An Algorithm for the Generation of Voronoi Diagrams on the Sphere Based on QTM

Jun Chen, Xuesheng Zhao, and Zhilin Li

Abstract

In order to efficiently store and analyze spatial data on a global scale, the digital expression of Earth data in a data model must be global, continuous, and conjugate, i.e., a spherical dynamic data model is needed. The Voronoi data structure is the only published attempt (Wright and Goodchild, 1997) and the only possible solution currently available (Li et al., 1999) for a dynamic GIS. However, the complexity of the Voronoi algorithm for line sets and area sets in vector mode limits its application in a dynamic GIS. So far, there is no raster-based Voronoi algorithm for objects (including points, arcs, and regions) on a spherical surface.

To overcome this serious deficiency, an algorithm for generating a spherical Voronoi diagram is presented, based on the O-QTM (Octahedral Quaternary Triangular Mesh). The principle of the dilation operation in mathematical morphology is extended to the spherical surface. A method is developed for a spherical distance transformation based on the QTM. A detailed algorithm is also presented. This algorithm can handle points, arcs, and area features on a spherical surface. Tests have shown that the computational time consumption of this algorithm with points, arcs, and areas is equal and proportionate to the levels of the spherical surface tessellation; and the difference (distortion) between the great circle distance *and the QTM cells distance is slightly related to spherical distance (not as the raster dilation on a planar surface), and is mainly related to the locations of the generating points.*

Introduction

In order to effectively store and analyze spatial data on a global scale, the digital expression of global data in a computer must be globally seamless (not broken into zones), continuous (unlike latitude/longitude), and conjugate (often updated in local regions) (Lukatela, 1987; Gold and Mostafavi, 2000). This leads to the need to construct a spherical dynamic data model. It has been argued by Li *et al.* (1999) that the Voronoi data structure is the only possible solution currently available to a dynamic GIS. This view is similar to that of Wright and Goodchild (1997), who point out that the Voronoi methods are the only published attempts that we are aware of that are applicable to a dynamic GIS. This is because Voronoi diagrams (VD) have many excellent properties in spatial analysis (Gold, 1992; Edwards, 1993; Chen *et al.*, 2001; Li *et al.*, 2002), dynamic operation (e.g., adding or deleting objects without destroying the bubble structure of the cells) (Roos, 1991; Gold, 1992; Gold and Condal, 1995; Gold and Mostafavi, 2000), and computational geometry (Aurenhammer, 1991; Okabe *et al.*, 2000), etc.

So far, the Voronoi diagram on a spherical surface has been applied to a few areas, such as global spatial indexing (Lukatela, 1987; Lukatela, 1989; Lukatela, 2000), sphere interpolation (Watson, 1988; Watson, 1998), dynamic operation (Gold, 1997a, 1997b; Gold and Mostafavi, 2000), etc. For example, Lukatela (1987) set up a digital geo-positioning model and developed an operational software package that provides geometrical and geo-relational functions to applications that manipulate spatial objects. A Voronoi tessellation is used as a base for a highly efficient indexing system to increase the speed of data manipulation (Lukatela, 1989) (Figure 1a). Watson (1988; 1998) developed a MODEMAP system and used a point-set Voronoi diagram for interpolation on a spherical surface (Figure 1b). In the MODEMAP, point density is calculated as the inverse of the area of the Voronoi neighborhoods. By using these densities, a density surface using natural neighbor interpolation can be interpolated on the sphere. Pixel colors are assigned according to the height of that density surface. Gold and Mostafavi (2000) have attempted to develop a global dynamic data structure with a Voronoi diagram. In such a structure, the Voronoi diagram acts as a basic data model to dynamically maintain spatial relationships.

It should be clear by now that the Voronoi diagram is a type of spatial data model which has become increasingly important, and that a lot of effort has been spent on the development of the algorithms. However, most of the algorithms are in vector mode and based on point sets on a planar surface. The algorithms for generating the Voronoi diagrams of line and area sets in vector mode are very complex. This complexity has greatly limited the application of the Voronoi data model in a dynamic GIS. On the other hand, on a spherical surface, there are only a few algorithms for generating Voronoi diagrams for point-sets, and none for arc-sets (or curve-face sets).

In raster mode, a Voronoi diagram for entities can be formed easily, and some approximation algorithms have been developed (Dehne *et al.*, 1989; Okabe *et al.*, 1992; Embrechts and Roose, 1996; Li *et al.*, 1999) for objects on a planar surface (including lines and areas). Judging from the available papers, however, no algorithm for generating Voronoi diagrams in raster mode has been developed for objects on a spherical surface.

This paper aims to present a QTM (Quaternary Triangular Mesh)-based method for the computation of a spherical Voro-

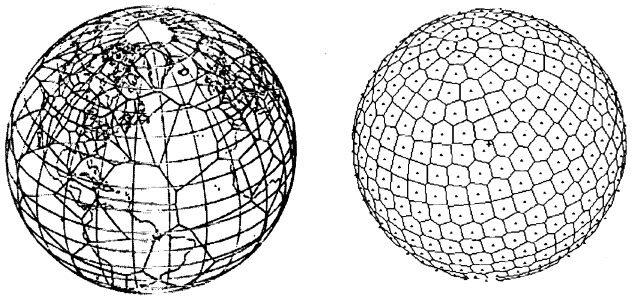
J. Chen is with the National Geometrics Center of China, 1 Baishengcun, Zizhuyuan, Beijing, China, 10004 (chenjun@nsdi.gov.cn).

X. Zhao is with the China University of Mining and Technology (Beijing), D11 Xueyuan Road, Beijing, China, 100083 (zxs@cumtb.edu.cn).

X. Zhao and Z. Li are with the Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Kowloon, Hong Kong (lsllz@polyu.edu.hk).

Photogrammetric Engineering & Remote Sensing
Vol. 69, No. 1, January 2003, pp. 79–89.

0099-1112/03/6901–079$3.00/0



(a) (b)

Figure 1. Voronoi diagram of spherical objects. (a) As a spatial index (Lukatela, 1987). (b) For interpolation (Watson, 1988).

noi diagram, which is similar to the raster mode on a planar surface. It makes use of the dilation operation through neighboring triangles, and is simplified by simply using their labeling codes. Another reason behind the development of the QTM-based method is that the QTM data structure is seamless, hierarchical, and numerically stable everywhere on the spherical surface. Its hierarchical data structure can be used to efficiently manage multi-resolution global data, and it allows spatial phenomena to be studied at different levels of detail in a consistent fashion across extensive regions of the sphere (Lee and Samet, 2000).

Following this introduction is a section critically reviewing current algorithms. The selection of tessellation methods on a spherical surface as a reference system is then introduced. This is followed by the presentation of a searching method of spherical neighbor triangles in detail. An algorithm for generating a Voronoi diagram of spherical objects is then presented in detail by dilation operation in QTM. Following that, the distortion between the great circle distance and the QTM distance will be analyzed. Finally, the conclusions and recommendations for future work will be presented.

A Critical Examination of Algorithms for the Generation of Voronoi Diagrams on a Sphere

The Voronoi diagram has been one of the research hotspots in the area of computer geometry since Shamos and Hoey (1975) introduced it to the computing domain as an efficient data structure. Most methods of Voronoi diagram generation are based on point sets on a planar surface, such as the incremental method, the divide and conquer method, the indirect generating method, and the parallel method (Aurenhammer, 1991; Okabe *et al.*, 1992; Li *et al.*, 1999; Okabe *et al.*, 2000). However, there are only a few algorithms for generating a spherical Voronoi diagram. In this section, these algorithms will be critically examined.

It is well known that distance on a sphere is different from that in Euclidean space. As a result, the Voronoi diagram on a sphere will have a different definition as follows [Okabe *et al.*, 2000; Lee and Samet, 2000]: Let $P = \{p_1, p_2, \ldots, p_n\}$ $(2 \leq n < \infty)$ be distinct points on a sphere S with the unit radius centered at the origin, and $\mathbf{X}$ and $\mathbf{X}_i$ be the location vectors of $p \in \mathbf{S}$ and $p_i \in \mathbf{S}$, respectively. The shortest distance from p to p_i on $\boldsymbol{S}$ is defined by the length of the lesser arc on the great circle (the circle whose center is at the center of $\boldsymbol{S}$) passing through p and p_i. Mathematically, this distance is written as

$$d_{gc}(p, p_i) = \arccos(\mathbf{X}^{\mathrm{T}} \mathbf{X}_i) \leq \pi. \quad (1)$$

This distance is called the *great circle distance*. The bisector defined with the great circle distance is given by the great circle that *perpendicularly* passes through the mid-point of the great circular arc combining p_i and p (*perpendicularly* meaning that sufficiently small segments of the two great circles around the mid-point are orthogonal). This bisector divides the sphere $\boldsymbol{S}$ into two disjoint hemispheres. Thus, the bisector defined with the great circle distance is well-behaved, and

$$V(p_i) = \{d_{gc}(p, p_i) \leq d_{gc}(p, p_j), j \in I_n \backslash \{i\}, p \in \mathbf{S}\} \quad (2)$$

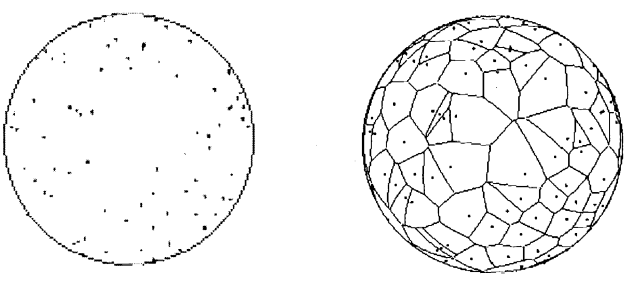
gives a non-empty set in $\boldsymbol{S}$. This set is called the spherical Voronoi polygon associated with p_i. The set of resulting spherical Voronoi polygons gives a generalized Voronoi diagram, which is called the spherical Voronoi diagram generalized by p_i on $\boldsymbol{S}$. Figure 2 shows a spherical Voronoi diagram.

A *Voronoi edge* is the intersection of two Voronoi regions that share a point, and a *Voronoi vertex* is the intersection of three or more Voronoi regions that share a point.

Augenbaum (1985) gives an insertion method for computing the Voronoi diagram of a set of n points on a sphere with time complexity $O(n^2)$, and Renka (1997) presents an incremental algorithm which can be constructed with time complexity $O(n\log n)$. However, as pointed out by Gold (1992) and Gold and Condal (1995), vector-based methods are good for point sets, very complex for line sets, and out of consideration for area sets. This is also true on a spherical surface. This serious deficiency is the main obstacle to the Voronoi data structure being widely applied in a dynamic GIS.

In order to solve this problem, Yang and Gold (1996) presented a *point-line* model. In this model, the complex objects are decomposed to points and lines. Voronoi diagrams for the points and lines are generated at first, and then translated to the Voronoi diagrams of the complex objects by removing the Voronoi edges between the points or line of the same object. More recently, Gold and Mostafavi (2000) extended this model onto a spherical surface. The advantage of this method is that it can generate Voronoi diagrams of relatively complex vector-based objects and deal with the dynamic changes of topological relations. However, this has been done with many additional steps, such as *decompose, calculation, remove*, and *compose*, etc. As a result, their algorithms and data structures are more complex. More importantly, the data structures of these vector-based methods lack hierarchical expression; therefore, it is very difficult to deal with the hierarchical expression of large quantities of spherical data.

To overcome the problem of the complex of vector-based methods, raster-based methods on a plane have been devised (Dehne *et al.*, 1989; Okabe *et al.*, 1992; Embrechts and Roose, 1996. These algorithms make use of distance transformations



(a) (b)

Figure 2. A Voronoi diagram on a sphere based on points sets (Geyer, 2000). (a) Discrete points on sphere. (b) Voronoi diagram on spherical points sets.

such as city blocks, chessboards, octagons, etc. (as shown in Figure 3). It is obvious that the approximation of these raster distances to the Euclidean distance becomes poorer and poorer when the distance becomes larger and larger. To solve this problem, Li *et al.* (1999) presented a raster-based method using a dynamic distance transformation with the principle of mathematical morphology. In their algorithm, the error is confined to one pixel. In addition, raster-based methods have concise conception and hierarchical recursion, and they can be easily extended to an *n*-dimensional Voronoi diagram.

However, these raster-based algorithms are limited to a plane and cannot be translated to a spherical surface directly because planar and spherical spaces are not homeomorphous. In this paper, a spherical mesh-based algorithm is presented for the generation of a Voronoi diagram on a spherical surface. The process of generating a Voronoi diagram is as follows: First, the spherical surface is subdivided into approximate triangles by QTM tessellation, and a QTM distance is defined which is similar to the planar raster distance (and will be discussed in the next section). Second, the *Edge* and *Vertex neighbor triangles* around the spherical objects (such as points, arcs, and regions) are determined by using algorithms for searching neighbor triangles in order to generate dilation traces of objects (which will be discussed in the section on "A Method for Searching Neighbors in QTM"). Finally, the spherical hexagon distance (which will be discussed in the section on "A QTM-Based Algorithm for the Generation of Voronoi Diagrams") is used for the dilation operation to compute spherical Voronoi diagrams of spatial objects.

Selection of a Tessellation Method for a Spherical Reference System

In order to generate a Voronoi diagram on a sphere, a tessellation method for a spherical surface and a labeling scheme should first be selected. The rules of selection are according to the efficiency of the transformation between the triangle code and its spherical coordinates and encoding method, which is suited to searching neighbor triangles.

Selection of Tessellation Method on Spherical Surface—O-QTM

Originally, the concept of spherical surface tessellation was presented by Fuller, a German cartographer, for studying mapping projections in the 1940s (Dutton, 1996). Since then, many researchers have approached this problem to project, analyze, and index global data. Many methods are based on *inscribed polyhedrons*, such as the *tetrahedron*, the *cube* (Snyder, 1992), the *octahedron* (Dutton, 1989; Goodchild *et al.*, 1991; Goodchild and Yang, 1992; Otoo and Zhu, 1993; Clarke and Mulcahy, 1995; Dutton, 1996; Dutton, 1999a; Dutton, 1999b), the *dodecahedron* (Wickman and Elvers, 1974), and the *icosahedron* (Fekete, 1990; White *et al.*, 1992; Lee and Samet, 2000), as shown in Figure 4. The edges of the polyhedron are projected to the spherical surface and form the edges of spherical triangles.

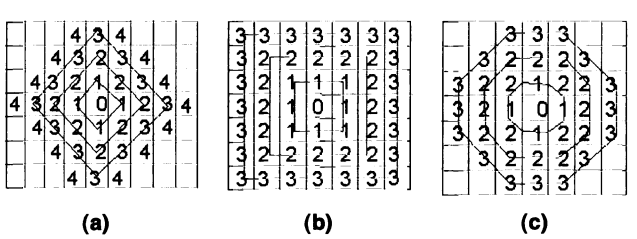


Figure 3. Various definitions of raster distance on a plane. (a) City block. (b) Chess board. (c) Octagon.

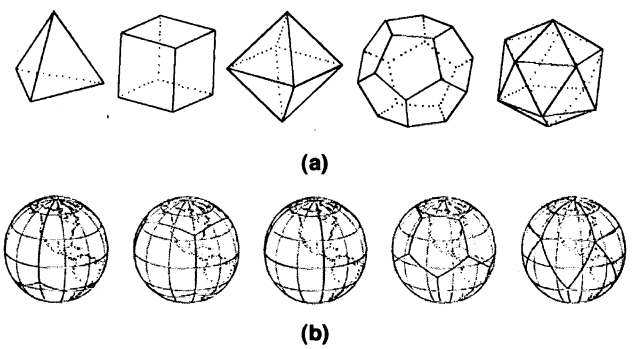


Figure 4. Spherical surface tessellation based on inscribed polyhedrons (White *et al.*, 1992). (a) Five polyhedrons. (b) Projected to the spherical surface.

In this study, the octahedron is chosen as a basis for an O-QTM. The reason for this selection is that it can be readily aligned with the conventional geographic grids of longitude and latitude. When this is done, its vertices occupy cardinal points and its edges assume cardinal directions, following the equator, the prime meridian, and the 90th, 180th, and 270th meridians, making it simple to determine which facet a point on the sphere occupies (Dutton, 1996). In addition, each facet is a right spherical triangle, and one subdivision line of each face is parallel to the equator.

There are some methods of recursive tessellation which satisfy the different requirements. It is well known that the efficiency of transformation between the triangle code and its spherical coordinates (latitude, longitude) is an important function in an application system (White and Kimerling, 1998). Thus, the *latitudes and longitudes average method* was selected for this study, as described by Dutton (1996). When a facet is subdivided, the latitudes and longitudes of pairs of its vertices are averaged to yield edge midpoint locations, and so on. This tessellation method makes it easy to exchange between spherical coordinates and triangle address codes. Clearly, with each level of subdivision, the triangles become smaller, and, at the 21st level of subdivision, their size is approximately 1 m, going down to 1 cm at the 28th level. Figure 5 illustrates levels 1, 2, and 3.

Selection of Encoding Method

A point on the Earth is usually defined by its longitude λ and latitude φ. In the spherical triangular quad-tree data structure, the position of a point is identified by the centroid of a decomposed triangle. The accuracy of the estimate of the point location improves with increasing levels of decomposition. The

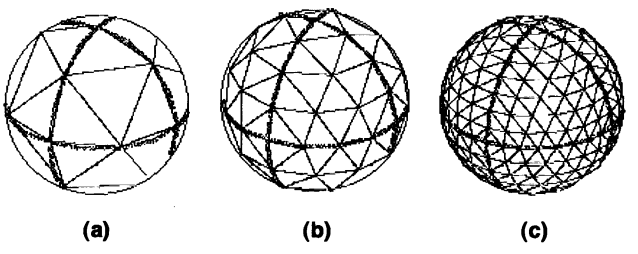


Figure 5. Hierarchical tessellation of spherical facet based on octahedron (Dutton, 1996). (a) Level 1. (b) Level 2. (c) Level 3.

addressing methods and data structures of the triangular regions are related to the operation and indexing of spherical surfaces. Several methods have been used, such as SQT (Sphere Quadtree) (Fekete, 1990), SQC (Semi-quadcode) (Otoo and Zhu, 1993), THDS (Triangular Hierarchical Data Structure) (Goodchild *et al.*, 1991), and QTM-IDs (Dutton, 1989; Dutton, 1996; Dutton, 1999a; Dutton, 1999b).

The triangle labeling method was selected for our study and is similar to that used for the octahedron (Goodchild *et al.*, 1991). The only difference is that the conversion between triangle codes and sphere coordinates used in Goodchild *et al.* (1991) has a worst case, where the execution time is proportional to the maximum level of decomposition. In contrast, the conversion method used in our approach is the *Row-Column Approach*, in which the directions of coding schemes are fixed and the transformation between triangle codes and sphere coordinates is quicker. The address code of each QTM cell consists of an octant number (from "1" to "8"), followed by up to 30 quaternary digits (from "0" to "3"), which names a leaf-node in a triangular quadtree rooted in the given octant. At the kth level of decomposition, the triangle address A is represented by $A = a_0a_1a_2a_3 \ldots a_k$, where a_1 to a_k are k quaternary digits, and a_0 is an octal digit representing the initial octahedral decomposition at level 0. Because the Earth is divided into eight equilateral spherical triangular faces, as shown in Figure 6a, an octal digit is assigned to a_0 for addressing each face as follows:

$a_0 = 0, 1, 2, 3 \quad 90° \geq \phi \geq 0°$, for the Northern hemisphere
$a_0 = 4, 5, 6, 7 \quad 0° > \phi \geq -90°$, for the Southern hemisphere

and

$a_0 = 0, 4 \quad 90° > \lambda \geq 0°$
$a_0 = 1, 5 \quad 180° > \lambda \geq 90°$
$a_0 = 2, 6 \quad -90° > \lambda \geq -180°$
$a_0 = 3, 7 \quad 0° > \lambda \geq -90°$

In the next code $a_1a_2a_3 \ldots a_k$, the code of the center triangular is "0," the up or down triangular is "1," left is "2," and right is "3," as shown in Figure 6b.

The Definition of QTM Distance on a Spherical Surface

As has been discussed previously, the Voronoi diagram on a sphere is formed by a series of contiguous Thiessen polygons, and a polygon is constructed according to the great circle distance between spherical points. The line of thought for the construction of a Voronoi diagram in the vector mode is still valid in the QTM mode. The critical problem arising is "how to determine the distances between points on a sphere in the QTM mode."

In the vector mode, the *distance* on a sphere means the great circle (or arc or geodesic) distance. The distance between two points $\mathbf{X}_1$ (λ_1, φ_1) and $\mathbf{X}_2$ (λ_2, φ_2) on a sphere is defined as

$$L_{X_1X_2} = R\theta = R \cos^{-1}(\mathbf{X}_1 \cdot \mathbf{X}_2) \quad (3)$$

where θ is an angle between $\mathbf{X}_1$ and $\mathbf{X}_2$, and the range of $cos^{-1}\theta$ is taken to be $[0, \pi]$.

In the QTM mode, the location codes of mesh pixels define the coordinates. For example, if there are two points $\mathbf{X}_1$ (A_1) and $\mathbf{X}_2$ (A_2) sharing a vertex (e.g., Figure 7a), then the Euclidean distance between them is 4/3 (=1.333) pixels (assuming the unit to be the number of pixels, e.g., the length of the edge of one triangle mesh on the sphere). The result in decimal form is inconvenient to use in the QTM mode; a distance in an integer number is more desirable and thus normally employed. Thus, either 1 or 2 could be the best candidate for use as a spherical mesh distance to approximate the *great circle distance.*

Initially, the concept of raster distance is directly related to the "number of neighbors" or the "directions of connection." There are eight neighbors if one is allowed to travel along the diagonals on a plane. However, this is somewhat different for a spherical triangle mesh. In the example given above, suppose that one is traveling from point $\mathbf{X}_1$ to point $\mathbf{X}_2$: there is only one step if one is allowed to travel along the diagonals. In this case, the most appropriate QTM cells distance between these two points is 1. This is the case with 12 neighbors. A diagrammatic representation of this type of distance is given in Figure 7b. The shape of this diagram is like a hexagon, thus the name "hexagon distance."

A Method for Searching Neighbors in QTM

The neighbor searching methods presented by Dutton (1990), Fekete (1990), Goodchild and Yang (1992), Otoo and Zhu (1993), and Lee and Samet (2000) are quite different from each other. For example, the methods used by Dutton (1990) and Fekete (1990) form new triangular regions for each level of decomposition by bisecting the edges of the parent triangle. This way of viewing the subdivision process results in a *floating* labeling scheme. The disadvantages of this labeling method are that the transformation and node labels depend on orientation, which makes it difficult to use binary arithmetic in these methods. Furthermore, neighbors searching that traverses different faces of an icosahedron is much harder (Lee and Samet, 2000). In contrast, the fixed labeling scheme used by Goodchild and Yang (1992) and Lee and Samet (2000) is considerably simpler and can be easily used to yield a neighbor searching method. The method used by Lee and Samet (2000) is based on an icosahedron and applied to navigate in a spherical quadtree. Although Goodchild and Yang (1992) gave a neighbor searching method based on an octahedron, they only presented the searching method of *edge-neighbor-triangles* in

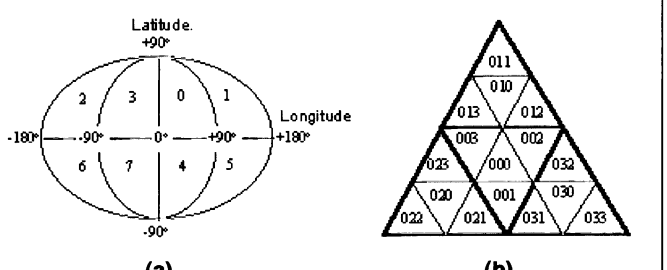


Figure 6. Surface partition and encoding. (a) Original partition of Earth surface. (b) Encoding in 0 unit.

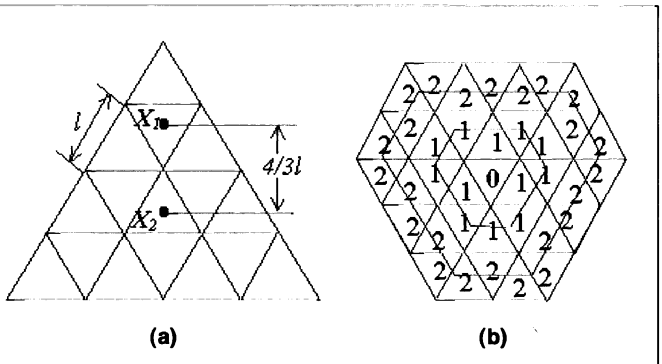


Figure 7. The definition of QTM distance on a spherical surface. (a) Different distance between two points. (b) Hexagon structuring element.

their paper. Indeed, they did not classify and analyze the differences between the searching methods while the triangles are at different locations near the borders of the octahedron. Furthermore, they did not discuss the searching method of *vertex-neighbor-triangles*. In this section, these problems will be tackled in detail and depth.

Definitions of Neighboring Triangles

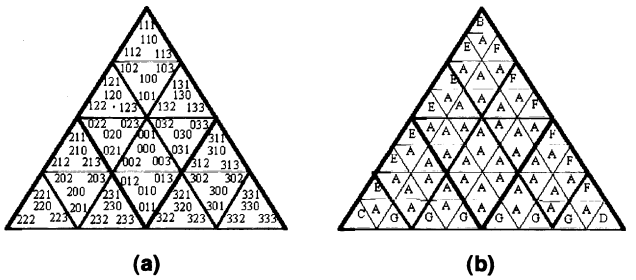
In Figure 8a, the definition of neighboring triangles in QTM is given. The neighbors with shared edges are called *edge-neighbor-triangles*. Those with only common vertices are called *vertex-neighbor-triangles*. In the data structure of O-QTM, the neighbors of a triangle, which are located in two adjacent octants must also be specially considered because the searching methods for different locations of *border triangles* are different. From Figure 8b, it can be seen that, if a triangle has edge(s) at the border of an octant, the triangle will have *edge-neighbor-triangle*(s) and *vertex-neighbor-triangle*(s) in its neighboring octant(s); if a triangle has vertices(s) at the border of an octant, the triangle will only have *vertex-neighbor-triangle*(s) in its neighboring octant(s). Border triangles can be classified into four categories: *edge, sub-edge, corner,* and *sub-corner triangles*, and these can be defined as follows (Goodchild *et al.*, 1991):

- *Edge triangle* (1) - if it has exactly one edge-neighbor-triangle in the adjacent octant.
- *Sub-edge triangle* (2) - if it has exactly three vertex-neighbor-triangles in the adjacent octant.
- *Corner triangle* (3) - if it has exactly two edge-neighbor-triangles in the adjacent octant.
- *Sub-corner triangle* (4) - if it has exactly six vertex-neighbor-triangles in the adjacent octant.

An Algorithm for Searching Edge-Neighbor-Triangles

All triangles have three edge-neighbor-triangles in QTM on a sphere. We use the codes *t, l, r* to represent the three edge-neighbor-triangles with common top, left, and right edges for a given triangle *U* inside an octant, and *T, L, R* to represent the edge-neighbor-triangle of a top, left, and right edge triangle lying in the adjacent octant. The different searching methods will be used if the triangles are at different locations in one octant. Border triangles can be classified into seven categories and defined as follows (see Figure 9):

- *Inside triangles (A)* - The address code of a given triangle *U* includes digital '0' or digital '1' *AND* '2' *AND* '3'. Its edge-neighbor-triangles are (*t, l, r*).
- *Top corner triangles (B)* - The address code of a given triangle *U* includes digital '1' only. Its edge-neighbor-triangles are (*t, L, R*).
- *Left corner triangles (C)* - The address code of a given triangle *U* includes digital '2' only. Its edge-neighbor-triangles are (*T, L, r*).



(a) (b)

Figure 9. Categories of triangles by different searching algorithms of edge-neighbor-triangle.

- *Right corner triangles (D)* - The address code of a given triangle *U* includes digital '3' only. Its edge-neighbor-triangles are (*T, l, R*).
- *Left edge triangles (E)* - The address code of a given triangle *U* includes digital '1' *AND* '2' only. Its edge-neighbor-triangles are (*t, L, r*).
- *Right edge triangles (F)* - The address code of a given triangle *U* includes digital '1' *AND* '3' only. Its edge-neighbor-triangles are (*t, l, R*).
- *Top edge triangles (G)* - The address code of a given triangle *U* includes digital '2' *AND* '3' only. Its edge-neighbor-triangles are (*T, l, r*).

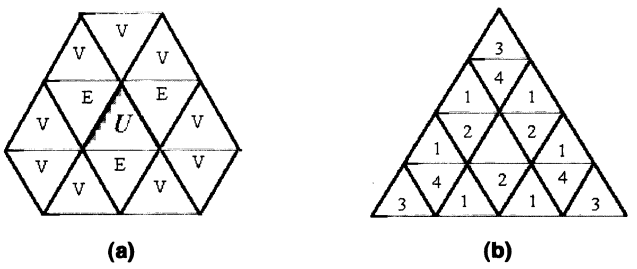
Let the triangular addresses of the direct neighbors of a given triangle *U* be represented by

$$t = t_1, t_2, t_3, \ldots, t_k$$
$$l = l_1, l_2, l_3, \ldots, l_k$$
$$r = r_1, r_2, r_3, \ldots, r_k$$
$$T = T_1, T_2, T_3, \ldots, T_k$$
$$L = L_1, L_2, L_3, \ldots, L_k$$
$$R = R_1, R_2, R_3, \ldots, R_k$$

The data strings *t, l, r, T, L,* and *R* can be obtained from the triangular address *U*. The details can be seen in Goodchild *et al.* (1991).

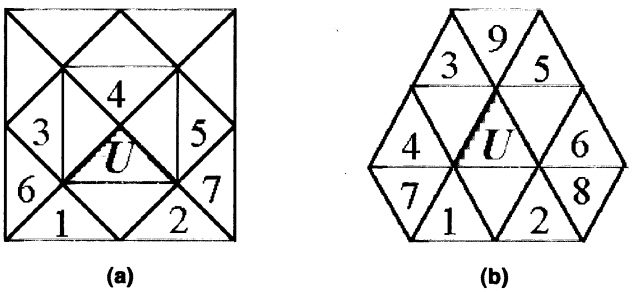
An Algorithm for Searching Vertex-Neighbor-Triangles

Of all triangles in an octant, the corner triangle has seven vertex-neighbor-triangles (Figure 10a), and the other has nine vertex-neighbor-triangles, as shown in Figure 10b. There are several methods of searching for vertex-neighbor-triangles. In



(a) (b)

Figure 8. Definitions of neighbor triangles and border triangles. (a) Edge-neighbor-triangles and vertex-neighbor-triangles. (b) Classifications of border triangles.



(a) (b)

Figure 10. Vertex-neighbor-triangles. (a) Corner triangle. (b) No-corner triangle.

this study, they have been found through their edge-neighbor-triangles.

Let the left edge-neighbor-triangle of triangle U be expressed as $L(U) = Left(U)$, the right edge-neighbor-triangle as $R(U) = Right(U)$, and the up or down edge-neighbor-triangle as $T(U) = Top(U)$.

The searching algorithm for vertex-neighbor-triangles varies with locations of U in the octant, especially when the border triangles are much more complex. These triangles can be classified into nine different types (shown as Figure 11): *inside triangles, top-corner triangles, left-corner triangles, right-corner triangles, left-edge triangles, right-edge triangles, left sub-edge triangles, right sub-edge triangles*, and *sub-corner triangles*. Each type has seven (or nine) vertex-neighbor-triangles expressed by its edge-neighbor-triangles as follows:

- *Inside triangles (9): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), R(R(U)), L(L(T(U))), R(R(T(U))), R(T(L(U)))*.
- *Top-corner triangles (7): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), L(L(T(U))), R(R(T(U)))*.
- *Left-corner triangles (7): L(T(U)), R(T(U)), L(L(U)), T(R(U)), R(R(U)), R(R(T(U))), L(T(R(U)))*.
- *Right-corner triangles (7): L(T(U)), R(T(U)), T(L(U)), L(L(U)), R(R(U)), L(L(T(U))), T(R(R(U)))*.
- *Left-edge triangles (9): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), R(R(U)), T(L(T(U))), R(R(T(U))), T(L(L(U)))*.
- *Right-edge triangles (9): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), R(R(U)), L(L(T(U))), T(R(T(U))), R(T(L(U)))*.
- *Left-sub-edge triangles (9): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), R(R(U)), T(L(T(U))), R(R(T(U))), R(T(L(U)))*.
- *Right-sub-edge triangles (9): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), R(R(U)), L(L(T(U))), T(R(T(U))), R(T(L(U)))*.
- *Sub-corner triangles (9): L(T(U)), R(T(U)), T(L(U)), L(L(U)), T(R(U)), R(R(U)), T(L(T(U))), T(R(T(U))), R(T(L(U)))*.

A QTM-Based Algorithm for the Generation of Voronoi Diagrams on the Sphere

The algorithm for generation of a Voronoi diagram on a sphere is based on a dilation operator of spherical triangles. In this section, we will develop the triangle-dilation operator in QTM according to the principle of the raster-dilation operator in mathematical morphology. From this and the neighbor searching method described in the previous section, the *QTM distance* diagram can be obtained, which consists of a number of distance contours radiated from each spherical object. The most distant contours of spherical objects form the approximate boundaries of the spherical Voronoi diagram.

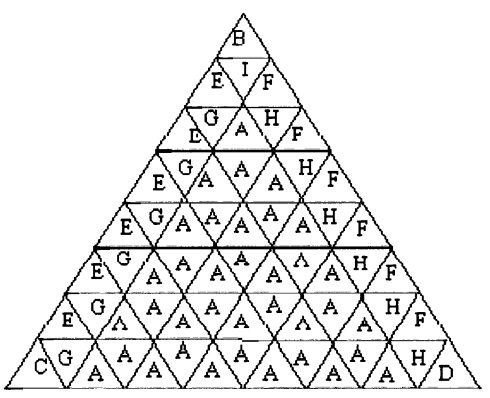


Figure 11. Nine different types in searching methods of vertex-neighbor-triangles.

Distance Contour in QTM Generated by the Dilation Operator

Dilation and erosion are two of the basic operators in mathematical morphology, and they have wide applications in digital image processing as well as geographical information science (Su *et al.*, 1997; Li *et al.*, 1999). These two basic operators in QTM cells can be defined as follows:

$$\begin{aligned} &dilation && A \oplus B = \cup b \in BA_b \\ &erosion && A \ominus B = \cap b \in BA_b \end{aligned} \tag{5.1}$$

where A is an original region on a sphere and B is a structuring element; an example of dilation and erosion is given in Figure 12.

The neighboring-triangle searching algorithm described in the previous section can be used directly for the dilation operation. This is because the dilation operation requires searching all neighbor (edge- and vertex-neighbor) triangles of spherical objects (including a point, arc, or region) in the QTM mode.

For an inside triangle, the region expanded is a hexagon with three edges of length $(m - 1) \times l$ and three edges of length $m \times l$ if the region does not cross to another octant, where m is the number of times the procedure is repeated and l is the edge length of the triangle at the given level. The form of the region changes if the region crosses the edge of an octant, as shown in Figure 13. The distance between the border of the dilated region and the nearest edge of a given triangle (point) varies from $nl\sqrt{3}/2$ *to* nl, a factor of 0.866, which is larger than in a rectangular raster where the edge-to-diagonal ratio of a square is 0.717 (Goodchild *et al.*, 1991), e.g., the error of forming a dilation region by a hexagonal structuring element in QTM is smaller than that by a chess structuring element (Figure 3) in rectangular cells on a plane. In addition, the topological and metrical properties of the region are preserved: i.e.,

- The region generated is connected and there is no hole, and
- The region dilated each time is a striped region surrounding the old region with a width of $l\sqrt{3}/2$.

The Principle of Generating a Voronoi Diagram on a Sphere in QTM

The algorithm for the generation of a Voronoi diagram on a spherical surface is based on the principle of dilation operation in mathematical morphology. In the QTM, a point is represented by a triangle, an arc is represented by a series of neighbor triangles, and a region is represented by a series of neighbor triangles on and within its boundary trace. The dilation operation of an arc or region can be simply done by dilation of all the triangles by which the arc or region is described. Thus, the process of generating a Voronoi diagram is as follows: First, the *edge* and *vertex neighbor triangles* around the spherical objects (such as points, arcs, and regions) are determined by using the algorithm of searching neighbor triangles presented in the previous section. Second, all duplicate triangles are removed and the dilation traces of the objects are generated. The spherical distances are approximately equal from the outer

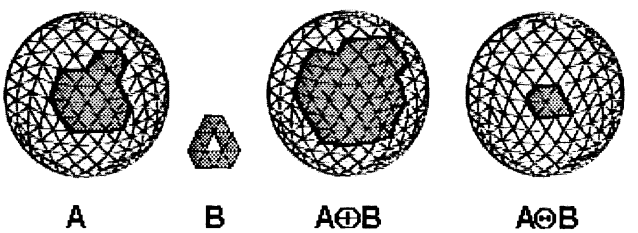


Figure 12. Dilation and erosion operation based on QTM.

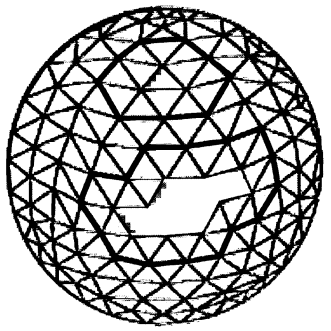


Figure 13. Triangular dilation operation on a spherical surface.

boundary of the dilation trace to the boundary of the object. Next, the dilation operations are repeated until the dilation trace is intersected with the other dilation trace. The intersecting trace is just the Voronoi edge between the two objects.

Algorithm

Input: tessellation level N and data sets in sphere surface: $\Gamma = \{A_1, A_2, A_3, \ldots, A_n\}$

Output: Voronoi diagram of sets Γ, and store in data file *VoronoiData*.

CsphVoronoiView::OnCalculateVoronoi ( )
{
step 1: **LongLatitude_to_ QTMcode (Γ);**
step 2: For every object A_i in Γ
{ step 2.1: For every QTMcode Q_j in A_i;
{ **Adjact12**(Q_j); //searching neighbor triangles
if Adjact12(Q_j) are copy code
Delete_copyQTMcode(Q_j);
Else Diltation_$A[i] \leftarrow$ Adjact12(Q_j)
}
step 2.2: For every Diltation_$A[i]$
{ For every QTMcode Q_{im} in $A[i]$ and every QTMcode Q_{jk} in $A[j]$, $i \neq j$
{ if ($Q_{im} = Q_{jk}$)
VoronoiData $\leftarrow Q_{jk}$;
}
}
}
step 3: **QTMcode_to_LongLatitude**(VoronoiData);
step 4: **Output( );**
}over

Based on the algorithm described above, we have developed a computer program in platform OpenGL with VC^{++} language. In this experiment, the spherical surface is tessellated into five levels (the number of triangles is $8 \times 4^5 = 8192$). Voronoi diagrams of point sets, arc sets, area sets, and any sets are generated by QTM address codes (the results are shown in Figure 14). The results indicate that the time consumption of this algorithm with points, arcs, and regions is equal and proportionate to the levels of spherical surface tessellation.

Distortion Analysis

Two types of distortion are produced in this algorithm for generating a spherical Voronoi diagram in the QTM mode: one is from the distortions of the areas and lengths of the triangles when the spherical triangles are recursively partitioned; the other is from the difference between the *great circle distance* and the QTM distance while the dilation operation is processing.

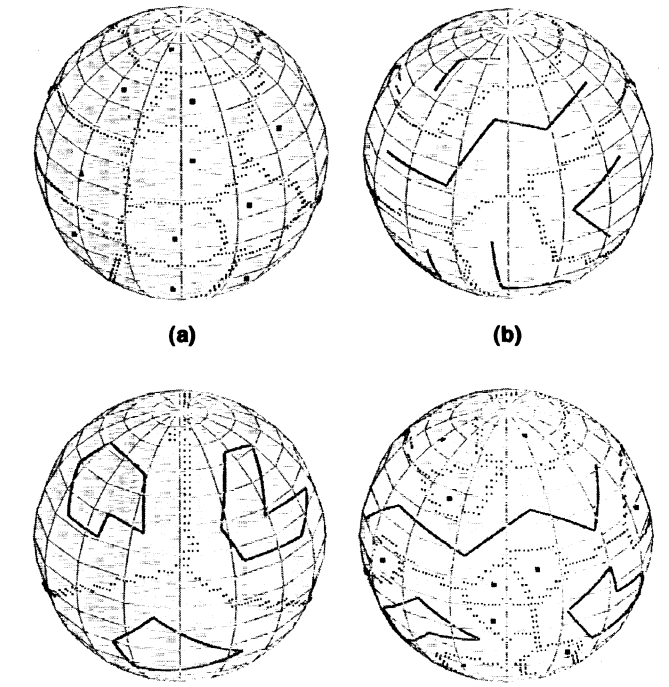


Figure 14. Voronoi diagrams of spherical objects. (a) Point sets. (b) Arc sets. (c) Area sets. (d) Composite object sets.

Distortions of Triangles in Areas and Edge Lengths

If recursive tessellation for the spherical triangle is done, the shapes and lengths of triangles have to be varied, i.e., the spherical surface cannot be subdivided into completely the same meshes in shape and area at arbitrary levels. The distortions of areas and lengths of triangles are calculated as the following: Let X_1, X_2, and X_3 be three vertex points of a spherical triangle. The length of two points-L-can be calculated from Equation 3.

The area $S_{X1\ X2\ X3}$ of the spherical triangle is

$$S = R^2\left[\sum_{i=1}^{3}\alpha_i - \pi\right] \tag{4}$$

where α_i are the interior angles of the triangle in radians. The sum of the three angles is always greater than π and less than 3π (for a planar triangle, $(\Sigma\alpha_i = \pi)$. The angle $\angle X_1X_2X_3$ of a spherical triangle can be calculated from the formula

$$\angle X_1X_2X_3 = \cos^{-1}\left[\frac{X_2 \times X_1}{|X_2 \times X_1|} \cdot \frac{X_2 \times X_3}{|X_2 \times X_3|}\right]. \tag{5}$$

The areas and lengths of triangles can be calculated using Equations 3 and 5. Their distortions are then shown as Table 1 and Figure 15.

From Figure 15, the ratio between the largest and smallest triangles increases with recursive subdivisions, but is no more than 1.83; similarly, the ratio between the longest and shortest edges increases and is no more than 1.87 (as shown in Table 1). The values of $(\Delta S^i - \Delta S^{i-1})/(\Delta S^{i-1} - \Delta S^{i-2})$ and $(\Delta L^i - \Delta L^{i-1})/(\Delta L^{i-1} - \Delta L^{i-2})$ are almost *constants* (about 0.25 and 0.63, respectively). That is to say, the following properties of the spherical surface in terms of finite spatial triangles are preserved in this tessellation scheme (Jones, 1997):

TABLE 1. THE DISTORTIONS OF AREA AND LENGTH OF TRIANGLES IN 20 LEVELS

Level	Number of triangles	Average area of triangles (km^2)	$\Delta S = S_{max}/S_{min}$	$\frac{\Delta S^i - \Delta S^{i-1}}{\Delta S^{i-1} - \Delta S^{i-2}}$	Average length of triangles (km)	$\Delta L = L_{max}/L_{min}$	$\frac{\Delta L^i - \Delta L^{i-1}}{\Delta L^{i-1} - \Delta L^{i-2}}$	scale
1	4	16,120,936	1.49448		5000	1.30656		
2	16	4,030,234	1.74583		2500	1.46622		
3	64	1,007,559	1.80803	0.2475	1250	1.59884	0.8306	
4	256	251,890	1.82320	0.2439	625	1.68914	0.6809	
5	1,024	62,927	1.82710	0.2571	313	1.75031	0.6774	
6	4,096	15,743	1.82808	0.2513	256	1.79058	0.6583	
7	16,384	3,936	1.82832	0.2505	78	1.81661	0.6463	
8	65,536	984	1.82839	0.2766	39	1.83327	0.6400	1:100M
9	262,144	246	1.82840	0.2154	20	1.84387	0.6363	1:50M
10	1,048,576	61	1.82840	.....	10	1.85058	0.6330	1:20M
11	4,194,304	15	1.82841	.....	5	1.85483	0.6333	1:10M
12	16,777,216	4	1.82841	.....	2	1.85751	0.6306	1:5M
13	67,108,864	0.960883	1.82841	.....	1	1.85920	0.6306	1:2M
14	2.6844E+08	0.240221	1.82841	.....	0.610	1.86027	0.6331	1:1M
15	1.0737E+09	0.060055	1.82841	.....	0.305	1.86094	0.6262	1:500K
16	4.2950E+09	0.015014	1.82841	.....	0.153	1.86136	0.6269	1:250K
17	1.7180E+10	0.003753	1.82841	.....	0.076	1.86163	0.6428	1:100K
18	6.8719E+10	0.000938	1.82841	.....	0.038	1.86180	0.6310	1:63K
19	2.7488E+11	0.000235	1.82841	.....	0.019	1.86191	0.6267	1:50K
20	1.0995E+12	0.000059	1.82841	.....	0.010	1.86198	0.6258	1:25K

- The tessellation scheme is hierarchical, with triangles at each level being subdivisions of triangles at the higher level;
- Triangles at any level of resolution should be approximately the same size, wherever they are located on the sphere;
- Triangles at any level are all approximately the same shape, wherever they are located; and
- The tessellation scheme preserves topological relationships correctly, particularly adjacencies.

The triangles are almost the same in area and edge length to recursive tessellation. Hierarchical operations, large quantities of data management, and especially recursive dilation operations of triangles are easily carried out in this tessellation scheme.

Distortion of Dilation Operations in Different Locations

In planar mode, it is well known that the maximum difference between a Euclidean distance and a raster distance is a linear function of the distance itself. The distortion resulting from the dynamic distance transformation is consistent with the maximum distortion of about 1 pixel, but requires more time (Li *et al.*, 1999). It is different for spherical QTM cells. In this case, we select 16 points at different locations in one octant unit, as shown in Figure 16. Distortions of dilation operations in different locations are calculated to illustrate the relations between the distortions and distances.

In this experiment, the spherical surface is tessellated into five levels. Each point needs to dilate 64 times until its dilation traces cover the whole globe. The distortion values of points are selected every four dilations and calculated using the following equation:

$$\varepsilon_{X_0}^k = |Max\{S_{X_0X_i}^k\} - \frac{1}{n}\sum_{i=1}^{n} S_{X_0x_i}^k| \quad (6)$$

where S_{X0Xi} is the great circle distance between the center of origin triangle X_0 and the center of dilation triangle X_i, K is the number of dilation steps, and n is the number of dilation triangles in the K^{th} step. The results are listed in Table 2 and a diagrammatic representation is given in Figure 17. The difference $\varepsilon_{X_0}^k$ is shown on the horizontal axis and the dilation time k (1 indicates 4 times dilation) is shown on the vertical axis.

From Table 2 and Figure 16, it can be seen that the distance distortion in dilations of QTM cells is not related to distance,

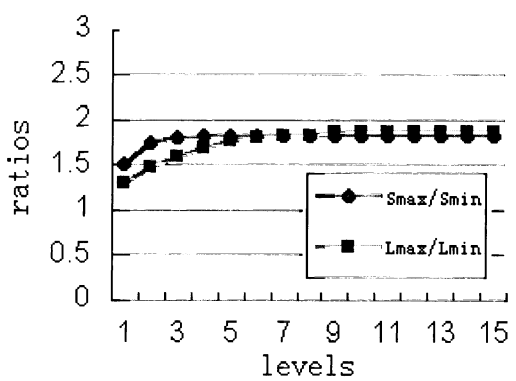


Figure 15. Distortions of areas and lengths of triangles in each level.

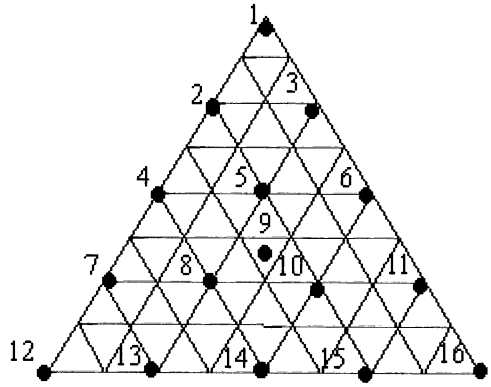


Figure 16. Different locations for generating points in one octant unit.

TABLE 2. DISTANCE DISTORTION IN DIFFERENT LOCATIONS IN ONE OCTANT

No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Code	011111	010222	010333	002222	000111	003333	020222	023111	000000	032111	030333	022222	023222	001111	032333	033333
	0.411907	0.69933	0.69932	0.64506	0.42734	0.64505	0.54937	0.41807	0.40959	0.41807	0.54938	0.36632	0.41511	0.36408	0.41512	0.36634
	0.649852	1.7747	1.7747	1.49899	1.15471	1.49899	1.52342	1.20256	0.99798	1.20256	1.52342	0.82271	1.33701	0.65641	1.33701	0.82276
	0.686777	2.49208	2.49209	2.1732	2.13388	2.1732	2.22594	2.63628	2.01132	2.63628	2.22596	1.2578	2.36597	1.06115	2.36596	1.25786
	0.701295	3.16096	3.16097	2.70467	2.89345	2.70467	2.31304	3.45141	3.03882	3.45141	2.31305	1.59	2.40389	1.56056	2.40389	1.59005
	0.708948	3.67458	3.67459	3.02013	3.76608	3.02013	2.77745	4.63031	4.37865	4.63028	2.77746	1.79546	2.56478	2.23154	2.56478	1.79547
	0.713609	3.9713	3.97131	3.30643	4.87559	3.30643	3.03323	4.68133	5.45334	4.68133	3.03323	1.88675	2.75142	3.01458	2.75143	1.88677
	0.716686	4.11667	4.11667	4.13283	5.39668	4.13283	3.03167	5.23225	5.79478	5.23223	3.03168	1.76682	3.09119	3.37315	3.09119	1.76683
	0.718814	4.22106	4.22107	4.80901	4.737	4.80901	3.68339	5.61678	5.31146	5.61678	3.6834	1.36678	3.72667	3.89026	3.72667	1.36679
	0.727156	4.30267	4.30267	5.29146	4.53272	5.29146	4.18155	6.45906	5.46465	6.45906	4.18156	1.02956	4.35537	4.48885	4.35537	1.02956
	0.725268	4.4943	4.4943	5.63538	4.92742	5.63538	4.65598	6.88181	5.49445	6.88181	4.65597	1.56067	4.47457	5.01014	4.47457	1.56063
	0.722814	4.77222	4.77222	5.87404	5.73635	5.87404	5.0131	7.27189	5.49483	7.27188	5.01309	1.78796	4.7645	5.43828	4.76449	1.78791
	0.717957	4.80471	4.80471	6.04726	6.7058	6.04726	4.52039	6.46844	5.49472	6.46844	4.52038	1.80435	4.91235	6.09912	4.91235	1.80437
	0.711696	4.76648	4.76648	6.50883	7.3606	6.50883	4.43745	6.06359	4.33333	6.06359	4.43745	1.6357	4.92346	6.89999	4.92346	1.63572
	0.699718	4.66048	4.66048	7.03693	6.99865	7.03693	4.22782	5.57075	2.33333	5.57071	4.22782	1.33239	4.79811	7.29015	4.79811	1.3324
	0.652448	4.53781	4.53781	7.56856	5	7.56856	3.99892	2.33103	0.82109	2.33103	3.99893	0.795491	4.57048	7.57821	4.57048	0.79550
	0.666573	4.28715	3.90809	8.01646	3.66838	8.01644	3.67028	0.72415	0	0.72416	3.67588	0.46746	4.39015	7.77551	4.39016	0.40564

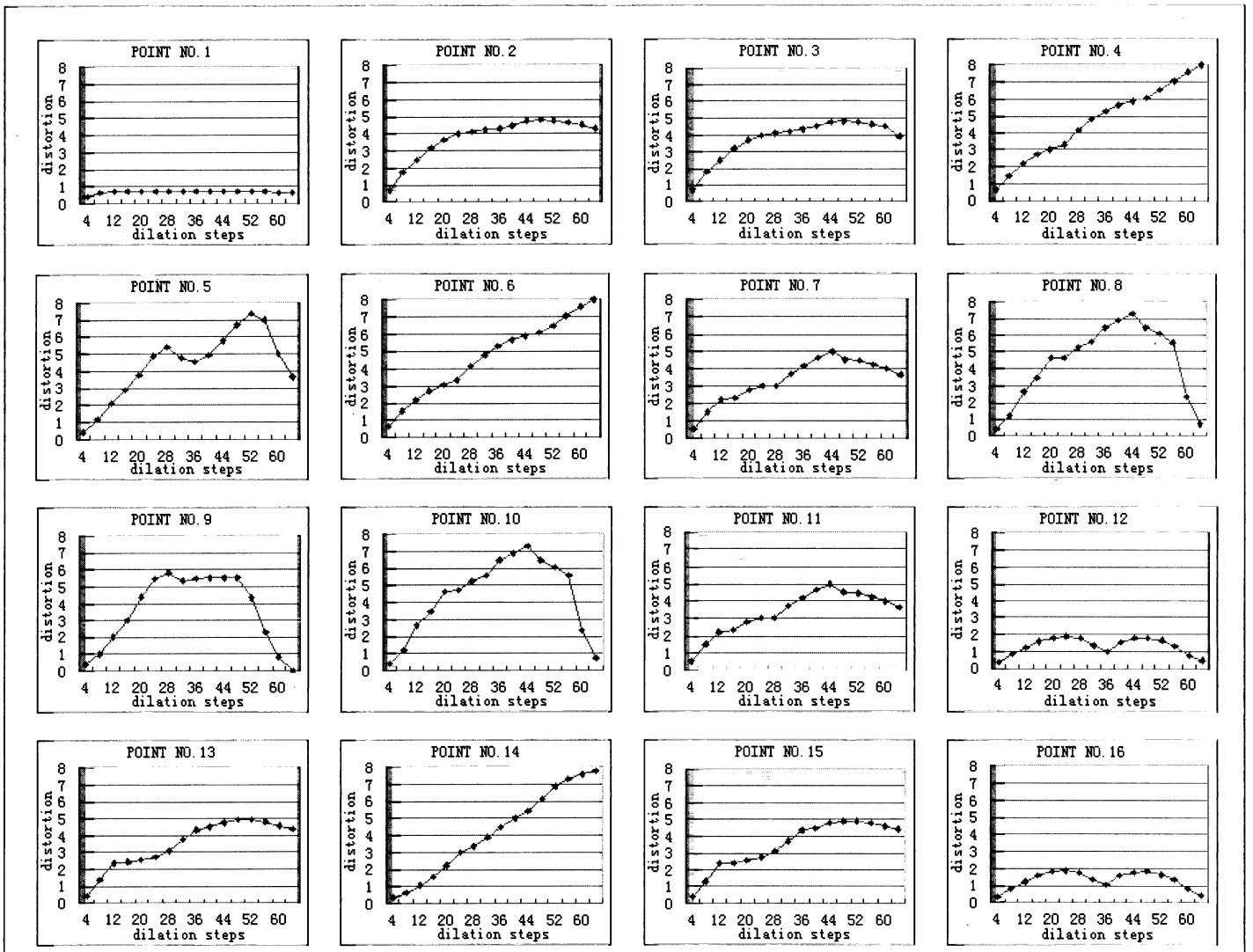


Figure 17. Distance distortions in dilation operation from different generating points.

but mainly to the original location of the generating point. The distortions vary differently with increasing dilation steps of the generating points in different locations. Special characteristics are found as follows:

- The distance distortion between the *great circle distance* and the QTM cells distance changes little when the generating point is located at or near corner triangles, shown as points 1, 12, and 16 in Figure 17.
- Distance distortion is an almost linear function of distance itself when the generating point is located at or near triangles in the midpoint of edges, shown as points 4, 6, and 14 in Figure 17.
- If the generating points are located at or near the center of the octant unit, the distortion increases when the dilation distance increases. However, after the dilation trace covers half the globe, distortion decreases when the dilation increases, and reduces to zero in the end, shown as points 9 in Figure 17.
- Distortions vary similarly if the generating points are in the same circle whose center is at the center of the octant unit, such as points 2, 3, 7, 11, 13, 15 or 5, 8, 10, etc., as shown in Figures 17 and 18.

On the other hand, the spatial relation (as a topological relation) reasoning is independent of measurement and only related to the resolution of triangles. Thus, errors in Voronoi diagram generation have little influence on management of the dynamic data of spherical objects. In other words, the influence is only on adjacent searching of distant objects.

The Method of Distortion Control—Discussion

In the raster mode on a plane, the dilation steps can be adjusted by selecting a different structuring element. Thus, the distortion of distance transformation can be controlled within one pixel by dynamic distance transformation (Li *et al.*, 1999). But this method has one use in the QTM mode because QTM is a non-equal triangle mesh partition the same in area and shape as the description given above. Only selecting a different structuring element in the dynamic distance transformation has a little influence on controlling the distortion between the QTM distance and the great circle distance. One resettling method may be taken into account, in which the QTM distance is adjusted by comparing the great circle distance with the QTM distance instead of selecting a different structuring element.

Conclusions

The Voronoi diagram data structure has become an efficient tool for solving global dynamic GIS with its dynamic stability; thus, much attention has been paid to its generating algorithm. However, there are fewer methods on a sphere than on a plane, and all of them are vector-based for point sets. In this paper, a new generating algorithm for the Voronoi diagram on spheres is developed by recursive dilation operation in QTM (Quaternary Triangular Mesh), which can easily handle arbitrary composite objects (include arcs and regions on a sphere). The principle of dilation operation in mathematical morphology is extended to a spherical surface. One contribution in this paper is the extension of neighbor-triangles searching methods so that they can be used for the dilation operation of spherical objects expressed in QTM. Another important contribution is the development of a method for generating Voronoi diagrams based on QTM. A detailed algorithm is also presented. This algorithm can handle arbitrary spherical objects (including points, arcs, and regions). It has also been experimentally shown that the time consumption of this algorithm with points, arcs, and regions is equal and proportionate to the levels of spherical surface tessellation; and the difference (error) between the *great circle distance* and the QTM cells distance is slightly related to spherical distance (not like the raster dilation on a plane), and mainly related to the locations of the generating points.

With the methods described in this paper, future work includes increasing the dilation speed of object sets by use of a spherical surface quadtree, increasing the precision by deleting or inserting dilation triangles by dynamic comparison of the spherical distances and QTM distances, and using the spherical Voronoi diagram for spatial query and for constructing the global dynamic data model.

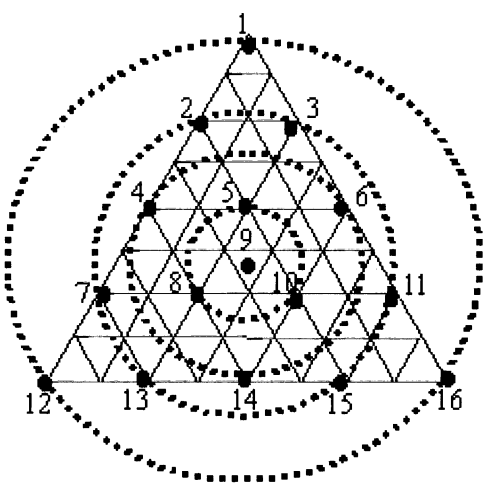


Figure 18. Distortions vary similarly if the generating points are located in the same circle.

Acknowledgments

The work described in this paper was supported by the National Natural Science Foundation of China (under grant No.69833010) and by the Research Grants Council of the Hong Kong Special Administrative Region (Project No. PolyU 5048/98E).

References

Augenbaum, M., 1985. On the construction of the Voronoi mesh on a sphere, *Computational Physics*, 59:177–192.

Aurenhammer, F., 1991. Voronoi diagram—A survey of a fundamental geometric data structure, *ACM Computing Survey*, 23(3):345–350.

Chen, J., C.M. Li, Z.L. Li, and C. Gold, 2001. A Voronoi-based 9-intersection model for spatial relations, *International Journal of Geographical Information Science*, 15(3):201–220.

Clarke, K.C., and K.A. Mulcahy, 1995. Distortion on the Interrupted Modified Collignon Projection, *Proceedings of GIS/LIS 95*, 14–16 November, Nashville, Tennessee (American Congress on Surveying and Mapping, Bethesda, Maryland), pp. 175–181.

Dehne, F., A. Hassenklover, and J. Sake, 1989. Computing the configuration space for a robot on a mesh-of-processors, *Parallel Computer*, 12(2):221–231.

Dutton, G., 1989. Modeling locational uncertainty via hierarchical tessellation, *Accuracy of Spatial Databases* (M. Goodchild and S. Gopal, editors), Taylor and Francis, London, United Kingdom, pp. 125–140.

———, 1990. Locational properties of quaternary triangular meshes, *Proceedings of the 4th International Symposium on Spatial Data Handling*, 23–27 July, Zurich, Switzerland, pp. 901–910.

———, 1996. Encoding and handling geospatial data with hierarchical triangular meshes, *Proceeding of 7th International Symposium on Spatial Data Handling* (M.J. Kraak and M. Molenaar, editors), 12–16 August, Delft, The Netherlands (Faculty of Geodetic Engineering, Delft University of Technology), II:8B.34–43.

———, 1999a. Scale, sinuosity, and point selection in digital line generalization, *Cartography and Geographic Information Science*, 26(1):33–53.

———, 1999b. *A Hierarchical Coordinate System for Geoprocessing and Cartography*, Lecture Notes in Earth Sciences, Springer-Verlag, Berlin, Germany, 230 p.

Edwards, G., 1993. The Voronoi model and cultural space: Applications to the social sciences and humanities, *Spatial Information Theory: A Theoretical Basis For GIS: European Conference, COSIT'93* (A.U. Frank and I. Compari, editors), 19–22 September, Marciana Marina, Elba Island, Italy (Lecture Notes in Computer Science 716, Springer-Verlag, New York, N.Y.), pp. 202–214.

Embrechts, H., and D. Roose, 1996. A parallel Euclidean distance transformation method, *Computer Vision and Image Understanding*, 63:15–26.

Fekete, G., 1990. Rendering and managing spherical data with sphere quadtree, *Proceedings of Visualization '90*, 23–26 October, San Francisco, California (IEEE Computer Society Press, Los Alamitos, California), pp. 176–186.

Geyer C., 2000. *Voronoi Diagram on the Surface of the Sphere*, http://www.cis.upenn.edu~cgeyer/sphr-vor.html, last accessed 01 May 2000.

Gold, C.M., 1992. , The meaning of *'Neighbor,' Theories and Methods of Spatio-Temporal Reasoning in Geographic Space* (A.U. Frank, I. Campari, and U. Formentini, editors), Lecture Notes in Computing Science, 639, Springer-Verlag, Berlin, Germany, pp. 220–235.

———, 1997a. The global GIS, *Proceeding of the International Workshop on Dynamic and Multi-Dimension GIS* (Y.C. Lee and Z-L. Li, editors), 25–26 August, The Hong Polytechnic University, Hong-Kong, China, pp. 80–91.

———, 1997b. *The Development of a Global Dynamic Data Structure*, http://www.cdg.qc.ca/repert/pr318.html, last accessed 11 October 1999.

Gold, C.M., and A.R. Condal, 1995. A spatial data structure integrating GIS and simulation in a marine environment, *Marine Geodesy*, 18:213–228.

Gold, C.M., and M. Mostafavi, 2000. Towards the global GIS, *ISPRS Journal of Photogrammetry and Remote Sensing*, 55(3):150–163.

Goodchild, M.F., and Yang Shiren, 1992. A hierarchical data structure for global geographic information systems, *Computer Vision and Geographic Image Processing*, 54(1):31–44.

Goodchild, M.F., Yang Shiren, and G. Dutton, 1991 *Spatial Data Representation and Basic Operations for a Triangular Hierarchical Data Structure*, NCGIA Report 91-8, National Center for Geographic Information Analysis, University of California, Santa Barbara, California, 14 p. (available at http://www.ncgia.ucsb.edu, last accessed 10 August 1999).

Jones, C., 1997. *Geographical Information Systems and Computer Cartography*, Longman Singapore Publishers Ltd., Singapore, 319 p.

Lee, M., and H. Samet, 2000. Navigating through triangle meshes implemented as linear quadtree, *ACM Transactions on Graphics*, 19(2):79–121.

Li, C., J. Chen, and Z. Li, 1999. Raster-based methods for the generation of Voronoi diagrams for spatial objects, *International Journal of Geographic Information Science*, 13(3):209–225.

Li, Z.L., R.L. Zhao, and J. Chen, 2002. A Voronoi-based spatial algebra for spatial relations, *Progress in Natural Science*, 12(7):528–536.

Lukatela, H., 1987. Hipparchus geopositioning model: An overview, *Proceedings of the Eighth International Symposium on Computer-Assisted Cartography*, 29 March–03 April, Baltimore, Maryland (American Society for Photogrammetry and Remote Sensing/American Congress on Surveying and Mapping, Bethesda, Maryland), pp. 87–96.

———, 1989. Hipparchus data structure: Points, lines and regions in spherical Voronoi grid, *Proceedings of the Ninth International Symposium on Computer-Assisted Cartography*, 03–07 April, Baltimore, Maryland (American Congress on Surveying and Mapping/American Society for Photogrammetry and Remote Sensing, Bethesda, Maryland), pp. 164–170.

———, 2000. Ellipsoidal area computations of large terrestrial objects, *International Conference on Discrete Global Grids*, 26–28 March, Santa Barbara, California (National Center for Geographic Information Analysis and go2 Systems, Inc.), available at http://www.ncgia.ucsb.edu/globalgrids/papers, last accessed 15 November 2000.

Okabe, A., B. Boots, and K. Sugihara, 1992. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, John Wiley and Sons, Chichester, England, 532 p.

Okabe, A., B. Boots, K. Sugihara, and S. Chiu, 2000. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams, Second Edition*, John Wiley and Sons Ltd, Chichester, England, 650 p.

Otoo, E., and H. Zhu, 1993. Indexing on spherical surfaces using semi-quadcodes, *Advances in Spatial Databases 3rd International Symposium, SSD'93* (J. Abel and C.O. Beng, editors), 23–25 June, Singapore (Lecture Notes in Computer Science 692, Springer-Verlag, Singapore), pp. 509–529.

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

Roos, T., 1991. *Dynamic Voronoi Diagrams*, PhD. Thesis, Bayerische Julius Maximilians University, Wurzburg, Germany, 197 p.

Shamos, M.I., and D. Hoey, 1975. Closest point problems, *Proceedings, 16th Annual IEEE Symposium on Foundations of Computer Science*, 13–15 October, The University of California, Berkeley, California, pp. 151–162.

Snyder, J.P., 1992. An equal-area map projection for polyhedral globes, *Cartographica*, 29(1):10–21.

Su, B., Z. Li, G. Lodwick, and J.C. Muller, 1997. Algebraic Models for the Aggregation of Area Features Based upon Morphological Operators, *International Journal of Geographical Information Science*, 11(3): 233–246.

Watson, D.F., 1988. Natural neighbor sorting on the n-dimensional sphere, *Pattern Recognition*, 21(1):63–67.

———, 1998. Modemap: *An Implementation of Natural Neighbor Interpolation on the Sphere*, http://members.iinet.net.au/~watson/modemap.html, last accessed 20 September 1998.

White, D., and A.J. Kimmerling, 1998. Comparing area and shape distortion on polyhedral based recursive tessellations of the sphere, *International Journal of Geographical Information Science*, 12(8):805–827.

White, D., J. Kimmerling, and W.S. Overton, 1992. Cartographic and geometric components of a global sampling design for environment monitoring, *Cartography & Geographical Information Systems*, 19(1):5–22.

Wickman, F.E., and E. Elvers, 1974. A system of domains for global sampling problems, *Geografiska Annaler*, 56(3/4):201–212.

Wright, D., and M.F. Goodchild, 1997. Data from deep: Implications for the GIS community, *International Journal of Geographical Information Science*, 11(6):523–528.

Yang, W., and C. Gold, 1996. Managing Spatial Objects With the VMO-Tree, *Proceeding of 7th International Symposium on Spatial Data Handling* (M.J. Kraak and M. Molenaar, editors), 12–16 August, Delft, The Netherlands (Faculty of Geodetic Engineering, Delft University of Technology), pp. 15–31.

(Received 29 January 2002; accepted 17 May 2002)