Edge-tracing algorithm for Euclidean Voronoi diagram of 3D spheres

Deok-Soo Kim* Youngsong Cho† Donguk Kim‡

Abstract

Despite of many important applications in various disciplines from sciences and engineering, Voronoi diagram for spheres in a 3-dimensional Euclidean distance has not been studied as much as it deserves. In this paper, we present an edge-tracing algorithm to compute the Euclidean Voronoi diagram of 3-dimensional spheres in O(mn) in the worst-case, where m is the number of edges of Voronoi diagram and n is the number of spheres. As building blocks for the algorithm, we show that Voronoi edges are conics and they can be represented in rational quadratic Bézier curves. In addition, an appropriate representation of Voronoi faces is also presented.

1 Introduction

Voronoi diagram has been one of the central topics in computational geometry and known for its diverse applications in various scientific and engineering disciplines [12, 15]. While one for ordinary point set has been extensively studied and its properties are well-known in 2 and higher dimensions, Euclidean Voronoi diagram for spheres in 3D has not been explored as much as it deserves even though it may have significant impacts on diverse applications in both science and engineering [1, 6, 11, 16, 18].

It is only very recently that the fast and robust construction of Voronoi diagram for circles in a plane became practical [7, 8]. To get a practical algorithm for the proposed problem, often referred to as an additively weighted Voronoi diagram [12], we have noticed that the idea proposed by Luchinikov et al. is simple yet powerful to get the correct Voronoi diagram of spheres in 3D [10].

2 Terminologies

Let B= b_1, b_2,$\cdots$, b_n be a set of generators where b_i is a three dimensional spherical ball. Hence, $b_i = (c_i, r_i)$ where $c_i = (x_i, y_i, z_i)$ and r_i denote the coordinate of center and the radius of ball, respectively. We assume that no ball is completely contained inside another ball. Associated with each ball b_i, there is a region VR$_i$, called a *Voronoi region* for b_i, where VR$_i = \{p \mid dist(p, c_i) - r_i \leq dist(p, c_j) - r_j, i \neq j\}$. Then, VD(B)= $\{$VR$_1$, VR$_2$,$\cdots$, VR$_n\}$ is called a *Voronoi diagram* for set B. In this paper, the ordinary L_2 distance is used.

Like an ordinary Voronoi diagram, some Voronoi regions corresponding to balls on the boundary of convex hull of B is unbounded. Other regions are bounded by a set of boundary faces, called *Voronoi faces*, where a Voronoi face is defined by two neighboring balls. Note that a face is always a hyperboloid.

A Voronoi face intersects another face to form a *Voronoi edge*. When Voronoi edges intersect, a *Voronoi vertex* is defined. In this paper, we assume that the degree of a vertex is always four. Hence, there is a sphere tangent to four balls centered at the vertex and this *tangent sphere* is said to be *empty*.

3 Computational primitives

3.1 Voronoi vertices

Given four generator balls b_i, i=1,2,3, and 4, an elegant algorithm to compute the sphere(s) tangent to the balls is presented by Gavrilova [4]. She showed that the tangent spheres are computed by solving a quadratic equation where the equation is obtained by an explicit formulation of equi-distant points from four balls. She showed that the solutions consists of groups with none, one and two solutions.

3.2 Voronoi edges

Unless it is a degenerate case, an edge is always defined as a locus of points equi-distant from the surfaces of three surrounding balls. Then, Voronoi edge is the solution of three equations and it can be shown that the Voronoi edges are planar and furthermore they are conics. This fact can be also easily shown that the edge is a spline curve of Dupin cyclide [13, 14].

Since Voronoi edges are conic, they can be exactly represented in a form of rational quadratic Bézier curve once five parameters as follows are known: two end points, tangent vectors at both end points, and a point through which the curve passes [3]. In our problem, two end points are Voronoi vertices. It turns out that the tangent vector at a Voronoi vertex is obtained as a vector equi-angular with three vectors starting from the vertex and ending at the centers of three balls defining the edge. The last parameter, a passing point, can be found as follows. Suppose that we define a plane P

*Department of Industrial Engineering, Hanyang University, Seoul, South Korea, dskim@hanyang.ac.kr

†Voronoi Diagram Research Center, Hanyang University, Seoul, South Korea, ycho@voronoi.hanyang.ac.kr

‡Department of Industrial Engineering, Hanyang University, Seoul, South Korea, donguk@voronoi.hanyang.ac.kr

passing through the centers of three balls. Then, the intersection of P with three balls results into three circles on P. Therefore, the passing point on this plane is the center point of a circle tangent to these three circles, and it is known that this is Apollonius 10^{th} Problem and its solution process is well-described in [8, 17].

Big Brothers Problem

When an edge is either circular or elliptic, we may run into a difficult situation called Big Brothers case. Suppose a small ball is located in-between larger ones. If the center point of the small ball is located on the line passing through the centers of two big-brothers, then the edge is circular. If the center of small ball is a little bit off the line, then the edge becomes now elliptic. This case requires a little more consideration for the construction of Voronoi diagram and is categorized into two sub-cases as edge-connected and edge-disconnected.

In the edge-connected case, it is not necessary to treat the situation in any special way but will be automatically handled via the edge-tracing algorithm. However, if it is an edge-disconnected case, the edge graph of the whole Voronoi diagram is not a single graph but forms a forest and therefore special care should be provided to handle the situation. In our experience, it is better to construct an edge graph for larger balls first and then another edge graph for smaller balls.

3.3 Voronoi faces

Two topologically neighboring balls b_i and b_j define a Voronoi face defined as $|p - c_i| - r_i = |p - c_j| - r_j$. Hence, a Voronoi face is a hyperboloid and its implicit equation can be easily obtained. The principal uses of Voronoi faces are the computations of volumes and boundary areas of Voronoi regions, and perhaps its visualization. Instead of rational parametric representation, we rather use the following approach for the representation of Voronoi faces. Suppose that two balls are transformed so that the center of larger ball is located at the origin and the center of smaller ball is on the positive Z-axis. Then, the Voronoi face between the balls is always a single-valued function w.r.t. X and Y. The boundary of the face, which are Voronoi edges in a rational quadratic Bézier curve form, can be also transformed. If Voronoi faces are represented in this way, both evaluation of a point on the face and testing if a point is on the face are much simpler.

Shown in Figure 1 is an example similar to the one presented in [10]. The Voronoi edges and faces are represented in rational quadratic Bézier curves and implicit surfaces.

4 Edge-tracing algorithm

The basic idea of the edge-tracing algorithm is quite simple as follows. The algorithm first locates a true Voronoi vertex

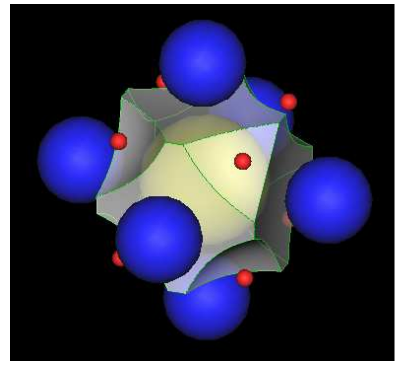


Figure 1: Voronoi region of yellow ball around which there are 14 balls with three different radii.

v_0 by computing an empty tangent sphere defined by four appropriate nearby balls. Provided that v_0 has been found, four edges e_0, e_1, e_2, and e_3 emanating from v_0 can be easily identified and pushed into a stack called Edge-stack. Hence, those edges have v_0 as their starting vertices. After popping an edge from the stack, the algorithm computes the end vertex of the edge. Note that a vertex can be found by computing a tangent sphere from each of $n - 3$ balls plus three balls which define the popped edge and testing if the tangent sphere is empty. If an empty tangent sphere is found, the center of the sphere becomes the end vertex of the popped edge.

Once the end vertex of currently popped edge is found, it is also possible to define three more edges emanating from this new vertex. Hence, three edges are created and the new vertex is used as the starting vertex of three new-born edges. Note that these edges are also pushed into Edge-stack. By following this process until Edge-stack is empty, the computation of Voronoi diagram of a connected graph is completed.

Note that the number of edges m is O(n^2) in the worst-case and O(n) on the average, where n is the number of balls. Even though the idea is simple, designing a correct and efficient algorithm is not so easy at all. Hence, we elaborate the details of the algorithm step by step as follows.

4.1 Initialization

The initialization process computes a tangent sphere from each combination of four balls and tests if the tangent sphere is empty or not. Since there are O(n^4) number of four-ball combinations, the brute-force test of the emptiness for each tangent sphere can be as high as O(n^5) in the worst-case. However, the computation of the initial vertex v_0 can be done rather fast in general since we locate only one such tangent sphere. In addition, it can be accelerated to O(1) on the average if the balls are classified and linked to an appropriate buckets as a preprocessing. Note that this preprocessing to arrange the balls in a bucket can take only O(n) and will be used to speed-up the performance for other operations in a later stage as well.

4.2 End vertex computation

Computing an end vertex of popped edge is equivalent to finding an end-ball associated with the end vertex. Hence, we compute a tangent sphere from each ball in the ball set. After a tangent sphere is computed, its emptiness is tested against all other balls except four balls defined the tangent sphere.

The naive algorithm design for the above takes O(n^2) in the worst-case. However, it can be improved by a more careful design as follows. Given the surrounding balls for a popped edge, we first compute a tangent sphere T_i with an arbitrary candidate ball. Then, we select another candidate ball b_j from the candidate ball set and construct a tangent sphere T_j with b_j and three surrounding balls. If b_j intersects T_i, the current T_i is replaced by T_j. If not, we choose the tangent sphere between T_i and T_j which is closer to the start vertex of popped edge in their angular distances. To locate it, we define an angular distance. Suppose an angle defined among the start vertex, the center of one of the surrounding balls, and the end vertex. Then, given the start vertex and three surrounding balls, the end vertex is determined by the one producing the minimum such angle. Since all balls in the candidate set is scanned only once, this process runs in O(n) in the both worst and average cases and finds the correct end vertex.

4.3 Topologies for vertices and edges

Suppose that a new vertex for an edge is computed. If the vertex is not previously computed, then we can safely use the new vertex to complete the edge. However, if it already exists, then we rather use the existing vertex to complete the definition of the edge since the existing vertex already has associated topology information partially determined. Hence, it is necessary to check if the new vertex is one already computed or not.

For the efficient search for a vertex, we devised a table of vertices already computed and named it Vertex Index Dictionary (VIDIC). An entry in the dictionary consists of indices to four balls defined the vertex and a pointer to the definition of vertex. Hence, the search for an existing vertex can be done. Note that the number of entries in the dictionary can be as many as the number of edges of Voronoi diagram and therefore is O(n^2) in the worst-case. However, if an appropriate ordering among the entries is used, a binary search can be done to take O($\log n$) in the worst case. In addition, we want to mention here that hashing is also possible in this dictionary so that the search can be done in O(1) on the average.

5 Implementation and discussions on time complexity

Shown in Figure 2.a is the Euclidean Voronoi diagram of proteins in PDB with entry code 1BH8 which consists of 1,074 atoms (680 C's, 181 N's, 203 O's, and 10 S's). Once the Voronoi diagram is computed, a ball-and-stick model and the convex-hull of all atoms can be computed in a linear time of faces (Figure 2.b and c). Protein 1BH8 consists of two groups of smaller proteins and shown in Figure 2.d is the surface separating two groups in the protein. Note that this separating surface is a subset of Voronoi faces and can be detected in the linear time of the number of Voronoi faces. It turns out that the interactions between proteins is important [2].

Given an initial vertex to start with, the presented algorithm runs in O(mn) in the worst-case. Note that m is O(n^2) in the worst-case. For each edge, it is necessary to do O(n) scan through all candidate balls once to compute a tangent sphere corresponding to each candidate ball. When a vertex is found, it is also necessary to check if it exists in VIDIC or not and a binary search takes O($\log n$) in the worst-case.

While the numbers of vertices, edges, and faces are O(n^2) in the worst-case, their average numbers are known as linear to the number of balls. In addition, if we devise an appropriate bucket, computing the end vertex can take only O(1) on the average. Searching for a vertex in VIDIC can also be reduced to O(1) on the average if appropriate hashing is used. Note that allocating an appropriate bucket scheme takes O(n) in the worst and the average case.

6 Conclusions

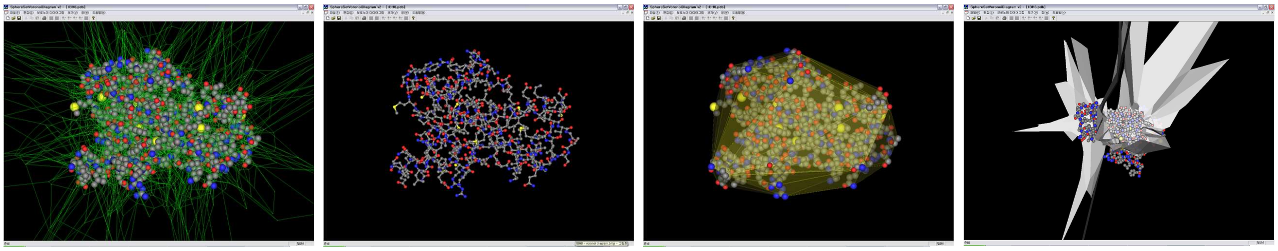
Voronoi diagram for spheres in 3-dimensional space in Euclidean distance has many applications from various disciplines of sciences and engineering. While the Voronoi diagram of spheres can be so important, the algorithm has not been studied as much as it deserves.

In this paper, we have presented an edge-tracing algorithm to compute the 3-dimensional Eulcidean Voronoi diagram of spheres in O(mn) in the worst-case, given an initial vertex to start with, where m and n are the number of edges and spheres, respectively. Note that this initialization takes O(n^5) in the worst-case. However, the algorithm can run in O(n) on the average if appropriate bucket and hashing are used. It is also shown in this paper that the Voronoi edges are conics and can be represented conveniently in rational quadratic Bézier curves. The representation of hyperboloidal Voronoi faces are also presented as the implicit surface with boundary curves in a parametric form.

However, the implementation of the proposed algorithm coping with the exact computation still remains as a challenge.

Acknowledgements

This research was supported by Creative Research Initiative grant from Ministry of Science and Technology, Korea. We thank to Prof. Chee K. Yap and Prof. Kokichi Sugihara for the discussions on the reduction of worst-case time complex-



(a) (b) (c) (d)

Figure 2: 1BH8 consistint of 1,074 atoms (680 C's, 181 N's, 203 O's, and 10 S's) obtained from PDB.

ity. We also thank to an anonymous referee for the constructive opinions.

References

[1] B. Angelov, J. -F. Sadoc, R. Jullien, A. Soyer, J. -P. Mornon and J. Chomilier. Nonatomic solvent-driven Voronoi tessellation of proteins: an open tool to analyze protein folds. *Proteins: Structure, Function, and Genetics*, 49, pages 446–456. 2002.

[2] P. Dafas, D. Bolser, J. Gomoluch, J. Park, and M. Schroeder. Using convex hulls to compute protein interactions from known structures. *Bioinformatics*, (in printing).

[3] G. Farin. *Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide, 4th edition*. Academic Press, San Diego, 1996.

[4] M. Gavrilova and J. Rokne. Updating the topology of the dynamic Voronoi diagram for spheres in Euclidean d-dimensional space. *Computer Aided Geometric Design*, 20, pages 231–242, 2003.

[5] M. Gerstein, J. Tsai, and M. Levitt. The Volume of Atoms on the Protein Surface: Calculated from Simulation, using Voronoi Polyhedra. *Journal of Molecular Biology*, 249, pages 955–966, 1995.

[6] A. Goede, R. Preissner, and C. Frömmel. Voronoi Cell: New Method for Allocation of Space among Atoms: Elimination of Avoidable Errors in Calculation of Atomic Volume and Density. *Journal of Computational Chemistry*, 18(9), pages 1113–1123, 1997.

[7] D. -S. Kim, D. Kim, and K. Sugihara. Voronoi diagram of a circle set from Voronoi diagram of a point set: I. Topology. *Computer Aided Geometric Design*, 18, pages 541–562, 2001.

[8] D. -S. Kim, D. Kim, and K. Sugihara. Voronoi diagram of a circle set from Voronoi diagram of a point set: II. Geometry. *Computer Aided Geometric Design*, 18, pages 563–585, 2001.

[9] S. H. Lee and K. Lee. Partial Entity Structrue: A Compact Non-Manifold Boundary Representation Based on Partial Topological Entities. In *Proc. 6th ACM Symposium on Solid Modeling and Application*, Sheraton Inn Ann Arbor, Michigan, June 6-8, pages 159–170, 2001.

[10] V. A. Luchnikov, N. N. Medvedev, L. Oger, and J. -P. Troadec. Voronoi-Delaunay analyzis of voids in systems of nonspherical particles. *Physical review E*, 59(6), pages 7205–7212, 1999.

[11] J. C. G. Montoro and J. L. F. Abascal. The Voronoi Polyhedra as Tools for Structure Determination in Simple Disordered Systems. *The Journal of Physical Chemistry*, 97(16), pages 4211–4215, 1993.

[12] A. Okabe, B. Boots, and K. Sugihara. *Spatial Tessellations Concepts and Applications of Voronoi Diagram*. John Wiley & Sons, 1992.

[13] M. Paluszny and W. Boehm. General cyclides. *Computer Aided Geometric Design*, 15, pages 699–710, 1998.

[14] H. Pottmann. *personal communication*. 2004. 5.

[15] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, 1985.

[16] F. M. Richards. The Interpretation of Protein Structures: Total Volume, Group Volume Distributions and Packing Density. *Journal of Molecular Biology*, 82, pages 1–14, 1974.

[17] J. Rokne. Appolonius's 10th Problem. *Graphics Gems II* (Edited by J. Arvo), pages 19–24. Academic Press, 1991.

[18] V. P. Voloshin, S. Beaufils and N. N. Medvedev. Void space analysis of the structure of liquids. *Journal of Molecular Liquids*, 96-97, pages 101–112, 2002.