A Plane Sweep Algorithm for the Voronoi Tessellation of the Sphere

Xiaoyu Zheng[1], Roland Ennis[2], Gregory P. Richards[1,3], and Peter Palffy-Muhoray[3]

[1] *Deparment of Mathematical Sciences, Kent State University, Kent, Ohio 44242, USA*

[2] *Pressco Technology, Inc, Cleveland, Ohio 44139, USA*

[3] *Liquid Crystal Institute, Kent State University, Kent, Ohio, 44242, USA*

Abstract

We have extended Fortune's sweep-line algorithm for the construction Voronoi diagrams in the plane to the surface of a sphere. Although the extension is straightforward, it requires interesting modifications. The main difference between the sweep line algorithms on plane and on the sphere is that that the beach line on the sphere is a closed curve. We have implemented this algorithm and tessellated spheres with up to one million sites. The running time for our sweep line algorithm on the sphere is $O(N \log N)$, and the storage space is $O(N)$.

1 Introduction

There are a number of fascinating phenomena in soft condensed matter physics whose description involves nonlinear partial differential equations on a sphere.

Nematic liquid crystals consist of anisotropic molecules, whose orientation may be described by a unit vector $\hat{\mathbf{m}}$ along their symmetry axes. Orientational order in such a system is described by the orientational probability distribution function, $\rho(\hat{\mathbf{m}})$. The unit vector $\hat{\mathbf{m}}$ corresponds to a point on a unit sphere, and so ρ is a scalar function on the sphere. If the system is not in equilibrium, its dynamics is described by time evolution of ρ which is governed by the Smoluchowski equation (cf. [1, 2, 3]), which is of the form

$$\frac{\partial \rho}{\partial t} = -\nabla_{\hat{\mathbf{m}}} \cdot \mathbf{J} = \nabla_{\hat{\mathbf{m}}} \cdot (k \nabla_{\hat{\mathbf{m}}} \rho + \mathbf{F}(\rho)), \tag{1}$$

where $\hat{\mathbf{m}}$ is the unit outer normal vector to the sphere, and the tangential gradient operator $\nabla_{\hat{\mathbf{m}}}$ is defined as

$$\nabla_{\hat{\mathbf{m}}} = (\mathbf{I} - \hat{\mathbf{m}}\hat{\mathbf{m}}) \cdot \nabla. \tag{2}$$

A problem in biophysics involves phase separation of lipids in giant unilamellar vesicles, where below a critical temperature, two coexisting liquid phases exist over a wide range of composition and temperature [4]. These liquid domains exhibit interesting behavior; they collide and coalesce and can form striped domains. Such a system can be modeled by a Cahn-Hilliard formalism [5]. The concentration c is a scalar function on the sphere, its time evolution is given by

$$\frac{\partial c}{\partial t} = D \nabla^2_{\hat{\mathbf{m}}} (-c + c^3 - \gamma \nabla^2_{\hat{\mathbf{m}}} c). \tag{3}$$

which describes conserved order parameter dynamics.In weather predictions, the shallow water equations on sphere [7], which capture most of the characteristics of flow of a thin layer of fluid tangent to the surface, are often used to approximate the flow on the earth's surface.

Since exact analytic solutions usually do not exist, and since approximate analytic solutions do not give a good description of the time evolution of the patterns observed, numerical solutions are generally sought. Candidate strategies are spectral methods [8], which do not require a grid, and finite difference, finite element [6, 7] and finite volume [12] methods, which do. For nonlinear

equations, spectral methods are challenging due to mode coupling. Regular or nearly regular grids are well suited to grid based numerical methods. The grid generated by spherical coordinates is generally not suitable due to the singularity at the poles. It is not possible to construct a regular grid on a sphere with more than 20 points. A variety of grid construction methods exist for the sphere [7, 9, 10, 11]. A random grid can be conveniently constructed by placing points randomly on the sphere, and then using the Voronoi tessellation [13] to construct the grid. Because of its natural simplicity, in this paper, we focus on this method. Finally, we note that the grid obtained via the Voronoi construction can easily be regularized [12].

The Voronoi tessellation associates a discrete set of given points, called sites, with regions of space, called cells, such that all points in the cell associated with a given site are closer to that site than to any other. In addition to generating grids, the Voronoi tessellation can be used to find facilities, such as hospitals or fuel depots, nearest to a given location, to identify the nearest neighbors of a site, as well as for many other applications in biology, physics and computer science. In $2-D$, Voronoi cells are polygons in the plane, each containing the associated site. Edges of Voronoi cells are bisectors of the line joining neighboring sites and consist of points equidistant from them. Vertices of Voronoi cells are intersections of edges and are equidistant from three or more sites. The Voronoi diagram shows the sites and the boundaries of the cells. The dual to the Voronoi diagram is the Delaunay triangulation, where lines connect each site with its nearest neighbors. The Delaunay triangulation is important in the meshing of non-uniform grids in finite element methods, because they minimize the degeneracy of the generated triangles. The extension to higher dimensions is straightforward; in $3-D$, Voronoi cells are polyhedra.

In $2-D$, the simple algorithm which finds the Voronoi tessellation by computing the common intersections of bisectors formed by the site i and all other sites $j \neq i$, requires $O(N^2 \log N)$ time. Algorithms of complexity $O(N \log N)$ exist; these include incremental construction with randomization [14], divide and conquer algorithm [15], and Fortune's sweep line algorithm [16]. The Fortune algorithm uses a sweep line, and generates parabolas defined by the sites and the sweep line. The intersections of neighboring parabolas are used to determine the cell edges.

Although some algorithms, such as the incremental algorithm, can be brought to run in $O(N)$ average time for well distributed sets of sites [17], Fortune's algorithm, which runs $O(N \log N)$ both worst time and average time, is very popular because it is, in general, easier to implement, and because of its aesthetic appeal. It is also important theoretically, since sweep line algorithms represent a fundamental approach to problems in computational geometry.

We show here that the sweep line algorithm for constructing the Voronoi diagram can be extended to manifolds other than the $2-D$ plane; in our case, to the surface of the sphere. The structure of our algorithm is similar to that of Fortune for the tessellation of the $2-D$ plane, but with a different definition of distance, and with interesting new features due to the topology of the sphere. One simplification also arises, since tessellation of the sphere is necessarily on a closed domain. Other algorithms for tessellating the sphere exist [18, 19, 20], but as in the case of the Fortune algorithm in the $2-D$ plane, the one presented here might be easier to implement.

In order to make the paper self-contained, we include all necessary definitions, properties and algorithms. Regarding Fortune's sweep line algorithm [16], we follow the interpretation of de Berg [21].

2 Preliminaries and definitions

Since any sphere can be scaled and shifted, we consider only the tessellation of the unit sphere centered at the origin. We denote the unit sphere as

$$S^2 = \{\hat{\mathbf{p}} \in \mathbf{R}^3 : \|\hat{\mathbf{p}}\|_2 = 1\} \tag{4}$$

where $||\cdot||_2$ is the L^2 norm. The (geodesic) distance between any two points $\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$ on sphere is denoted as $d(\hat{\mathbf{p}}, \hat{\mathbf{q}})$, and

$$d(\hat{\mathbf{p}}, \hat{\mathbf{q}}) = \arccos(\hat{\mathbf{p}} \cdot \hat{\mathbf{q}}). \tag{5}$$

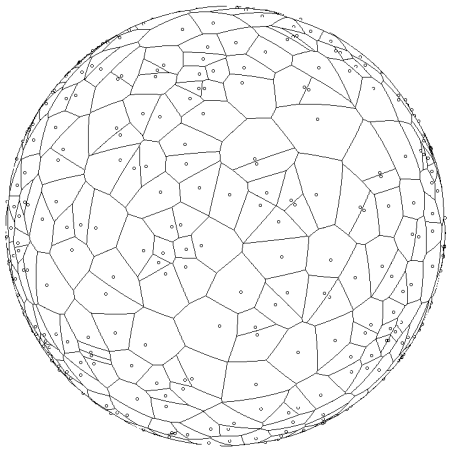

Figure 1: The Voronoi diagram of 500 random sites.

This is the arclength of the shorter segment of the great circle through $\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$, with $\hat{\mathbf{p}}$ and $\hat{\mathbf{q}}$ as endpoints. A region $H \subset S^2$ is convex if a geodesic joining any two points in H is contained in H.

We identify the sites as the set of N distinct points $P := \{\hat{\mathbf{p}}_i\}_{i=1}^N$ on the sphere. We denote the Voronoi diagram of P by $Vor(P)$. We define $V(\hat{\mathbf{p}}_i)$, the Voronoi cell for site $\hat{\mathbf{p}}_i$, to be the set of points on the sphere that are closer to $\hat{\mathbf{p}}_i$ than to any other site:

$$V(\hat{\mathbf{p}}_i) = \{\hat{\mathbf{r}} \in S^2 : \ d\,(\hat{\mathbf{r}}, \hat{\mathbf{p}}_i) < d\,(\hat{\mathbf{r}}, \hat{\mathbf{p}}_j)\,, \ \forall j \neq i\}. \tag{6}$$

The bisector B_{ij} is the set of points equidistant to sites $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}_j$,

$$B_{ij} = \{\hat{\mathbf{r}} \in S^2 : \ d\,(\hat{\mathbf{r}}, \hat{\mathbf{p}}_i) = d(\hat{\mathbf{r}}, \hat{\mathbf{p}}_j)\}. \tag{7}$$

The bisector is a great circle with normal along $\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i$. Each Voronoi cell $V(\hat{\mathbf{p}}_i)$ is a closed convex region on sphere, since it is formed by $N-1$ intersection of bisectors. A Voronoi edge E_{ij} between sites $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}_j$ is the intersection of B_{ij} with the boundary of the Voronoi cell,

$$E_{ij} = \partial V(\hat{\mathbf{p}}_i) \cap B_{ij} = \partial V(\hat{\mathbf{p}}_j) \cap B_{ij}. \tag{8}$$

All edges in the Voronoi diagram on the sphere are circular arcs; segments of great circles. A Voronoi vertex $\hat{\mathbf{v}}_{ij\cdots k}$ shared by sites $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j, \cdots, \hat{\mathbf{p}}_k$ is the point on the boundary of Voronoi cell and equidistant to those sites,

$$\hat{\mathbf{v}}_{ij\cdots k} = \{\hat{\mathbf{r}} \in \partial V(\hat{\mathbf{p}}_i) : \ d\,(\hat{\mathbf{r}}, \hat{\mathbf{p}}_i) = d(\hat{\mathbf{r}}, \hat{\mathbf{p}}_j) = \cdots = d(\hat{\mathbf{r}}, \hat{\mathbf{p}}_k)\}. \tag{9}$$

In general, the vertices of a cell are not in a plane.

We consider the general case when the sites do not all lie on a great circle. We recall that the Euler characteristic for a convex polyhedron with v vertices, e edges and f faces is

$$v - e + f = 2. \tag{10}$$

This can be applied here, we only need to replace the number of faces by number of cells. If N_n is the number of n-sided cells, then, for a nondegenerate Voronoi diagram, where each edge is shared by two, and each vertex by three sites, we have

$$\begin{aligned} \sum_n (N_n n/3 - N_n n/2 + N_n) &= 2, \\ \sum_n (6-n) N_n &= 12. \end{aligned} \tag{11}$$

The average number of edges of the Voronoi cell is

$$\frac{\sum_n nN_n}{N} = 6 - \frac{12}{N}. \tag{12}$$

Thus the average number of edges of Voronoi cells is less than 6, and approaches 6 for as the number of sites becomes large. The total number of vertices is $2N-4$, and the total number of edges is $3N-6$.

A circumcircle on a sphere which passes through the three sites $\hat{\mathbf{p}}_i$, $\hat{\mathbf{p}}_j$, $\hat{\mathbf{p}}_k$ is the intersection of the plane determined by these sites and the sphere. The circumcenter of the circle is defined to be on the sphere, and is equidistant to these three sites, thus it is the intersection of any two of the three possible bisectors. The intersection $\hat{\mathbf{v}}_{ij}^{kj}$ between two bisectors B_{ij} and B_{kj}, is on both great circles, thus orthogonal to both normals, and is given by the cross product of the normals of the respective great circles,

$$\hat{\mathbf{v}}_{ij}^{kj} = \pm \frac{(\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j) \times (\hat{\mathbf{p}}_k - \hat{\mathbf{p}}_j)}{|(\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j) \times (\hat{\mathbf{p}}_k - \hat{\mathbf{p}}_j)|}. \tag{13}$$

There are two such intersections which lie on the two ends of the diameter of the sphere. The radius of the circumcircle is given by

$$r = d\left(\hat{\mathbf{v}}_{ij}^{kj}, \hat{\mathbf{p}}_i\right) = \arccos\left(\hat{\mathbf{v}}_{ij}^{kj} \cdot \hat{\mathbf{p}}_i\right). \tag{14}$$

We note that the radius depends on the choice of the circumcenter $\hat{\mathbf{v}}_{ij}^{kj}$, and may be greater than $\pi/2$. We will discuss later the choice of circumcenter in our algorithm. For the Voronoi diagram $Vor(P)$, a circumcenter $\hat{\mathbf{v}}_{ij}^{kj}$ is a vertex of $Vor(P)$ if and only if the circumcircle is empty, that is, if it does not contain any sites in the interior. The interior of the circumcircle is defined to be the spherical cap of the sphere cut by the plane determined by the three sites, containing the circumcenter $\hat{\mathbf{v}}_{ij}^{kj}$.

3 The Plane-Sweep Algorithm

3.1 Overview

The Voronoi edge has the property that the points it contains are equidistant from two sites. If we define a parabola β_1 as the locus of points that are equidistant from a point $\hat{\mathbf{p}}_1$ and a circumcircle, which we denote as the line L, and parabola β_2 as the locus of points equidistant from a second point $\mathbf{p}_2$ and the same line L, then the intersection between β_1 and β_2 is equidistant from both $\mathbf{p}_1$ and $\mathbf{p}_2$, assuming both sites are on the same side of the line L. Making use of this fact is the central point of the Fortune algorithm; we proceed likewise.

We define the sweep line L on the sphere as the intersection of a plane – the sweep-plane – with the sphere. The normal of the sweep-plane is $\hat{\mathbf{n}}$, we call the point defined by $\hat{\mathbf{n}}$ the north pole. The line L is characterized by the normal $\hat{\mathbf{n}}$, and a parameter ξ, so that

$$L(\hat{\mathbf{n}}, \xi) = \{\hat{\mathbf{r}} : \hat{\mathbf{r}} \cdot \hat{\mathbf{n}} = \cos\xi\}. \tag{15}$$

In our algorithm, the line sweeps the sphere as follows. First from top ($\xi = 0$) to bottom ($\xi = \pi$), and then from bottom ($\xi = \pi$) towards top ($\xi = 2\pi$). It is convenient to imagine that, for $\xi > \pi$, the line is on the inside surface of the sphere. The line starts to sweep from the north pole towards the south pole, with increasing ξ.

Each site $\hat{\mathbf{p}}_i$ above (north of) the sweep line, together with the sweep line, determines a parabola β_i. The boundary of the union of all such parabolas is the front, called the beach line, situated above the sweep line for $\xi \leq \pi$, as shown in Fig. 2. It is easy to show that all points which lie above the beach line are closer to some site $\hat{\mathbf{p}}_i \in P$ above the beach line than to the line L, thus their closest site will not be below the sweep line. Hence points which lie above the beach line have already been incorporated into the diagram, leaving the region below the beach line to be considered. Two neighboring arcs on the beach line intersect, and we call such intersections breakpoints. Each

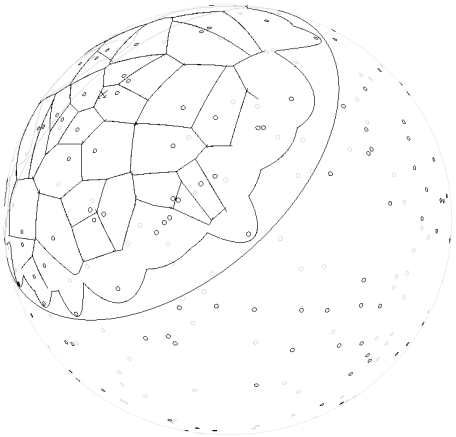

Figure 2: Beach line formed by the boundary of union of the parabolic arcs which divides the whole sphere into two regions. The region above the beach line has been incorporated into the diagram, while the region blow has yet to be considered.

breakpoint is equidistant from two sites, which define the arcs of the parabolas, and hence must lie on some Voronoi edge. Our sweep plane algorithm maintains the beach line as the line L sweeps, and traces the paths of the breakpoints as they move along the edges of the Voronoi diagram. The beach line is a line of latitude, any great circle passing through the north pole $\hat{\mathbf{n}}$ intersects with the beach line exactly twice. The azimuthal angles of the intersections differ by π.

A key difference between our scheme and the planar case is that the beach line on the sphere is a closed curve, while in the planar case it is a curve with open ends. In the planar case, the domain is usually infinite, and the line can sweep indefinitely to finish the diagram. However, the surface of a sphere is a closed domain. When sweep line reaches the south pole, the diagram is not finished. Even though all sites are located above the beach line, the region below the beach line has not yet been processed. We therefore continue to sweep the line upward, figuratively on the inside of the sphere, for values $\xi > \pi$, until the tessellation is complete.

3.2 Parabolas on the sphere

We now derive the equation of parabolas on the sphere. We use the definition that a parabola β is the set of points $\hat{\mathbf{q}}$ that are equidistant, in a geodesic sense, from a point $\hat{\mathbf{p}}$ (focus) and a line L on the sphere. As before, we denote the sweep line by the set of points $\hat{\mathbf{r}}$:

$$L\left(\hat{\mathbf{n}}, \xi\right) = \{\hat{\mathbf{r}} : \hat{\mathbf{r}} \cdot \hat{\mathbf{n}} = \cos\xi = c\}. \tag{16}$$

The points $\hat{\mathbf{q}}$ on the parabola are defined by

$$d(\hat{\mathbf{p}}, \hat{\mathbf{q}}) = d(\hat{\mathbf{R}}, \hat{\mathbf{q}}), \tag{17}$$

or

$$\hat{\mathbf{q}} \cdot \hat{\mathbf{p}} = \hat{\mathbf{q}} \cdot \hat{\mathbf{R}}, \tag{18}$$

where $\hat{\mathbf{R}}$ is the point on the sweep line nearest to $\hat{\mathbf{q}}$. This implies that $\hat{\mathbf{R}} \cdot \hat{\mathbf{n}} = c$, and, as easily shown, that $\hat{\mathbf{R}} = \mu\hat{\mathbf{n}} + \nu\hat{\mathbf{q}}$. It follows that

$$1 = \mu c + \nu(\hat{\mathbf{q}} \cdot \hat{\mathbf{p}}) \tag{19}$$

and

$$c = \mu + \nu(\hat{\mathbf{q}} \cdot \hat{\mathbf{n}}). \tag{20}$$

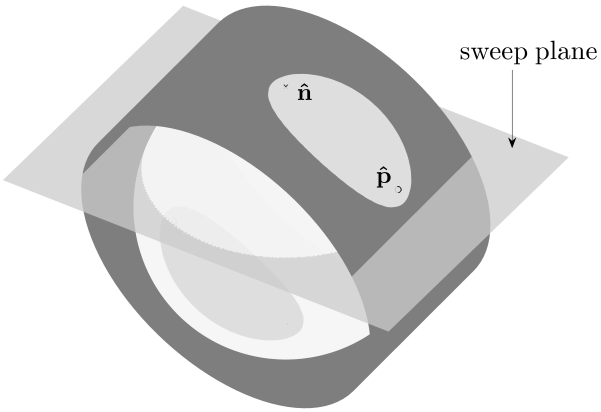

Figure 3: The parabolas are the intersections of the elliptic cylinder and the sphere. The one which encloses the north pole $\hat{\mathbf{n}}$ and the site $\hat{\mathbf{p}}$ is the proper parabola, while the one which encloses the south pole $-\hat{\mathbf{n}}$ and $-\hat{\mathbf{p}}$ is the inverse.

These give

$$\mu = \frac{c(\hat{\mathbf{q}}\cdot\hat{\mathbf{p}}) - (\hat{\mathbf{q}}\cdot\hat{\mathbf{n}})}{(\hat{\mathbf{q}}\cdot\hat{\mathbf{p}}) - c(\hat{\mathbf{q}}\cdot\hat{\mathbf{n}})}, \tag{21}$$

and

$$\nu = \frac{1-c^2}{(\hat{\mathbf{q}}\cdot\hat{\mathbf{p}}) - c(\hat{\mathbf{q}}\cdot\hat{\mathbf{n}})}. \tag{22}$$

Equating the distances from $\hat{\mathbf{q}}$ to $\hat{\mathbf{p}}$ and $\hat{\mathbf{R}}$ gives

$$(\hat{\mathbf{q}}\cdot\hat{\mathbf{p}}) = \mu(\hat{\mathbf{q}}\cdot\hat{\mathbf{n}}) + \nu, \tag{23}$$

and on substitution, after some some algebra, we obtain

$$\frac{(\hat{\mathbf{q}}\cdot(\hat{\mathbf{n}}+\hat{\mathbf{p}}))^2}{2(1+c)} + \frac{(\hat{\mathbf{q}}\cdot(\hat{\mathbf{n}}-\hat{\mathbf{p}}))^2}{2(1-c)} = 1. \tag{24}$$

This is one of our key results. It is interesting to note that the parabola is also an ellipse and a hyperbola on the sphere [22]. Eq. (24) shows that the locus of points $\hat{\mathbf{q}}$ on a sphere, equidistant from a point $\hat{\mathbf{p}}$ and a line (circumcircle) $\hat{\mathbf{r}}\cdot\hat{\mathbf{n}} = c$, is the intersection of the sphere and an elliptical cylinder. The cylinder axis is in the $\hat{\mathbf{n}}\times\hat{\mathbf{p}}$ direction, and the principal axes of the ellipse are along $\hat{\mathbf{n}}+\hat{\mathbf{p}}$ and $\hat{\mathbf{n}}-\hat{\mathbf{p}}$, with lengths $\sqrt{(1+c)/(1+\hat{\mathbf{n}}\cdot\hat{\mathbf{p}})}$ and $\sqrt{(1-c)/(1-\hat{\mathbf{n}}\cdot\hat{\mathbf{p}})}$ respectively. Since the length of one axis is always greater than unity while the other is always less, each parabola consists of two closed curves as shown in Fig. 3. Since Eq. (24) is even in $\hat{\mathbf{q}}$, one curve is the inverse of the other. To distinguish the two curves, we note that the sweep line divides the surface of the sphere into two domains, and that one curve lies entirely in the domain which contains the site or focus. We call this curve the parabola proper, or simply the parabola, and we call the other the inverse parabola, or simply the inverse.

On the sphere, the distance between $\hat{\mathbf{q}}$ and the line $L(\hat{\mathbf{n}},\xi)$ is

$$d(\hat{\mathbf{q}}, L) = \xi - \arccos(\hat{\mathbf{q}}\cdot\hat{\mathbf{n}}). \tag{25}$$

This is valid for $0 < \xi < 2\pi$. For $\xi > \pi$, the shortest path from $\hat{\mathbf{q}}$ to the line L lies partly, in our representation, on the inside of the sphere. Thus the point $\hat{\mathbf{q}}$ on the parabola β satisfies

$$\hat{\mathbf{q}}\cdot\hat{\mathbf{p}} = \cos(\xi - \arccos(\hat{\mathbf{q}}\cdot\hat{\mathbf{n}})). \tag{26}$$

Next, if we parametrize points on the sphere in the canonical form

$$\hat{\mathbf{r}}\,(\theta,\phi) = (\cos\phi\sin\theta\ \hat{\mathbf{x}}, \sin\phi\sin\theta\hat{\mathbf{y}}, \cos\theta\hat{\mathbf{z}}), \tag{27}$$

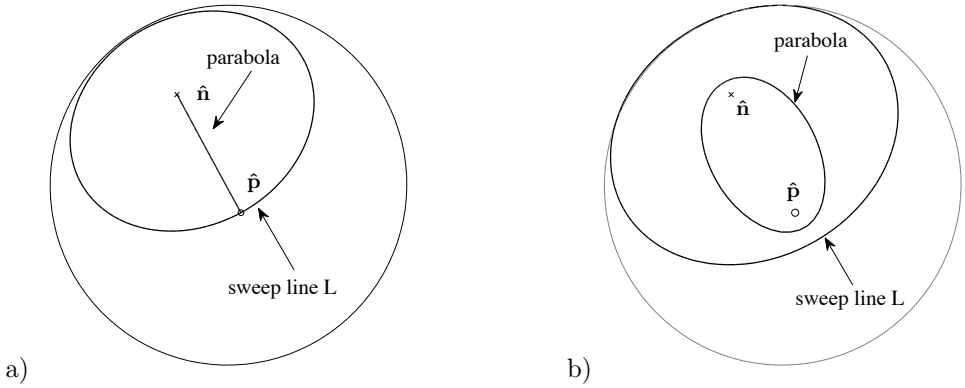

Figure 4: a) The degenerate parabola. when the site $\hat{\mathbf{p}}$ is on the sweep line L, is a part of the great circle pass through the site $\hat{\mathbf{p}}$ and north pole $\hat{\mathbf{n}}$. b) The parabola, determined by a site $\hat{\mathbf{p}}$ and a line L, is a closed curve enclosing the site and north pole $\hat{\mathbf{n}}$.

with $\hat{\mathbf{n}} \equiv \hat{\mathbf{z}}$, then Eq. (26) can be solved to give

$$\tan\theta = \frac{\cos\xi - \cos\theta_p}{\sin\theta_p \cos(\phi - \phi_p) - \sin\xi}, \tag{28}$$

where θ_p and ϕ_p designate the locations of the site $\hat{\mathbf{p}}$ and ϕ, θ those of the point on the parabola, respectively. The parabola is a closed curve, since there is only one θ for each value of ϕ and $\phi + 2n\pi$.

Two parabolas of sites on the same side of the sweep plane intersect at two points which are uniquely identified by their respective azimuthal angles ϕ_{int}, given by

$$\sin(\phi_{int} + \gamma) = \pm\frac{\xi}{\sqrt{a^2 + b^2}}, \tag{29}$$

where

$$\begin{aligned} a &= (\cos\xi - \cos\theta_2)\sin\theta_1\cos\phi_1 - (\cos\xi - \cos\theta_1)\sin\theta_2\cos\phi_2, \\ b &= (\cos\xi - \cos\theta_2)\sin\theta_1\sin\phi_1 - (\cos\xi - \cos\theta_1)\sin\theta_2\sin\phi_2, \\ \xi &= [(\cos\theta_1 - \cos\theta_2)]\sin\xi, \\ \sin\gamma &= \frac{a}{\sqrt{a^2 + b^2}},\ \cos\gamma = \frac{b}{\sqrt{a^2 + b^2}}, \end{aligned} \tag{30}$$

where (θ_1, ϕ_1) and (θ_2, ϕ_2) are the locations of sites $\hat{\mathbf{p}}_1$ and $\hat{\mathbf{p}}_2$. The plus sign is chosen in Eq. (29) if site $\hat{\mathbf{p}}_1$ is encountered first moving counterclockwise on the beach line.

3.3 Beach line and events

The beach line changes its structure due to the appearance of new arcs and the disappearance of existing arcs when the sweep line passes through a special point. The passing of the beach line through such special points is referred to as an "event". As in the planar case, there are only two types of events: A *site event* and a *circle event* and the sweep line is always be advanced from one event to the next, ignoring all intermediate positions. Thus the information describing the parabolic arcs of the beach line is never stored; it is useful only for conceptual purposes. In this section, we consider the general events as normally encountered; degeneracies will be discussed in the subsequent section.

A *site event* occurs when the sweep line reaches a new site, and a new arc appears consequently on the beach line. At this location of the sweep line, the parabola defined by the new site is, simply the portion of the great circle between the site $\hat{\mathbf{p}}$ and the pole $\hat{\mathbf{n}}$, as shown in Fig. 4a. As the sweep line continues to sweep downward, the new parabola broadens, as shown in Fig. 4b. The new arc

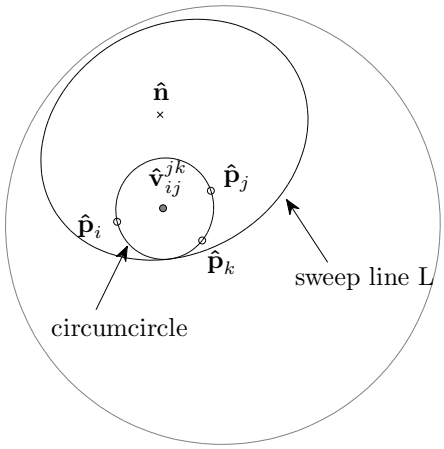

Figure 5: The circumcircle determined by three sites $\hat{\mathbf{p}}_i$, $\hat{\mathbf{p}}_j$, and $\hat{\mathbf{p}}_k$. The sites are ordered clockwise when looking from the circumcenter $\hat{\mathbf{v}}_{ij}^{jk}$ toward the center of the sphere. The circumcircle is tangent to the sweep line L at the lowest point on the circumcircle. The corresponding polar angle is used to sort the circle events.

can appear on the beach line in one of two ways. The first is when the new parabola splits an existing arc on the beach line into two, forming a new arc between them. The second is when the new parabola appears in between two arcs, in this case the new arc is simply inserted between them. This corresponds to one of the degeneracies to be discussed later. At a site event, therefore, two new breakpoints appear. The new breakpoints coincide at first, and move in opposite directions to trace out the same edge. Initially, this new edge is disconnected from the rest of the existing Voronoi diagram. Later, as it becomes longer, it will run into other edges, and become connected to the rest of the diagram.

It can be easily shown that there are at most $2N-2$ arcs of parabolas in the front: each site event splits at most one existing arc into two (except that the second site simply adds one arc due to the periodicity of the beach line) and there are no other ways for an arc appear on the beach line. All site events are known in advance after the sites are sorted by their polar angles. If there are two sites with same polar angle, then they are ordered by their azimuthal angles.

A *circle event* occurs when an existing arc shrinks to a point and disappears. Let β' be the disappearing arc, and β and β'' be the two neighboring arcs before β' disappears. These three arcs are defined by three distinct sites, $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j$,and $\hat{\mathbf{p}}_k$. When arc β' disappears, two breakpoints meet at $\hat{\mathbf{v}}_{ij}^{kj}$ and two Voronoi edges E_{ij}, E_{jk} also meet there. The vertex $\hat{\mathbf{v}}_{ij}^{kj}$ is therefore equidistant from the three sites and also from the line L. This implies that $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j$,and $\hat{\mathbf{p}}_k$ are located on the circumcircle with $\hat{\mathbf{v}}_{ij}^{kj}$ as its circumcenter, and tangent to the sweep line L. The circumcircle doesn't contain any other sites in its interior, else the point $\hat{\mathbf{v}}_{ij}^{kj}$ would be closer to a site than to the line L, in contradiction of $\hat{\mathbf{v}}_{ij}^{kj}$ being on the beach line. It follows that $\hat{\mathbf{v}}_{ij}^{kj}$ is a vertex of the Voronoi diagram, which is traced out by the breakpoints on the beach line.

Circle events are not known *a priori*, and need to be detected dynamically as the algorithm proceeds. For every consecutive set of three sites $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j$,and $\hat{\mathbf{p}}_k$ the circumcircle is computed. As discussed above, there are two centers of this circumcircle. That circumcenter $\hat{\mathbf{v}}_{ij}^{kj}$ is chosen for which the sites $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}_j$,and $\hat{\mathbf{p}}_k$ are ordered clockwise when looking from the circumcenter toward the center of the sphere, as shown in Fig. 5. This corresponds to the positive sign in Eq. (13). The location of the sweepline at the circle event is determined by the polar angle of the lowest point on the circle: $\theta_{circle} = \arccos\left(\hat{\mathbf{v}}_{ij}^{kj} \cdot \hat{\mathbf{n}}\right) + \arccos(\hat{\mathbf{v}}_{ij}^{kj} \cdot \hat{\mathbf{p}}_i)$. A circle event needs to be created only when the lowest point on the circle is below the sweep line L , that is, when $\theta_{circle} > \xi$.

Although the polar angle for a circumcenter $\hat{\mathbf{v}}_{ij}^{kj}$ is always less than or equal to π, the position

of the corresponding circle event θ_{circle} can have a polar angle which is greater than π. In order to process such circle events, the sweep line L has to move up from the bottom, on the inside of the sphere. The beach line however stays on the outside of sphere during the process, because at each circle event the parabolic arcs on the beach line disappear at the location of the circumcenters, which are located on the outside of the sphere. The algorithm stops when only two parabolic arcs remain on the beach line; then no more circle events can be created, and all vertices and their associated sites have been determined. The two breakpoints at which the two parabolic arcs on the beach line intersect eventually merge at a vertex that has already been incorporated into the diagram. At this point, the tessellation is complete.

3.4 Data Structures

- Circle events and site events are stored in two separate priority queues. To implement the priority queues, we use heaps. All events are sorted in lexicographic order: if $\hat{\mathbf{e}} = \hat{\mathbf{x}}\cos\phi\sin\theta + \hat{\mathbf{y}}\sin\phi\sin\theta + \hat{\mathbf{z}}\cos\theta (\phi \in [0, 2\pi))$, then

$$\hat{\mathbf{e}}_1 < \hat{\mathbf{e}}_2 \quad \text{if} \quad (\theta_1 < \theta_2) \quad \text{or} \quad (\theta_1 = \theta_2)\,\&\,(\phi_1 < \phi_2)\,. \tag{31}$$

  For a site event, we simply store the site location itself. For a circle event, we store its lowest point and the circumcenter, with a pointer to the beach line (see below) that represents the disappearing arc.

- We represent the beach line by a skiplist [23]. It can also be represented by a balanced search tree; we choose the skiplist because it is simple to use, and provides search time of the same order, $O(\log N)$. The highest level of the skiplist is set to be 20, corresponding to 2^{20} sites. Each level of the skiplist is a circularly-doubly-linked list, where the lowest level represents the whole beach line. Each node in the skiplist represents an arc β on the beach line. However, we don't explicitly store the parabolic arcs, instead we store the site that gives rise to the parabolic arc. At the lowest level, each node has pointers to its left and right arcs; at higher levels, the left and right pointers may point to arcs several arcs away to the left and to the right. Since the circularly linked list starts and ends at same position, we need to find a position where we start to search through the skiplist. We define the reference position to be the arc of the first site in the site queue. If it disappears from the beach line at some point, then we use its right neighbor arc to be the new reference position. Each node also stores pointers to a node in the circle event queue, namely, the node that represents the circle event at which the arc β will disappear. The pointer can be nil if no circle event exists, or if such circle event has not been detected yet.

- The partial Voronoi diagram that has been constructed above the beach line is stored in a doubly-connected edge list $\mathcal{D}$. In the planar case, care has to been taken, since the some edges are unbounded, and often a big bounding box is added to enclose all sites in it. In the spherical case, we don't have such a difficulty because all edges are constrained to be on the sphere, thus are bounded. In the doubly-connected edge list, the Voronoi edges are represented by half edges. Each half edge is represented by its starting point, pointers to the cell and to its proceeding, following half edges, and its twin in the neighboring cell. The half edges are ordered counterclockwise in each cell. In a Voronoi cell $V(\hat{\mathbf{p}}_i)$, the vertex $\hat{\mathbf{v}}_j$ represents the half edge HE_j, the vertex $\hat{\mathbf{v}}_k$ represents half edge HE_k, and if half edge HE_k is the next half edge to HE_j, then we must have

$$(\hat{\mathbf{v}}_j \times \hat{\mathbf{v}}_k) \cdot \hat{\mathbf{p}}_i > 0. \tag{32}$$

3.5 Algorithm

The sweep line algorithm consists of simulating the growth of the beach line as the sweep line moves first down, outside then up, inside, the sphere. The beach line formed by the parabolas changes its shape continuously during the line sweep. As with in the case of the plane-sweep algorithm, we are

only interested in discrete "events" when there is change in the topology of the Voronoi diagram and structure of the beach line, and skip all intermediate steps.

Input: A set $P := \{\hat{\mathbf{p}}_i\}_{i=1}^{N}$ of N distinct points on the sphere.
Output: The Voronoi diagram $Vor(P)$ given in a doubly-connected edge list $\mathcal{D}$.

1. Sort the sites in P and put them in the event queue $\mathcal{Q}_{site}$; Initialize the empty event queue $\mathcal{Q}_{circle}$; initialize the empty skiplist, and a doubly connected edge list $\mathcal{D}$.
2. While $\mathcal{Q}_{site}$or $\mathcal{Q}_{cirlce}$ is not empty
3. if $\mathcal{Q}_{site}$ is not empty, and the site $\hat{\mathbf{p}}_i$ is before the events in the $\mathcal{Q}_{circle}$
4. then HandleSiteEvent($\hat{\mathbf{p}}_i$), and remove the node from $\mathcal{Q}_{site}$
5. else HandleCircleEvent(γ), where γ is a node in the skiplist that representing the arc that will disappear
6. Traverse the half-edges of the doubly-connected edge list to add cell records and pointers to and from them.

The Procedures to handle the events are defined below.
HandleSiteEvent($\hat{\mathbf{p}}_i$)

1. If the skiplist is empty, insert $\hat{\mathbf{p}}_i$ into it, and set the previous and next pointers to point to itself. If the skiplist only contains one node, then simply append the $\hat{\mathbf{p}}_i$ to it, and reset all pointers correspondingly. Otherwise, continue with steps 2-4.
2. Search the skiplist from the reference position for the arc α which will intersect with the great circle through $\hat{\mathbf{p}}_i$ and north pole $\hat{\mathbf{n}}$. Assume that the two end points of α are (θ_1, ϕ_1) and (θ_2, ϕ_2) such that points on α with azimuthal angle which lies between ϕ_1 and ϕ_2. If
$$\phi_1 \le \phi_i \le \phi_2, \text{ for } \phi_1 < \phi_2 \tag{33}$$
or
$$\phi_i \le \min(\phi_1, \phi_2) \text{ or } \phi_i \ge \max(\phi_1, \phi_2), \text{for } \phi_1 > \phi_2 \tag{34}$$
then the arc is found. If the arc α has a pointer to a circle event in the $\mathcal{Q}_{circle}$, then the circle event is a false alarm, and is removed from $\mathcal{Q}_{circle}$.
3. Duplicate arc α and insert the new arc for $\hat{\mathbf{p}}_i$ between them in the skiplist. Set the previous and next pointers appropriately. Suppose that prior to the insertion, the beach line sequence was
$$..., \hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, \hat{\mathbf{p}}_j, \hat{\mathbf{p}}_3, \hat{\mathbf{p}}_4, ... \tag{35}$$
The insertion of $\hat{\mathbf{p}}_i$ splits the arc associating $\hat{\mathbf{p}}_j$ into two, denoted $\hat{\mathbf{p}}'_j$ and $\hat{\mathbf{p}}''_j$, involving the same site, $\hat{\mathbf{p}}_j$, the new sequence will be
$$\cdots, \hat{\mathbf{p}}_1,\ \hat{\mathbf{p}}_2,\ \hat{\mathbf{p}}'_j,\ \hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}''_j,\ \hat{\mathbf{p}}_3,\ \hat{\mathbf{p}}_4,\ \cdots \tag{36}$$
4. Check the triple of consecutive arcs $\hat{\mathbf{p}}_2, \hat{\mathbf{p}}'_j, \hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}_i, \hat{\mathbf{p}}''_j, \hat{\mathbf{p}}_3$ for valid circle events. If found, insert the circle event(s) into $\mathcal{Q}_{circle}$ and add pointers between the node(s) in the skiplist and in $\mathcal{Q}_{circle}$.

HandleCircleEvent(γ)

1. Let $\hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_j,\ \hat{\mathbf{p}}_k$ be the three sites that generate this circle event. Delete the node for $\hat{\mathbf{p}}_j$ representing the disappearing arc from the skiplist and remove the circle event from the $\mathcal{Q}_{circle}$; Suppose that prior to the deletion, beach line sequence was
$$\cdots,\ \hat{\mathbf{p}}_1,\ \hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_j,\ \hat{\mathbf{p}}_k,\ \hat{\mathbf{p}}_2,\ \cdots \tag{37}$$
after the event, the sequence this becomes
$$\cdots,\ \hat{\mathbf{p}}_1,\ \hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_k,\ \hat{\mathbf{p}}_2,\ \cdots \tag{38}$$

2. Add the circumcenter of the circumcircle causing the circle event as a vertex record to the doubly-connected edge list $\mathcal{D}$. Create three half-edge records starting with the new vertex in three cells corresponding to $\hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_j,\ \hat{\mathbf{p}}_k$.

3. Remove all circle events associated with the triples $\hat{\mathbf{p}}_1,\ \hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_j$ and $\hat{\mathbf{p}}_j,\ \hat{\mathbf{p}}_k,\ \hat{\mathbf{p}}_2$ from $\mathcal{Q}_{cirlce}$, this can be done by using pointers from the predecessor and the successor of $\hat{\mathbf{p}}_j$ in the skiplist. Check two new triples of consecutive arcs involving both former left and right neighbor arcs of α, that is, $\hat{\mathbf{p}}_1,\ \hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_k$ and $\hat{\mathbf{p}}_i,\ \hat{\mathbf{p}}_k,\ \hat{\mathbf{p}}_2$ for valid circle events. If found, insert the circle event(s) into $\mathcal{Q}_{circle}$.

3.6 Degenerate cases

The degeneracies are very similar to those in the planar geometry. The first degenerate case occurs when the first two or more sites have the same polar angle. For example, such sites may be $\hat{\mathbf{p}}_1$ and $\hat{\mathbf{p}}_2$. Special care needs to be taken here since there will be no parabolic arc above $\hat{\mathbf{p}}_2$ yet. Here the arc for $\hat{\mathbf{p}}_2$ is simply appended to the beach line, which is the parabola for $\hat{\mathbf{p}}_1$. These two arcs intersect at the north pole $\hat{\mathbf{n}}$**.** A new vertex record is created, as well as two new half edge records.

A second kind of degeneracy occurs when either two circle events or a circle event and a site event coincide. This corresponds to the case when more than three sites are equidistant from a single point on the sphere. This can occur, for example, when sites are arranged on circles. In such cases, nothing special needs to be done, the events can be processed in arbitrary order.

Degeneracies can also occur when handling a site event, when the new parabola initially intersects two parabola arcs simultaneously. In such a case, one single arc is not split into two, instead, a vertex record is created at the location of the breakpoints of the two parabolic arcs and three half-edge records are created for the three related cells.

3.7 Running time and storage

The total running time is $O(N \log N)$, and the required total storage space is $O(N)$.

Handling each site event requires searching for the arc where the new parabolic arc can be inserted; this requires time $O(\log N)$ using the skiplist. Inserting the arc into the beach line requires constant time $O(1)$. Handling each circle event requires insertion of the vertex and half edge records into the double-connected edge list, which again requires constant time $O(1)$. Checking for the circle events requires constant time $O(1)$ as well. There are N site events and $2N-4$ circle events, thus the total running time is $O(N \log N) + O(N)$. We have implemented the algorithm in Fortran 90, and compiled it using Intel Fortran Composer XE. Programs have been executed on an Intel Core2Quad processor (2.66 GHz). The data shows the $O(N \log N)$ running time, as shown in Fig. 6. We made the comparison of running times with the available algorithm Stripack [19], which implements the Delaunay triangulation first and then constructs the Voronoi tessellation through its dual. The comparison was made at the end of the Voronoi construction; it shows that our algorithm is much faster than Stripack. We note that our code was not optimized, so it is expected that the running time can improved.

All data structures require the storage space $O(N)$. The beach line has at most $2N-2$ arcs, and the highest level in the skiplist is set to be 20. Since all circle events in the queue are associated with sites in the beach line, the maximum length of the circle event queue is $2N-2$. The site event queue has N nodes. The expected number of edges in the Voronoi diagram is $3N-6$, thus doubly-connected edge list $\mathcal{D}$ contains $6N-12$ half edges.

The main results of this work were presented at conferences, including *Emerging Topics in Dynamical Systems and Partial Differential Equations*, May 31-Jun 4, 2010, Barcelona, Spain (http://www.dspdes2010.org/admin/files/fileabstract/a510.pdf) and the *Annual SIAM Meeting*, July 12-16, 2010, Pittsburgh, PA (http://www.siam.org/meetings/an10/An10abstracts.pdf).

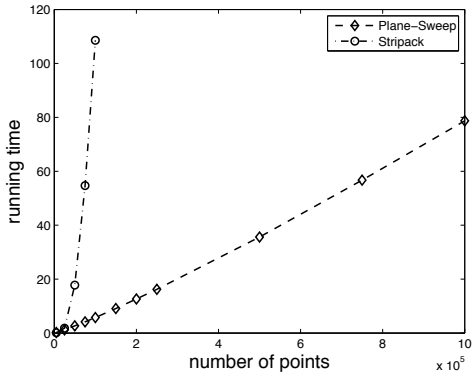

Figure 6: Comparison of running time of the plane-sweep algorithm and Stripack.

4 Conclusion

We have successfully extended Fortune's sweep-line algorithm for the construction Voronoi diagrams in the plane to the surface of a sphere. As in the case of the Fortune algorithm in the plane, we considered the problem where the sites are distinct points. Although the extension is straightforward, it requires interesting modifications. The distance used in this algorithm is the geodesic distance, and the equations of parabolas and circles are derived accordingly. The main difference between the sweep line algorithms on plane and on the sphere is that that the beach line on the sphere is a closed curve. Surprisingly, the sweep line needs to sweep from the top to the bottom on the outside of the sphere, and then back, towards top, on the inside of the sphere. We have implemented this algorithm and tessellated spheres with up to one million sites. We have verified the structure of the tessellations by confirming that all sites have neighbors, all neighbor indices are consistent, all cells are convex, and the sum of the areas of the cells equals 4π.[1] The running time for our sweep line algorithm on the sphere is $O(N \log N)$, and the storage space is $O(N)$.

We were unaware, up to the time of publication, of the work on the same topic [24], presenting essentially the same ideas.

Acknowledgements We acknowledge support by the National Science Foundation under DMS-0908470.

References

[1] B. Bird, R.C. Armstrong, and O. Hassager. Dynamics of polymeric liquids, **1**(2). John Wiley and Sons, New York, 1987.

[2] M. G. Forest and Q. Wang. Monodomain response of finite-aspect ratio macromolecules in shear and related linear flows. Rheol. Acta, **42**,20–46, 2003.

[3] H. Zhang, P. Zhang. Stable dynamic states at the nematic liquid crystals in weak shear flow. Physica D: Nonlinear Phenomena, **232** (2), 156–165, 2007.

[4] S. L. Veatch and S. L. Keller. Separation of liquid phases in giant vesicles of ternary mixtures of phospholipids and cholesterol. Biophysical Journal, **85**(5), 3074–3083, 2003.

[5] J. W. Cahn and J. E. Hilliard. Free energy of a nonuniform system 1. interfacial free energy. Journal of Chemical Physics, **28**(2),258–267, 1958.

[1]The picture of the Voronoi tessellation of a sphere with one million sites is available at http://mpalffy.lci.kent.edu/voronoi

[6] J.R. Baumgardner and P. O. Frederickson, Icosahedral Discretization of the Two-Sphere, SIAM J. Numer. Anal. 22, 1107-1115, 1985.

[7] T. Heinze, A. Hense, The shallow water equations on the sphere and their Lagrange-Galerkin-solution, Meteorology Atmospheric Phys. **80** (1-2), 129–37, 2002.

[8] R. Zhou, M. G. Forest, Q. Wang, Kinetic structure simulations of nematic polymers in plane Couette cells, I: The algorithm and benchmarks, SIAM Multiscale Model. Simul., **3** (4), 853-870, 2005.

[9] C. Ronchia, R. Iaconoa and P. S. Paoluccib, The cubed sphere : A new method for the solution of partial differential dquations in spherical geometry, Journal of Computational Physics, **124** (1), 93–114, 1996.

[10] G. Stuhne and W. Peltier, New icosahedral grid-point discretizations of the shallo water equations on the sphere, J. Comput. Phys., **148**, 23¨C58, 1999.

[11] A. Layton, Cubic spline collocation method for the shallow water equations on the sphere, J. Comput. Phys., **179**, 578–592, 2002.

[12] Q. Du, M.D. Gunzburger, L.Ju, Voronoi-based finite volume methods, optimal Voronoi meshes, and PDEs on the sphere, Comput. Methods Appl. Mech. Engrg., 192, 3933–3957, 2003.

[13] A. Okabe, B. Boots, K.Sugihara, Spatial Tessellations : Concepts and Applications of Voronoi Diagrams. New York: John Wiley & Sons, Incorporated, 2000.

[14] L.J. Guibas, D.E. Knuth and M. Sharir, Randomized incremental construction of Delaunay and Voronoi diagrams, Algorithmica **7** (1-6), 381–413, 1992.

[15] M.I. Shamos and D. Hoey. Closest-point problems. In Proc. 16th Annu. IEEE Sympos. Found. Comput. Sci., 151-162, 1975.

[16] S.J. Fortune, A sweepline algorithm for Voronoi diagrams. Algorithmica, 2: 153–174, 1987.

[17] T. Ohya, M. Iri, and K. Murota. Improvements of the incremental method for the Voronoi diagram with computational comparison of various algorithms. J. Oper. Res. Soc. Japan, 27(4):306–336, 1984.

[18] J. M. Augenbaum and C. S. Peskin. On the construction of the Voronoi mesh on the sphere. Journal of Computational Physics, **59**, 177–192, 1985.

[19] R.J. RENKA, Algorithm 772: STRIPACK: Delaunay triangulation and Voronoi diagram on the surface of a sphere, ACM Transactions on Mathematical Software, **23** (3), 416–434, 1997.

[20] H. Na, C. Lee, O. Cheong, Voronoi diagrams on the sphere, Computational Geometry, **23** (2), 183–194, 2002.

[21] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry: Algorithms and Applications, Second, revised edition, Springer, New York, 2000.

[22] G.S. Skyes and B. Peirce, Spherical conics, Proceedings of the American Academy of Arts and Sciences, **13**, 375-395, 1877.

[23] W. Pugh, Skip lists: a probabilistic alternative to balanced trees. Communications of the ACM, **33** (6), 668–676, 1990.

[24] J. Dinis and M. Mamede, Sweeping the sphere, 2010 International Symposium on Voronoi Diagrams in Science and Engineering, 151–160, 2011.