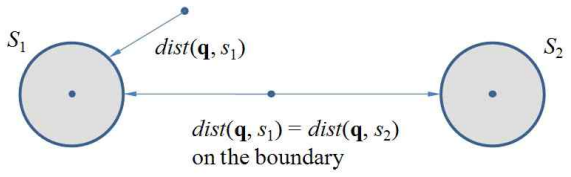


그림 1. 보로노이 영역의 경계
Fig. 1. Boundary between Voronoi regions

본 논문에서는 이산 공간에서 보로노이 다이어그램을 계산한다. 3차원 공간을 균일한 크기를 가진 직육면체 형태의 복셀(voxel)들로 분할하고, 복셀들 중에서 보로노이 영역의 경계면을 포함하는 것들을 추출하여 보로노이 다이어그램을 근사한다.

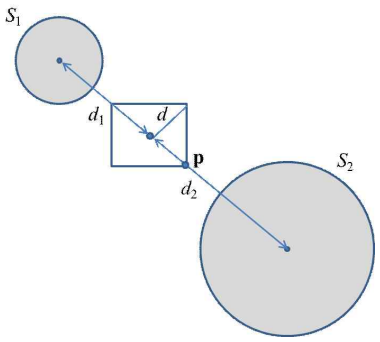


그림 2. 복셀 중심점과 구 간의 거리
Fig. 2. The distance between a voxel center and a sphere

그림 2에서와 같이, 복셀의 중심점 **q**에서 가장 가까운 구 s_1까지의 거리를 d_1이라 하면, d_1 = dist(**q**, $\mathbf{c}_1$) - r_1 이다. 또한, **q**에서 다른 구 s_2까지의 거리를 d_2라 하면 d_2 = dist(**q**, $\mathbf{c}_2$) - r_2 이다. 복셀의 중심점 뿐만 아니라 복셀 내의 모든 점 $\mathbf{p}_i$ 에서 항상 dist($\mathbf{p}_i$, $\mathbf{c}_1$) - r_1 < dist($\mathbf{p}_i$, $\mathbf{c}_2$) - r_2 인 관계가 성립하면, 해당 복셀 전체가 region(s_1)에 속한다고 판정할 수 있다. 만약, d_1, d_2 사이의 차이가 크지 않다면, 복셀의 중심점에서 멀리 떨어진 점들에 대해서는 dist($\mathbf{p}_i$, $\mathbf{c}_1$) - r_1 < dist($\mathbf{p}_i$, $c2$) - r2의 관계가 역전될 수도 있다.

복셀의 직경의 절반이 d 라고 가정하면, 복셀 내의 각 점들이 복셀 중심점에서 멀어질 수 있는 최대 거리는 d 가 된다. 복셀에 포함되면서, 복셀 중심점에서 가장 멀리 떨어진 점**p** 를 가정하면, 각각의 구 s_1, s_2 까지의 거리는 다음과 같이 계산된다 (그림 2 참조).

$$d_1' = \text{dist}(\mathbf{p}, \mathbf{c}_1) - r_1 = \text{dist}(\mathbf{q}, \mathbf{c}_1) - r_1 + d = d_1 + d$$
$$d_2' = \text{dist}(\mathbf{p}, \mathbf{c}_2) - r_2 = \text{dist}(\mathbf{q}, \mathbf{c}_2) - r_2 - d = d_2 - d.$$

특정 구에서 복셀 중심점까지의 거리에 $2d$ 를 더한 거리보다 가까운 다른 구가 존재한다면, 복셀 내의 일부 영역은 다른 보로노이 영역에 속할 가능성이 있다. 이러한 복셀들에 대해서는 보로노이 다이어그램에 속할 가능성이 있다고 판정해야 한다. 우리는 이를 바탕으로, 해당 복셀이 어느 보로노이 영역 또는 보로노이 페이스에 속하는 지를 판정하는 알고리즘을 다음과 같이 설계하였다.

Kernel Program

for each voxel, with its center point at **q** and half of the diagonal length, d, **do**

step 1. calculate $dist(\mathbf{q}, s_i)$ from the voxel center point **q**, for all sphere site s_i.

step 2. get $d_{\min}$ = min { $dist(\mathbf{q}, s_i)$ } and its corresponding nearest sphere site $s_{\min}$.

step 3. count the number n of sphere sites whose $dist(\mathbf{q}, s_i)$ is less than $d_{\min} + 2d$.

step 4. if n = 1, the voxel belongs to $region(s_{\min})$. otherwise, it may belong to the boundary of corresponding sphere sites.

이 알고리즘은 모든 복셀에 대해서 수행되어야 하므로, 단일 코어 CPU를 이용하여 수행할 경우에는 상당한 시간이 소요될 것이다. 제안된 알고리즘은 대화형 처리를 위해, GPU 하드웨어를 이용하였다. 각 복셀에 대한 정보를 CUDA의 텍스처메모리에 저장하고, 각 복셀을 CUDA thread에 할당하여, 병렬처리가 가능한 구조로 다음과 같이 구현하였다.

procedure framework

step 1. subdivide the space into n voxels.

step 2. invoke kernel programs for each voxel

step 3. find the voxels with boundary area

4. 분자에 대한 BVH 표현

단백질 분자는 펩타이드 체인(peptide chain)들로 구성되고, 각 펩타이드 체인은 아미노산들이 연결되어 구성된다. 각 아미노산은 원자들로 구성된다. 각 원자가 반데르 바스(Van der Waals) 반경을 갖는 구로 대응될 때, 분자는 계층적인 성질을 갖는 구의 집합으로 취급될 수 있다. 이에 따라 분자를 BVH (bounding volume hierarchy) 형태로 표현한다.

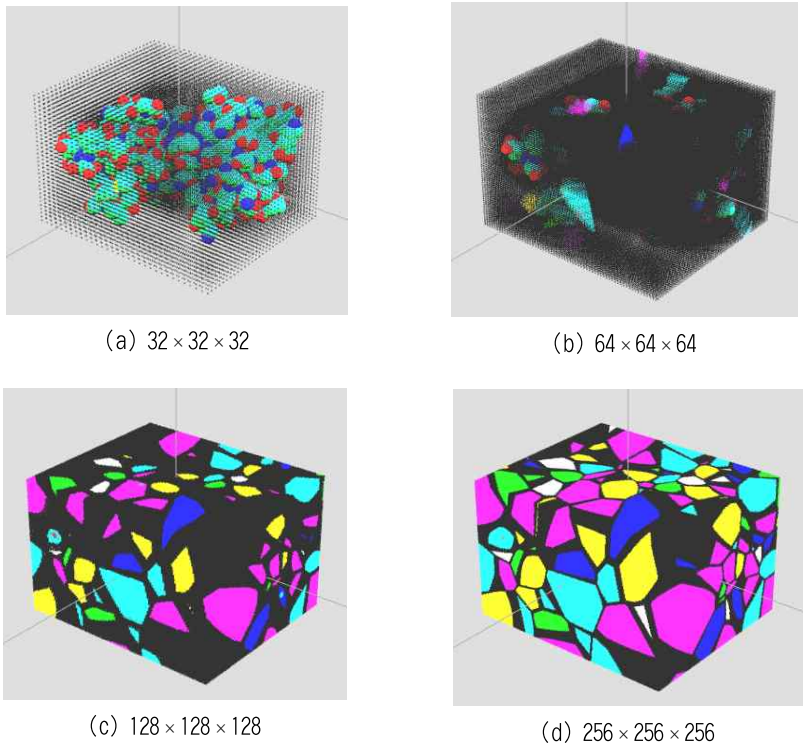
분자에 포함된 펩타이드 체인을 각각 한 개의 트리에 대응시키고, 각 트리의 루트는 대응된 펩타이드 체인에 속한 모든 아미노산의 정보를 갖는다. 루트 노드에 포함된 아미노산들을 중심부에서 이분한 뒤, 이분된 아미노산들을 각각 2개의 자식 노드에 나누어 저장한다. 노드가 한 개의 아미노산을 가질 때까지 연결된 아미노산들을 이분하는 작업을 재귀적으로 수행하며, 이에 따라 리프 노드(leaf node)는 한 개의 아미노산을 가진다. BVH를 구성하는 과정에서 트리 내의 각 노드마다 대응되는 아미노산들에 대한 바운딩 박스 정보를 추가하며, 이 정보는 분자에 대한 거리 계산에 효과적으로 사용된다.

공간 상의 임의의 점 **p**로부터 BVH로 구성된 분자에 대해 최소 거리를 구할 경우, **p**와 루트 노드에 대응되는 바운딩 박스 간의 거리를 먼저 계산한다. 이 거리가 d일 때, 자식 노드들 중에서 바운딩 박스와 **p** 사이의 거리가 d보다 큰 것이 있으면, 이 노드의 자손 (descendent) 노드들에 대해서는 거리 계산을 수행하지 않는다. 이와 같이 트리에 대한 가지 치기 조건을 부과함으로써 거리 계산 시에 효율성을 높일 수 있다.

표 1. CPU 기반 알고리즘과 GPU 기반의 제안된 알고리즘의 실행 시간 비교
Table 1. Comparison of computation time between CPU and GPU based algorithms

복셀의 개수	실행 시간 (msec)			Speed up	
	C.T[1]	G.T[2]	H.T[3]	C.T/G.T	C.T/H.T
$2^5 \times 2^5 \times 2^5$	515	78	32	6.60	16.09
$2^6 \times 2^6 \times 2^6$	4,368	125	54	34.94	80.89
$2^7 \times 2^7 \times 2^7$	33,309	359	152	92.78	219.14
$2^8 \times 2^8 \times 2^8$	270,848	2,293	836	118.12	323.98

[1]C.T : CPU 기반 프로그램의 실행시간
[2]G.T: [11]에서 제시된 프로그램의 실행 시간
[3]H.T: BVH 표현을 이용한 GPU기반 프로그램의 실행시간



(a) 32 × 32 × 32 (b) 64 × 64 × 64

(c) 128 × 128 × 128 (d) 256 × 256 × 256

그림 3. 분자에 대한 보로노이 다이어그램의 계산 예
Fig. 3. Examples of Voronoi diagrams for a molecule

5. 실험 결과

본 논문에서 제시한 알고리즘의 계산 시간 효율성을 보이기 위해 참고문헌 [11]의 알고리즘 및 단일 코어 CPU 용의 알고리즘을 함께 구현하여 비교하였다. 표 1은 Intel® Core™ i5 2.80GHz CPU, 4GB RAM과 nVIDIA GeForce GTX 590, 3GB Video RAM 환경에서의 수행 시간을 보이고 있다. 실험에는 단백질 분자 (PDB id. 2FAE)에서 추출한 총 1,195개의 구를 사용했다.

표 1에서와 같이, CPU 기반의 구현에 비해, 제안된 알고리즘은 최소 16배, 최대 323.98배의 성능 향상을 달성하였다. CUDA 기반의 구현은 대화형으로 사용하는 것이 가능할 정도로 빨랐고, 이들에 대한 예제 화면들이 그림 3에 제시되어 있다.

그림 3에 제시된 스냅샷에서 각 복셀의 중심은 점으로 표현되고, 단백질은 원자의 종류에 따라 색이 다른 구의 집합으로 표현되었다. 보로노이 다이어그램의 후보가 되는 복셀은 짙은 회색으로 나타내고, 보로노이 영역에 속하는 복셀은 영역에 따라 보로노이 다이어그램과 구별되는 임의의 색으로 나타내었다.

6. 결 론

본 논문에서는 단백질 분자를 BVH로 표현한 뒤, 이를 이용하여 GPU 기반으로 분자에 대한 보로노이 다이어그램을 계산하는 알고리즘을 제시하였다. CPU 기반의 구현 결과에 대비하여 사용자가 설정하는 정밀도에 따라 최소 16배에서 최대 323배의 가속이 가능했다.

참 고 문 헌

[1] Y. -E. A. Ban, H. Edelsbrunner and J. Rudolph, "Interface surfaces for protein-protein complexes," Proceedings of the 8th annual international conference on research in computational molecular biology, pp.205-212, 2004.

[2] J. Bernauer, J. Aze, J. Jain and A. Poupon, "A new protein-protein docking scoring function based on interface residue properties," Bioinformatics, Vol.23, No.5, pp.555-562, 2007.

[3] A. Bondi, "van der Waals Volumes and Radii," The Journal of Physical Chemistry, Vol.68, No.3, pp.441-451, 1964.

[4] I. Fischer and C. Gotsman, "Fast Approximation of High-Order Voronoi Diagrams and Distance Transforms on the GPU," Journal of Graphics Tools, Vol.11, No.4, pp.39-60, 2006.

[5] M. Gerstein and F. M. Richards, "Protein Geometry: Distances, Areas, and Volumes," International Tables for Crystallography (Rossmann, M. and Arnold, E., eds.), Vol. F: Crystallography of biological macromolecules, pp.531-539, 2001.

[6] K. E. Hoff III, J. Keyser, M. Lin, D. Manocha and T. Culver, "Fast computation of generalized Voronoi diagrams using graphics hardware," SIGGRAPH'99 Proc. of the 26th annual conference on Computer graphics and interactive techniques, pp.277-286, 1999.

[7] H. -H. Hsieh and W. -K. Tai, "A simple GPU-based approach for 3D Voronoi diagram construction and visualization," Simulation Modelling Practice and Theory, Vol.13, pp.681-692, 2005.

[8] C. -M. Kim, C. -I. Won, Y. Cho, D. Kim, S. Lee, J. Bhak and D. -S. Kim, "Interaction interfaces in proteins via the Voronoi diagram of atoms," Computer-Aided Design, Vol.38, pp.1192-1204, 2006.

[9] D. -S. Kim, Y. Cho and D. Kim, "Euclidean Voronoi diagram of 3D balls and its computation via tracing edges," Computer-Aided Design, Vol.37, pp.1412-1424, 2005.

[10] D. -S. Kim, D. Kim and Y. Cho, "Euclidean voronoi diagrams of 3D spheres: Their construction and related problems from biochemistry," Mathematics of Surfaces 2005, LNCS 3604, pp.255-271, 2005.

[11] K. -J. Kim, J. -E. Lee, N. Baek, "Voronoi diagram computation for protein molecules using graphics hardware (poster)," Proceedings of ACM SIGGRAPH, 2010.

[12] A. Poupon, "Voronoi and Voronoi-related tessellations in studies of protein structure and interaction," Current Opinion in Structural Biology, Vol.14, No.2, pp.233-241, 2004.

[13] N. Ray, X. Cavin, J. -C. Paul and B. Maigret, "Intersurf: dynamic interface between proteins", Journal of Molecular Graphics and Modelling, Vol.23, No.4, pp.347-354, 2005.

[14] F. Richards, "The interpretation of protein structures: total volume, group volume distributions and packing density," Journal of Molecular Biology, Vol.82, pp.1-14, 1974.

[15] G. Rong and T. -S. Tan, "Variants of jump flooding algorithm for computing discrete Voronoi diagrams," Proceedings of the 4thInternational Symposium on Voronoi Diagrams in Science and Engineering, pp.176-181, 2007.

[16] A. Varshney, F. P. Brooks, D. C. Richardson, W. V. Wright and D. Manocha, "Defining, computing, and visualizing molecular interfaces," Proceedings of the IEEE visualization, pp.36-43, 1995.

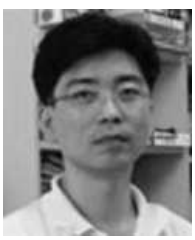
이 정 은

e-mail : highshia@nate.com

2010년 경북대학교 컴퓨터공학과(학사)

2010년~현 재 경북대학교 전자전기 컴퓨터학부 석사과정

관심분야 : 컴퓨터 그래픽스, 계산생물학



백 낙 훈

e-mail : oceancru@gmail.com

1992년 2월 한국과학기술원 전산학과 (공학석사)

1997년 2월 한국과학기술원 전산학과 (공학박사)

2004년 3월~현 재 경북대학교 컴퓨터학부 교수

관심분야 : 모바일 그래픽스, 리얼타임 그래픽스

김 구 진

e-mail : kujinkim@yahoo.com

1990년 이화여자대학교 전자계산학과(학사)

1992년 한국과학기술원 전자계산학과(석사)

1998년 포항공과대학교 컴퓨터공학과(박사)

1998년～2000년 Dept. of Computer Sciences, Purdue University, PostDoc.

2000년～2002년 아주대학교 정보통신전문대학원 BK21 조교수

2002년～2003년 Dept. of Mathematics and Computer Science, University of Missouri-St. Louis, Visiting Assistant Professor

2004년～2007년 경북대학교 컴퓨터학부 조교수

2008년～현 재 경북대학교 컴퓨터학부 부교수

관심분야 : 컴퓨터 그래픽스, 기하모델링, 계산생물학