Title
Spatial Data Representation and Basic Operations for a Triangular Hierarchical Data Structure (91-8)

Authors
Goodchild, Michael F.
Shiren, Yang
Dutton, Geoffrey

Publication Date
1991-04-01

Spatial Data Representation and Basic Operations for a Triangular Heirarchical Data Structure

April 1991

Michael F. Goodchild, Yang Shiren, and Geoffrey Dutton

NCGIA
Department of Geography
University of California at Santa Barbara
Santa Barbara, CA 93106

National Center for Geographic Information and Analysis

Report 91-8

SPATIAL DATA REPRESENTATION AND BASIC OPERATIONS FOR A TRIANGULAR HIERARCHICAL DATA STRUCTURE

Michael F. Goodchild[1] *Yang Shiren*[1] *Geoffrey Dutton*[2]

Abstract

A triangular hierarchical data structure has been proposed as the basis for a global geographical information system. In this paper we briefly review one such scheme based on recursive subdivision of an octahedron, and conversion algorithms to and from latitude/longitude. Schemes for representing point, line and area objects on the earth's surface are described. We present algorithms for identifying triangle neighbors, region filling and object dilation.

Introduction

A triangular hierarchical data structure (THDS), based on triangular quadtree decomposition of an octahedron fitted to the earth, has been proposed for use in a global geographical information system [1]-[5]. Algorithms for conversion of triangular addresses to and from Cartesian coordinates, for calculation of triangle area assuming a spherical earth, and for finding the neighbors of triangles have been developed [6]. In this article, the representations of points, lines and regions in THDS, methods for filling a region with color, calculation of the area of a region with an ellipsoidal earth, and an algorithm for dilation of points, lines and regions in the triangular data structure are introduced. These operations are essential for a data structure to be used in a practical GIS. We review the conversion algorithms briefly in section 1. In section 2, the concept of neighbor triangles is extended based on the neighboring triangle finding algorithm proposed. In section 3, a chain encoding of THDS is suggested for representation of lines and the borders of regions. A region filling algorithm and the expression for calculating the area of decomposed triangles with an ellipsoidal earth are introduced in sections 4 and 5 respectively. THDS is a quadtree data structure; therefore, a region can be represented by nodes and leaves and this representation will be described in section 6. Dilation of points, lines and regions is discussed in section 7. High-level objects that unite THDS subtrees are described in section 8, and their properties are discussed. The algorithms developed and the experiments described in this paper show that these basic operations can be implemented very well in the THDS.

1.Representation of Points and Lines

A point on the earth is usually defined by its longitude λ and latitude ϕ. In the

[1] National Center for Geographic Information and Analysis and Department of Geography, University of California at Santa Barbara, CA 93106 [2] Spatial Effects, 150 Irving St, Watertown, MA 02172

triangular hierarchical data structure, the position of a point is identified by the centroid of a decomposed triangle. The accuracy of estimates of the locations of points improves with increasing amounts of decomposition. At the k-th level of decomposition, the *triangle address* A is represented by

$$A = a_0 a_1 a_2 a_3 \ldots\ldots a_k$$

where a_1 to a_k are k quaternary digits, while a_0 is an octal digit representing the initial octahedral decomposition at level 0. Since the earth is divided into 8 equilateral spherical triangular faces as shown in Figure 1, an octal digit is assigned to a_0 for addressing each face as follows:

$a_0 = 0, 1, 2, 3$	for	$90^\circ \geq \phi \geq 0^\circ$	i.e., for the Northern hemisphere
$a_0 = 4, 5, 6, 7$	for	$0^\circ > \phi \geq -90^\circ$	i.e., for the Southern hemisphere

and

$a_0 = 0, 4$	for	$90^\circ > \lambda \geq 0^\circ$
$a_0 = 1, 5$	for	$180^\circ > \lambda \geq 90^\circ$
$a_0 = 2, 6$	for	$-90^\circ > \lambda \geq -180^\circ$
$a_0 = 3, 7$	for	$0^\circ > \lambda \geq -90^\circ$

We call these eight faces octants 0 to 7 respectively.

In octant 0, the relationships between longitude/latitude (λ, ϕ) and Cartesian coordinates (x, y) are as follows:

$$x = \frac{2^n}{\pi}[\phi + 2\lambda(1 - \frac{2}{\pi}\phi)], \qquad y = \frac{2^n\sqrt{3}}{\pi}\phi \tag{1-1}$$

or

$$\lambda = \frac{\pi}{2^{n+1}}\frac{\sqrt{3}x - y}{\sqrt{3} - 2^{1-n}y}, \qquad \phi = \frac{\pi}{2^n\sqrt{3}}y \tag{1-2}$$

where n is the maximum level of triangular decomposition.

For conversion of triangle addresses to Cartesian coordinates, we first transform the triangle address a_i $(i = 1, 2, .., k)$ to Cartesian coordinates in octant 0, then transform the Cartesian coordinates into longitude and latitude using the expressions (1-2) and determine the octant according to the value of a_0. The expressions for conversion of triangle addresses to Cartesian coordinates in octant 0 are

$$x_k = [\,2^k + \sum_{i=1}^{k}[\,(-1)^{a_{i2}+1} a_{i1} 2^{k-i}\,]]\; 2^{n-k-1} \tag{1-3}$$

$$y_k = \frac{2^k + \sum_{i=1}^{k}[\,(-1)^{NZ_i + a_{i1}}\,[2(\bar{a}_{i1} \cap a_{i2}) + a_{i1}]\; 2^{k-i}\,]}{\sqrt{3}}\; 2^{n-k-1} \tag{1-4}$$

where (x_k, y_k) are the Cartesian coordinates represented by a triangular address at the k -th level, and

$$NZ_k = \sum_{i=1}^{k-1} [\overline{a}_{i1} \cap \overline{a}_{i2}] \quad (1\text{–}5)$$

denotes the number of zeroes in a_1 to $a_{k-1.}$ a_{i1} and a_{i2} are two binary digits representing each quaternary digit a_i and $\overline{a}_{i1}, \overline{a}_{i2}$ are the logical negative of a_{i1}, a_{i2}. These equations are used to transform a triangle address to the Cartesian coordinates of its centroid. For a point with given longitude and latitude, the triangular address a_1 to a_k of the containing triangle can be obtained by a recursive approximation process [6]. a_0 is determined from the containing octant. The expressions (1-1) and (1-2) are simple arithmetic calculations and expressions (1-3) and (1-4) are bitwise operations. Conversions between triangular address and longitude and latitude are quite simple and efficient to compute.

2. Neighboring Triangles in THDS

The adjacency or connectedness of data elements is one of the most useful aspects of many data structures used in GIS. In the THDS, we term neighbors with shared edges *direct neighbors* and with common vertices *indirect neighbors*. An algorithm has been proposed to find the three direct neighbors of a triangle inside an octant [6]. For the data structure of a global GIS, neighboring triangles which are located in two adjacent octants must also be considered. From Figure 2 we can see that if a triangle has edge(s) at the border of an octant, the triangle will have direct neighbor(s) in its neighboring octant(s); if a triangle has vertices(s) at the border of an octant, the triangle will have indirect neighbor(s) in its neighboring octant(s). There are in total 13 different neighbor patterns depending on the location of the triangle. They can be classified into 5 categories: *inside, edge, sub-edge, corner* and *sub-corner triangles* and can be defined as follows:

(1). A triangle is called an *inside triangle* if all its direct and indirect neighbors are within the same octant. An inside triangle has three direct neighbors and nine indirect neighbors

(2). A triangle is called an *edge triangle* if it has exactly one direct neighbor in the adjacent octant.

(3). A triangle is called a *sub-edge triangle* if it has exactly three indirect neighbors in the adjacent octant.

(4). If a triangle has exactly two direct neighbors in adjacent octants, it is called a *corner triangle.*

(5). If a triangle has exactly six indirect neighbors in adjacent octants, it is called a *sub-corner triangle.*

Figure 2 shows the inside, edge, sub-edge, corner and sub-corner triangles in an octant decomposed to the third level. The neighbor relation of a triangle can be determined by the location of the triangle in the octant which can be easily identified by the pattern of its triangular address (ignoring the first digit a_0) as follows (see Figure 3 for illustration):

(1). In the decomposition of a triangle, its top, left and right triangles are numbered 1, 2 or 3 respectively. At the k-th level, the triangles at left edges are either top or left triangles of their parent ($k-1$ th level) triangles, which are also left edge triangles. Therefore, their addresses must consist of digits "1" or "2" only. The addresses of right and top edge triangles consist of digits "1" and "3", and "2" and "3" respectively.

(2). A left sub-edge triangle's address consists of a sequence identifying a left edge triangle at level $\le k-1$, followed by exactly one 0 and optionally a series of 2s. Similar rules identify right and top sub-edge triangles, substituting optional series of 3s and 1s respectively.

(3). A top corner triangle has the properties of a left edge *and* a right edge triangle; a left corner triangle has the properties of a left edge *and* a top edge triangle; a right corner triangle has the properties of a right edge *and* a top edge triangle.

(4). A sub-top corner triangle has the properties of a sub-left edge *and* a sub-right edge triangle; a sub-left corner triangle has the properties of a sub-left edge *and* a sub-top edge triangle; a sub-right corner triangle has the properties of a sub-right edge *and* a sub-top edge triangle.

(5). If a triangle is not an edge, sub-edge, corner or sub-corner triangle, it is an inside triangle.

All triangles have three direct neighbors. There are seven indirect neighbors for corner triangles and nine indirect neighbors for other triangles. We use the codes t, l and r to represent the three direct neighbors with common top, left and right edges for a given triangle inside an octant; the code T to represent the direct neighbor of a top edge triangle lying in the adjacent octant, W the direct neighbor of a left edge triangle in the adjacent octant, and E the direct neighbor of a right edge triangle in the adjacent octant (Figure 3). The direct neighbors can be found by the following algorithm:

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

$$
\begin{aligned}
t &= t_1, t_2, t_3, \ldots\ldots, t_k, \\
l &= l_1, l_2, l_3, \ldots\ldots, l_k, \\
r &= r_1, r_2, r_3, \ldots\ldots, r_k, \\
T &= T_1, T_2, T_3, \ldots\ldots, T_k, \\
E &= E_1, E_2, E_3, \ldots\ldots, E_k, \\
W &= W_1, W_2, W_3, \ldots, W_k.
\end{aligned}
$$

First, the triangular address U is copied to the data strings t, l, r, T, E and W. Then:

(1). To find the top neighbor inside an octant, t_0 does not change and for $i=k$

$$
\begin{aligned}
t_i &= 2 \qquad \text{if} \qquad t_i = 2 \\
t_i &= 3 \qquad \text{if} \qquad t_i = 3
\end{aligned}
$$

and process t_{i-1}. On the other hand if $t_i \ne 2$ and $t_i \ne 3$, then

$$
\begin{aligned}
t_i &= 1 \qquad \text{if} \qquad t_i = 0 \\
t_i &= 0 \qquad \text{if} \qquad t_i = 1
\end{aligned}
$$

and the search finishes.

(2). To find the left neighbor inside an octant, l_0 does not change and for $i=k$

$l_i = 3$	if	$l_i = 1$
$l_i = 1$	if	$l_i = 2$

and process l_{i-1}. On the other hand if $l_i \neq 1$ and $l_i \neq 2$, then

$l_i = 2$	if	$l_i = 0$
$l_i = 0$	if	$l_i = 3$

and the search finishes.

(3). To find the right neighbor inside an octant, r_0 does not change and for $i=k$

$r_i = 2$	if	$r_i = 1$
$r_i = 1$	if	$r_i = 3$

and process r_{i-1}. On the other hand if $r_i \neq 1$ and $r_i \neq 3$, then

$r_i = 3$	if	$r_i = 0$
$r_i = 0$	if	$r_i = 2$

and the search finishes.

(4). To find the direct neighbor with a common top edge at the top border of the octant

$T_0 = T_0 + 4$	if	$T_0 \leq 3$
$T_0 = T_0 - 4$	if	$T_0 \geq 4$
T_i does not change	for	$i = 1$ to $i = k$

(5). To find the direct neighbor with a common left edge at the left border of the octant

$W_0 = W_0 - 1$	if	$W_0 \neq 0$	or	$W_0 \neq 4$
$W_0 = W_0 + 3$	if	$W_0 = 0$	or	$W_0 = 4$
W_i changes to 3	if	$W_i = 2$	for	$i=1$ to $i=k$

(6). To find the direct neighbor with a common right edge at the right border of the octant

$E_0 = E_0 + 1$	if	$E_0 \neq 3$	or	$E_0 \neq 7$
$E_0 = E_0 - 3$	if	$E_0 = 3$	or	$E_0 = 7$
E_i changes to 2	if	$E_i = 3$	for	$i=1$ to $i=k$

The *direct neighbor finding algorithms* are simple bitwise operations and it has been proved that the average number of bitwise operations for $t, l,$ and r is less than 2 [6], and the average number of bitwise operations for $T, E,$ and W is equal to $k/2$.

We use the symbol

$\alpha = right(\beta)$; $\gamma = WEST(\alpha)$;

to define triangle α as a right neighbor of β (the right edge of β is the left edge of α) while γ is a *WEST* neighbor of α with common edge at the left border of the octant of α; *East,* *left, top* and *Top* are defined similarly.

There are many possible indirect neighbor configurations depending on the location of a triangle in an octant as shown in Figure 3. They can be found through the direct neighbors of a triangle if the location of a given triangle is identified. For example, the indirect neighbor h of an inside triangle U can be found as

$$\alpha = top(U); \qquad h = left(\alpha)$$

where α is an intermediate address.

3. Representation of lines in THDS

A line is one of the most important entities in a spatial database. For example, hydrology, transportation, terrain relief and region boundaries are commonly represented by lines. In order that a line can be represented correctly and compactly, it should satisfy the following requirements:

(1). A line should be represented in compact form to save computer storage; this includes approximation by a series of straight lines or curves and stepwise representations such as chain codes (Pavlidis 1982).

(2). A line should be able to be interpolated to achieve higher levels of detail or filtered to achieve lower levels for display or other manipulations.

(3). A region border represented by a line should enable further processing such as area filling or area and perimeter calculation.

(4). A line representation should be simple to display.

In this section, we introduce algorithms for interpolation of lines, conversion of lines to and from chain codes and smoothing of lines in the THDS. Since the conversion of triangular addresses to and from longitude/latitude (or Cartesian) coordinates, and finding neighbors of triangle addresses in the THDS are quite efficient, the interpolation of a line can be implemented by the following steps:

(1). Convert the triangle addresses to longitudes and latitudes, then interpolate them by linear or B-spline interpolation functions.

(2). Transform the interpolated longitude and latitude values back to triangular addresses.

(3). If two consecutive triangles are not neighbors, continue to interpolate their longitudes and latitudes and transform them to triangular addresses until consecutive pairs of triangles are neighbors or have the same triangular address.

(4). If adjacent triangles have the same triangular address, remove the duplicated addresses.

The triangular addresses thus obtained to represent a line will be a series of direct or indirect neighbors. Figures 4(a) and (b) show the parallels and meridians of the earth at 15^o intervals, drawn by connecting the centroids of neighboring triangles at levels 6 and 10 respectively.

Chain code representation of lines [7] has been widely used in computer cartography and geographic information systems for data compression. In a chain code, the vector joining two consecutive points is assigned a symbol from the members of a finite set of codes. The simplest chain code in the THDS uses a sequence of vectors concatenating centers of directly connected triangles. The 3-code representation of a border line of a region is shown in Figure 5(a). However, lines represented by 3-code are too erratic and triangles in adjacent octants cannot be connected. On the other hand, the set of indirect neighbors is too large to be the basis of a useful chain code. Therefore, we propose that only the east and west oriented indirect neighbors with codes e and w, and three additional codes T, E and W for connecting triangles in adjacent octants be added. Using this 8-code, any line on the earth can be represented by codes t, l, r, e, w, T, E, W and each code can be encoded by three binary bits. We call this code 8-code the *chain code* of THDS. For example, 00001, 00002 and 00003 are t, l and r neighbors, and 00031 and 00021 are e and w neighbors of triangle 00000 respectively. The T neighbor of 02323 is 42323, the E neighbor of 01313 is 11212 and the W neighbor of 01212 is 31313. We term the set of triangles forming a boundary *8-connected* if it can be represented in 8-code. The 8-code representation of a border line of a region is shown in Figure 5(b). A line in point mode can be converted to chain code and vice versa using interpolated direct and indirect neighbors.

When the resolution of data is reduced, a line may exhibit unnecessary "zig-zags" or even acquire spikes. For example, the meander and sharp bend of a river may change to spikes after reducing the decomposition level. The spikes usually have to be removed and zig-zags in a line may need smoothing. In chain code, the zig-zag form of lines is demonstrated as consecutive $l\,l\,l \ldots l$ or $r\,r\,r \ldots r$. The line can be smoothed by replacing them with an appropriate number of $w\,w \ldots w$ or $e\,e \ldots e$. The spikes in a line appear as a combination of two consecutive $t\,t$ or a series of consecutive $..l\,r\,l\,r\,l\,r..$ or both. The spikes can be eliminated by removing these consecutive codes. Figure 6(b) shows the border lines of Figure 6(a) with removal of zig-zag and spikes. Tables 1 (a) and (b) illustrate the 28 pairs of latitude and longitude values of the coastline of Great Slave Lake in Canada interpolated to 43 triangular addresses at level eight, converted to connected neighbors, smoothed and translated to 8-code. Figure 7 shows the coastlines of the western hemisphere with a total of 8248 longitude/latitude pairs interpolated and converted to 15993 neighboring triangles at level 8 and displayed orthographically with rotation of the earth at -90^o around the Polar axis and 30^o around an axis in the WE direction. It is interesting to note that 83.74% of the 15993 neighboring triangles in Figure 7 are direct neighbors t, l and r.

4. Region Filling in the Triangular Hierarchical Data Structure

Filling a closed region is important for representation of a region by nodes and leaves, for calculating the area of a region and for overlay of two or more data layers in a GIS. In this section, we introduce an algorithm for region filling in the THDS where a region is defined by its border line represented by a series of triangular addresses. We assume that the boundary of the region is 8-connected.

Two commonly used approaches for determining the region within a given border line are the polygon-based and parity check-based methods (Pavlidis 1982). The parity check method is preferred since it can be implemented more easily in THDS.

In a region filling algorithm, holes inside a region should be excluded but the border line should be included in the region and in the area calculated. It is convenient to arrange the data set of boundaries in row order since the area of a triangle at a given level of decomposition is a function of latitude which can be identified by the row number.

The algorithm fills the region row by row starting from the row (or scan line) with lowest absolute latitude value. Within a row, the points inside the region are filled from west to east. The region filling algorithm consists of the following steps:

(1). Assign a flag to each boundary triangle. The flag is determined by the relation of the point to its preceding and succeeding points. There are five different flags:

Flag *a* is assigned to a point which has no *east* or *west* neighbor, its preceding triangle is a *top* neighbor and its succeeding triangle is a *left* or a *right* neighbor, or its preceding triangle is a *left* or a *right* neighbor and its succeeding triangle is a *top* neighbor. We call these *single intersection points.*

Flag *b* is assigned to a point which has no *west* neighbor. If the triangle is upward, its preceding triangle is a *top* neighbor and its succeeding triangle is an *east* neighbor; if the given triangle is downward, its preceding triangle is a *left* or a *right* neighbor and its succeeding triangle is an *east* neighbor. We call these *upward tangent entry points.*

Flag *c* is assigned to a point which has no *west* neighbor; if the triangle is upward, its preceding triangle is a *left* or a *right* neighbor and its succeeding triangle is an *east* neighbor; if the given triangle is downward, its preceding triangle is a *top* neighbor and its succeeding triangle is an *east* neighbor. We call these *downward tangent entry points.*

Flag *d* is assigned to a point which has no *east* neighbor; if a given triangle is upward, its preceding triangle is a *west* neighbor and its succeeding triangle is a *top* neighbor; if the given triangle is downward, its preceding triangle is a *west* neighbor and its succeeding triangle is a *left* or a *right* neighbor. The given triangle has no *east* neighbor. We call these *downward tangent exit points.*

Flag *e* is assigned to a point that it has no *east* neighbor; if a given triangle is upward, its preceding triangle is a *west* neighbor and its succeeding triangle is a *left* or a *right* neighbor; if the given triangle is downward, its preceding triangle is a *west* neighbor and its succeeding triangle is a *top* neighbor. We call these *upward tangent exit points.*

These points and their forms are listed in Figure 8 and 9. All other boundary triangles which do not belong to the above categories are called simple points.

(2). Convert the triangular addresses into latitudes and longitudes $\phi[i]$ and $\lambda[i]$ $(i=1,2,..,n)$ and sort the data sets in ascending order of ϕ, where n is the number of points of the border line represented by 8-code.

(3). Sort the data array with given ϕ in ascending order of λ. We use a pair of two-dimensional arrays $A[i][j]$ and $flag[i][j]$. The subscript i denotes the number of the row (or scan line) calculated from the lowest row of the region; the subscript j is the number of points in the i *-th* row.

(4). In order that a region with complex topology can be filled correctly, we use two counters n_1 and n_2 that are set to zero at the beginning of a row. Then

- Increase the number n_1 by 1 for each upward tangential entry point,
- Increase the number n_2 by 1 for each downward tangential entry point.
- Decrease the number n_1 by 1 for each downward tangential exit point.
- Decrease the number n_2 by 1 for each upward tangential exit point.

(5). An *even/odd* flag *eo* is used to determine whether the area between points in a scan line is *inside* or *outside* the region. *eo* is set to zero at the beginning of a row; we add 1 to the *eo* flag when there is

- a single intersection point
- a downward tangential exit point with $n_1 \geq 1$ (i.e., downward tangential entry points already existing in the row)
- an upward tangential exit point with $n_2 \geq 1$ (i.e., upward tangential entry points already existing in the row)

(6). Copy the border points of the region or holes to the corresponding rows so that they can be included in the region.

(7). If the *eo* flag is odd and the triangular address of the point is not equal to the triangular address of its succeeding point, fill the scan line by *west* or *WEST* neighbors, by comparing the point's triangular address with the triangular address of the succeeding point. Terminate the filling process if the two triangular addresses are equal to each other.

The region shown in Figure 5 was filled with level 4 triangles using the region filling algorithm. The total number of triangles enclosed by the border line (including the border line) is 127. Figure 10 shows the South and North American continents in Figure 7 filled with triangles at level 8. At this level the lakes inside the continents are excluded as expected. The moire in Figure 10 is due to the interference between the resolution of the display screen and the fill pattern. Orthographical projection of the data changes the scale with sinusoidal functions which exaggerates the moire effect.

When the filling algorithm is applied to a region with the North or South Pole inside the region, the region should be divided into two or more sub-regions. Each sub-region should have a vertex at the Pole.

5. Area of a Region on a Spherical or Ellipsoidal Earth

It has been shown that for a spherical earth, the area of a triangle can be calculated by the following expression when the level of decomposition is high [6]

$$\Delta A_\phi \approx k \frac{\cos\phi}{\frac{\pi}{2} - \phi} = k \frac{\sin\chi}{\chi} = k \; sinc\, \chi \qquad (5\text{–}1)$$

where

$$sinc\,\chi = \frac{\sin\chi}{\chi}, \qquad k = \frac{\pi^3 R^2}{2^{2n+4}}, \qquad \chi = \frac{\pi}{2} - \phi.$$

R is the average radius of the earth, and n is the level of decomposition.

For an ellipsoidal earth, the expression for calculating the area of a triangle at level k becomes

$$\Delta A_\phi = k \, [\, sinc\,\chi + \frac{r^2}{8R^2} sinc\,\chi + \frac{r^2}{8R^2} \frac{\sin 3\chi}{\chi} \,]. \qquad (5\text{–}2)$$

where R is the major axis and r is the difference of major and minor axes of the ellipsoid. The derivation of expression (5-2) is given in the Appendix. The ratio of triangle area at $\phi = \frac{\pi}{2}$ to area at $\phi = 0$ is

$$\frac{\Delta A_{\frac{\pi}{2}}}{\Delta A_0} = \frac{\pi}{2} \, [\, 1 + \frac{r^2}{2R^2}]$$

For comparison, the ratio is $\frac{\pi}{2}$ for a spherical earth, and about 0.313 percent larger for a spherical earth.

Expressions (5-1) and (5-2) imply that at a high level of decomposition, the area of a decomposed triangle is a function of latitude ϕ, which is related to the Cartesian coordinates by

$$\phi = \frac{\pi}{2^n \sqrt{3}} y = \frac{\pi \, v}{3 \times 2^{n+1}} \qquad (5\text{–}3)$$

The values of v in the expression are related to the row number and are sorted in ascending order in the region filling algorithm described in the previous section. The area of a row in a region is equal to the number of points within the region in the row multiplied by ΔA_ϕ. The total area of the region is the sum of the areas of all rows in the region. Since holes in the region are removed in the region filling algorithm, the area of holes is also excluded in the area calculation.

The row-by-row representation of a region is useful for displaying the region in raster-like display devices. In the triangular hierarchical data structure, the areas of decomposed triangles are a function of latitude or row number. Resampling points in each row using expressions (5-1) or (5-2) is necessary if the display of a region is to be adjusted for area distortion. This is a one-dimensional resampling process widely used in digital image processing and will not be discussed here in detail.

6. Node Representation in the Triangular Hierarchical Data Structure

In the THDS, a filled region can be constructed by successive decomposition of a triangle covering the entire region until we obtain a set of triangles labeled either 1 or 0 to represent the triangles inside or outside the region. This process can be represented by a tree of out-degree 4 similar to that of the quadtree [10]-[12], [13]. We call these triangles *nodes* in THDS; they may be leaves of the tree. Region representation by nodes is a more compact form of data structure. Figure 11(b) is a tree representation of the region of Figure 11(a), where *black* and *white* (or 1 and 0) nodes represent triangles within and outside the region respectively. When the region is filled with "1", the nodes can be found by merging the points labeled with "1" starting from the highest (most decomposed) level. The points that cannot be merged belong to leaves. The nodes at the next highest level are merged again. Figure 12 shows the node representation of the earth land area with orthogonal projection at different orientations.

7. Dilation algorithm in THDS

The purpose of a dilation is to expand a point, line or region to a given distance. In GIS, answers to queries such as "find the area where the distance to a shopping center is no more than a given distance", "what is the area of all wheat fields within a given distance of the river ?" and "find the residential area within a given distance of the university campus" etc., are often implemented by dilation algorithms. Dilation in a GIS context is also called "buffering". Applying a dilation operation to a point, line or region produces a region with uniform distance between its border and the original point, line or region. Triangular, like rectangular quadtrees cannot be counted upon to generate exact dilations; the results will be as exact as the level of decomposition allows. The neighboring triangle-finding algorithm described in section 2 can be used directly for dilation since the dilation requires finding all neighbors of a point, line or region.

The neighbor-finding algorithm identifies all directly and indirectly connected triangles which surround the point and form a region. If this procedure is repeated, i.e., the neighboring triangles of all triangles in the region are found, and duplicate triangles are removed, the region is expanded. This procedure can be iterated until the region expands to the required size. 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. However, the following topological and metrical properties of the region are preserved.

(1). The region generated is connected and contains no holes.

(2). The region dilated each time is a stripewise region surrounding the old region with width $\frac{l \times \sqrt{3}}{2}$.

(3). The distance between the border of the dilated region and the nearest edges of a given triangle (point) varies from nl to $\frac{n\sqrt{3}l}{2}$, a factor of 0.866, which is larger than in rectangular quadtrees where the ratio of edge to diagonal of a square is 0.707.

In the THDS, a line is represented by a series of points and a region is represented by its boundary trace. Dilation of a line or region can be done simply by dilation of all the points describing the line or border line of a region and then removing all duplicate triangles. The computing time for removing duplicate triangles is proportional to the square of the number of points. However, it would be easy to derive algorithms that avoid producing duplicates.

8. Higher Order Locations

THDS elements are tiles in a "quaternary triangular mesh" (QTM) repeated eight times across the planet [2][3]. "Geodetic control" for converting coordinates to THDS addresses is furnished by the octahedral geometry, method of subdivision (see section 1), and figure of the Earth (ellipsoid) employed. Having established these bases, all georeferencing operates with respect to fixed points defined by this tessellation: the vertices of the octahedron's triangles, and those of subsequent decompositions. Until now, we have paid little attention to these points and their properties. In this section, we examine the nature of these locations and their potential influence on THDS spatial data handling.

While the vertices of THDS tiles are arguably more "primitive" than the tiles themselves (being of topological dimension 0, while tiles are dimension 2), they are equally worthy of study. Not only do vertices control the geometry of THDS, but they determine a number of its topological properties. This in part derives from an intrinsic duality between the vertices and tiles of a THDS hierarchy.

When a triangle subdivides, three "outer" tiles and one "inner" tile appear. Each of the outer tiles, labeled 1, 2 and 3, are associated with one of the parent's vertices, while the inner tile (0) touches no prior vertex. Each parent vertex, in turn, is associated with (surrounded by) six child tiles (except for the six initial octahedral vertices, which are surrounded by four tiles each). At a higher level of abstraction, one can say that any geographic location occupying a 1, 2 or 3 tile may be represented by the corresponding outer vertex of that tile. That is, a vertex becomes an *alias* for all points lying within a hexagonal region of space centered at the vertex. This region is, of course, composed of six tiles in six different subtrees of the THDS hierarchy. Dutton [4] describes the structure and properties of these regions, termed *attractors* because they are dominated by a locus of attraction (a THDS grid node). This duality is also explored by Fekete [8] as a logical consequence of the tile numbering scheme employed by his sphere quadtree (SQT) which is essentially the same as that proposed by Dutton (Fekete numbers SQT's central tiles 4, while Dutton numbers them 0; in addition, Fekete roots his hierarchy in an icosahedron, rather than an octahedron).

Using Dutton's numbering scheme (which differs from the one described above for THDS; see figure 14), it can be shown that the quaternary digit assigned to an outer tile (1, 2 or 3) will be the same for all tiles that meet at a given vertex, regardless of which level of the hierarchy they occupy. That is, once a vertex comes into being, it will have a basis number equal to 1, 2 or 3. Subsequent subdivisions involving that vertex will generate tiles whose ID's terminate with the basis number. For example, if the North and South Poles are each assigned a vertex ID of 1, all tiles which touch either pole will also be assigned ID's of 1. This yields THDS addresses for polar tiles of n1, n11, n111,..., where n is an octant ID from 0 to 3 (North) or 4 to 7 (South); these are two of the six 4-neighbor cases involving octahedral vertices. Each set of four k-digit triangles surrounding a pole defines a square region that shrinks around its centerpoint, its diameter decreasing by a factor of two at each level (in non-axial locations, such regions are of course concentric hexagons, not squares). Figure 14 illustrates the attractors occupying one THDS octant to three levels of detail. This pattern propagates to all higher levels and across octant boundaries; the half-hexagons along octant edges are matched by identical patterns in adjoining octants.

The black triangles in Figure 14 identify central tiles, coded as 0 in each level. At any level of detail, 0-tiles cover 25 percent of the surface of the planet. Such tiles are not adjacent to any parent vertex, and therefore have no attractor at their specific level of detail. Nearly all of them do fall within some attractor at a low level of detail, and therefore can be assigned to one of these attractors. They can also be considered as "default locations" for all points falling within a non-decomposed triangle (0-tiles have the same centroid as their parents).

Attractors generalize THDS addresses to discrete polygonal realms that span adjacent subtrees. Each tile (unless its address ends with 0) is a portion of one and only one attractor. All points occupying an attractor can be aliased to it, just as all points within a given tile alias to its address. Both forms of georeferencing serve as 2D hashing functions for coordinate points: one evaluates to a THDS vertex, the other to one of its tiles. Although they require computing an extra level of indirection, attractors are better representations of spatial neighborhoods, and offer a solution to a dilemma common to all region quadtree schemes: how to associate arbitrarily close locations that happen to fall into different subtrees due to the (arbitrary) alignment of quadtree tiles to the spatial manifold they partition.

Algorithms for assigning attractors of THDS addresses have been developed to exploit the properties identified in this section. These methods include procedures to translate tile addresses to and from Dutton's (or Fekete's) address ordering and those that this paper describes for THDS. Identifying the triangles that comprise an attractor involves a variant of the neighbor-finding algorithms described in section 2 (there is also the property, mentioned by Fekete, that within an octant, the addresses of triangles incident to a node differ by exactly one digit; the converse, however is not true). The method does not in itself specify how attractors themselves should be numbered; one could use Fekete's approach, in which grid nodes are assigned the lexically lowest triangle ID incident to them. In comparison, the THDS addressing model appears to be more computationally efficient for encoding data and manipulating tiles, while the other schemes provide the generality required to deal with attractors; we believe that a complete synthesis of these addressing models is possible, one which would preserve the strengths of each approach and eliminate the need for address conversion.

9. Conclusion

To effectively store, retrieve and analyze spatial data on a global scale, alternatives to current GIS data models are urgently needed. If we are to meet the challenge of supporting global environmental science, we must retool GIS database technology at a rather basic level. This is because so many of the "features", "layers", "partitions" and "projects" of GIS's administrative databases are modeled as a planar, Cartesian chunks of the world, represented as maps. Although many systems can perform transformations between projections and into latitude and longitude, this is usually only done to "register" coverages by fitting coordinates to a preferred planar projection. A few GISs store spherical coordinates in their databases, and thus are in principle capable of working on a global scale. But non-isotropism of geographic coordinates, the expense of procedures needed to manipulate such data, and the ambiguities inherent in attempting to positively identify points and their loci (even in latitude and longitude) will continue to confound GIS applications, especially when data quality information is lacking or goes unused. Casting coordinate data into hierarchical planetary tesselations can address many of the constraints and uncertainties that limit the usefulness of current GIS technology: any planetary location can be canonically identified, regardless of where it is or the scale at which it is captured; features could be identified and integrated with certifiable accuracy; different datasets (suitably qualified) could be merged relatively easily.

The THDS data model is a region quadtree composed of triangles which may be aggregated into higher-level hexagons. The polyhedral hierarchy can be represented in a linear, scale-specific address space for spheroidal geographic data. The model consists of a forest of 8 quadtrees representing the faces of an octahedron, and provides consistent georeferencing anywhere on a planet at a hierarchy of scales. Specific aspects of THDS have been set forth: tile numbering, neighbor identification, chain encoding, region filling, area computation, dilation procedures, and higher-level structures called *attractors*. The latter construct may offer a way to handle aliasing artifacts exhibited by adjacent subtrees of region quadtrees, and can aid in identifying and even preventing the formation of polygonal "slivers" when overlaying cartographic features in the THDS realm.

Acknowledgement

The National Center for Geographic Information and Analysis is supported by the National Science Foundation, grant SES 88-10917.

References

[1]. G. Dutton, Geodesic modeling of planetary relief, *Cartographica*, 21: 188 - 207, 1984

[2]. ---, Planetary Modeling via hierarchical tessellation, *Proceedings, AutoCarto ASPRS/ACSM* Fall Church VA 462-471, 1989

[3]. ---, Modelling locational uncertainty via hierarchical tessellation, M.F.Goodchild and S.Gopal, eds., *Accuracy of Spatial Database* London, Taylor and Francis, 125-140, 1989

[4]. ---, Locational properties of Quaternary Triangular Meshes, *Proceedings, Fourth International Symposium on Spatial Data Handling* University of Zurich, 1990

[5]. ---, Polyhedral hierarchical tesselations: the shape of GIS to come, *Geo Info Systems* 49-55, February, 1991

[6]. M. F. Goodchild and Yang Shiren, A hierarchical spatial data structure for a global geographic information system. To appear in *Computer Vision, Graphics and Image Processing*

[7]. H. Freeman, On the encoding of arbitrary geometric configurations, *Transactions on Electronic Computers* EC10: 260-268, 1961

[8]. G. Fekete, Rendering and managing spherical data with sphere quadtrees, *Proceedings of Visualization 90* Association for Computing Machinery, New York, 1990

[9]. T. Pavlidis, *Algorithms for Grphics and Image Processing,* Computer Science Press, Rockville, MD., 1982

[10]. H. Samet, The quadtree and related hierarchical data structures, *Computing Surveys* 16:187-260, 1984

[11]. ---, *The Design and Analysis of Spatial Data Structures* Reading MA, Addison-Wesley, 1990

[12]. ---, *Applications of Spatial Data Structures: Computer Graphics, Image Processing and GIS,* Reading MA, Addison-Wesley, 1990

[13]. G. Tobler, W. R. and Zi-tan Chen, A quadtree for global information storage, *Geographical Analysis* 18:360-71, 1986

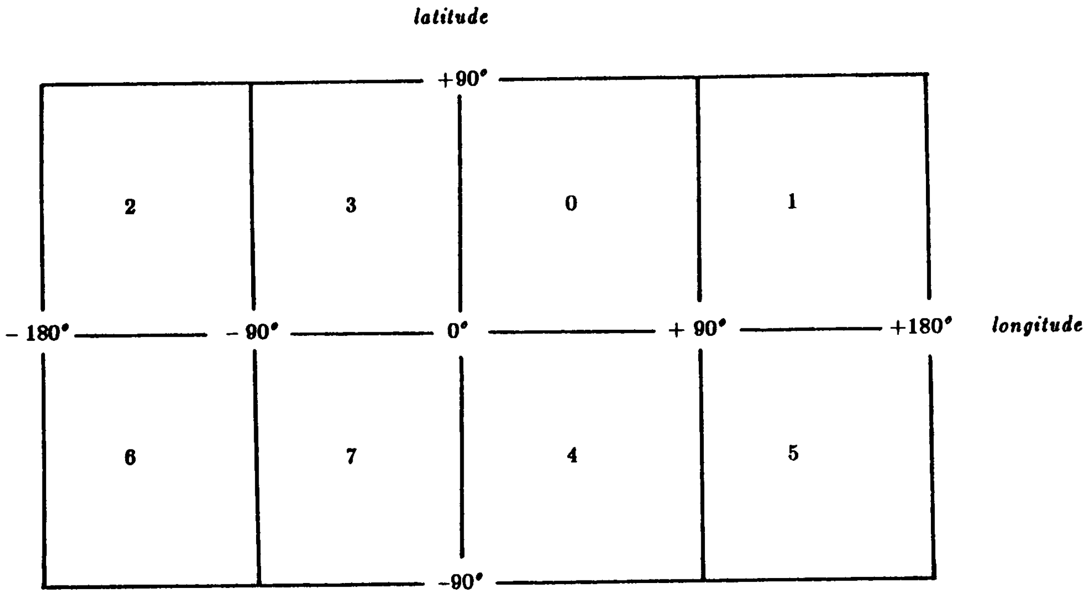


Figure 1. Initial octahedral decomposition of the earth at level 0

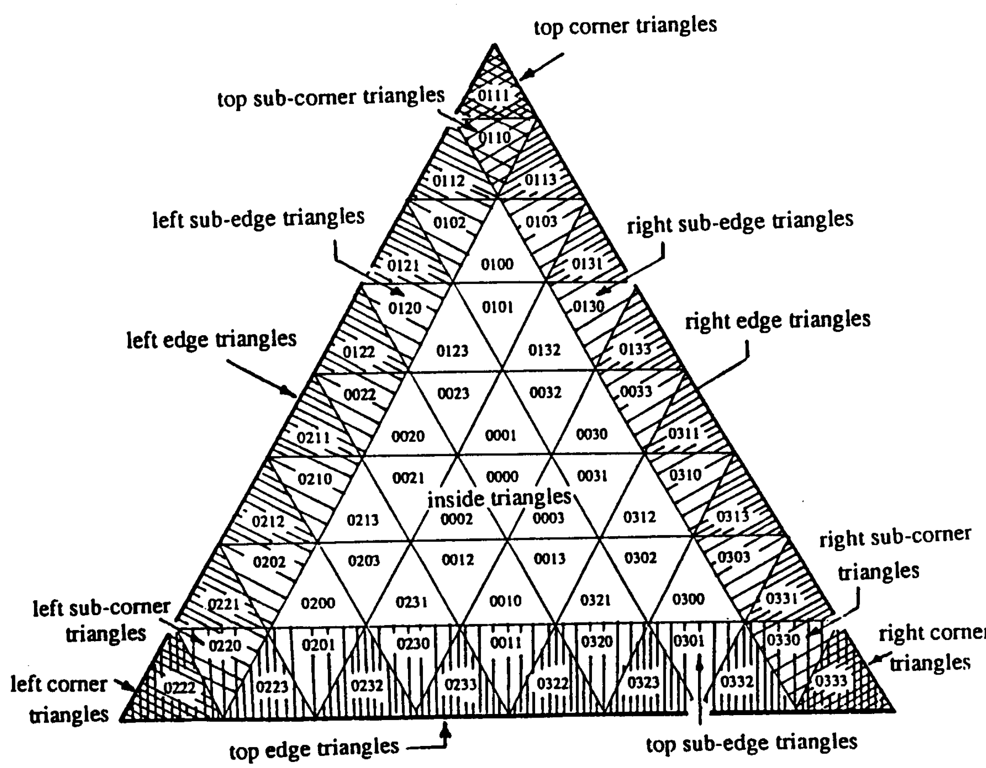
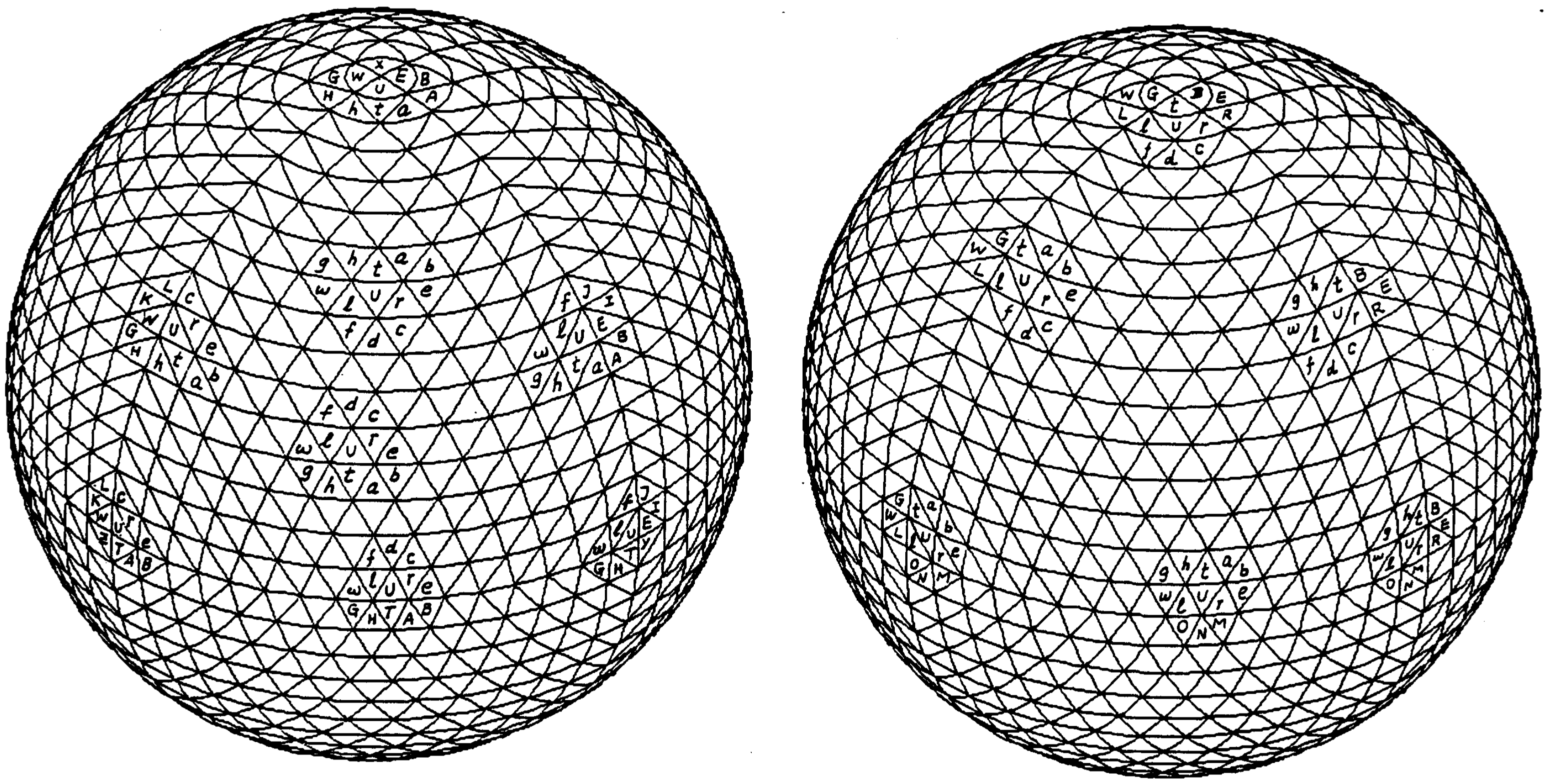


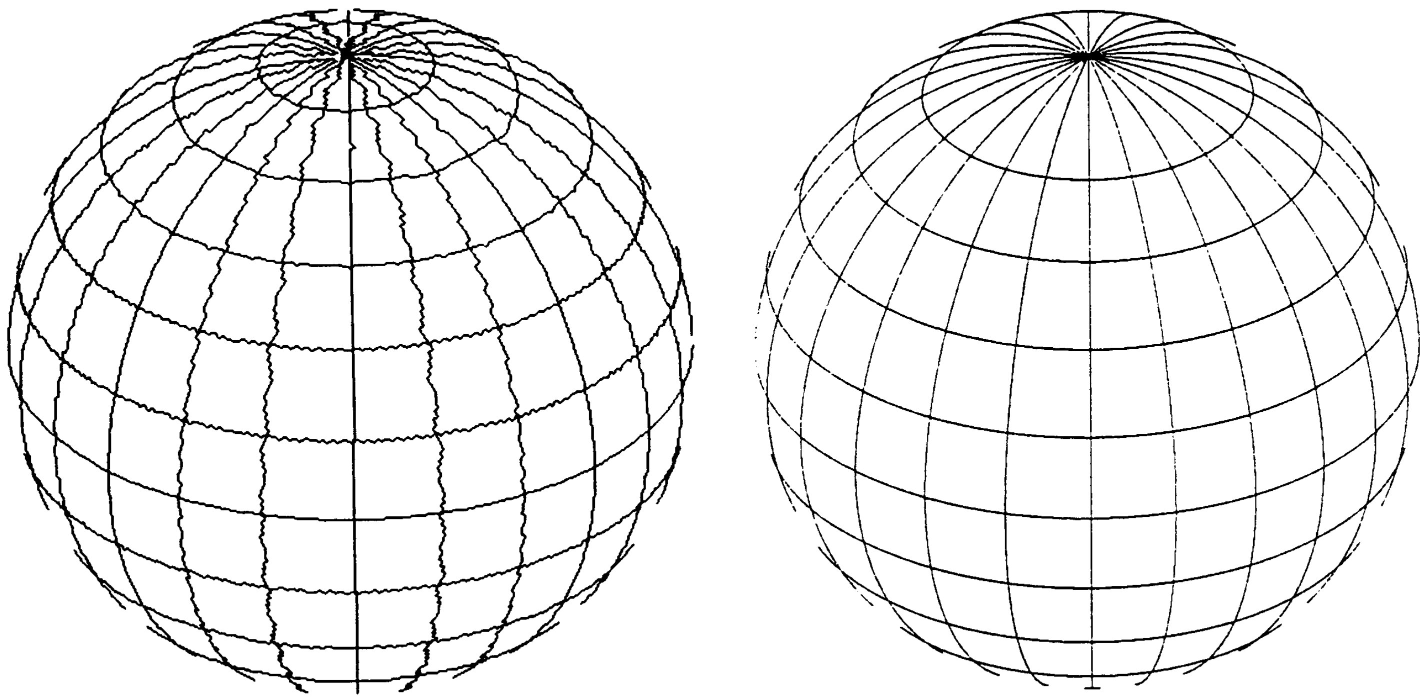
Figure 2. The inside, edge and corner triangles in an octant



(a). The neighbors of inside, edge and corner triangles

(b). The neighbors of sub-edge and sub-corner triangles

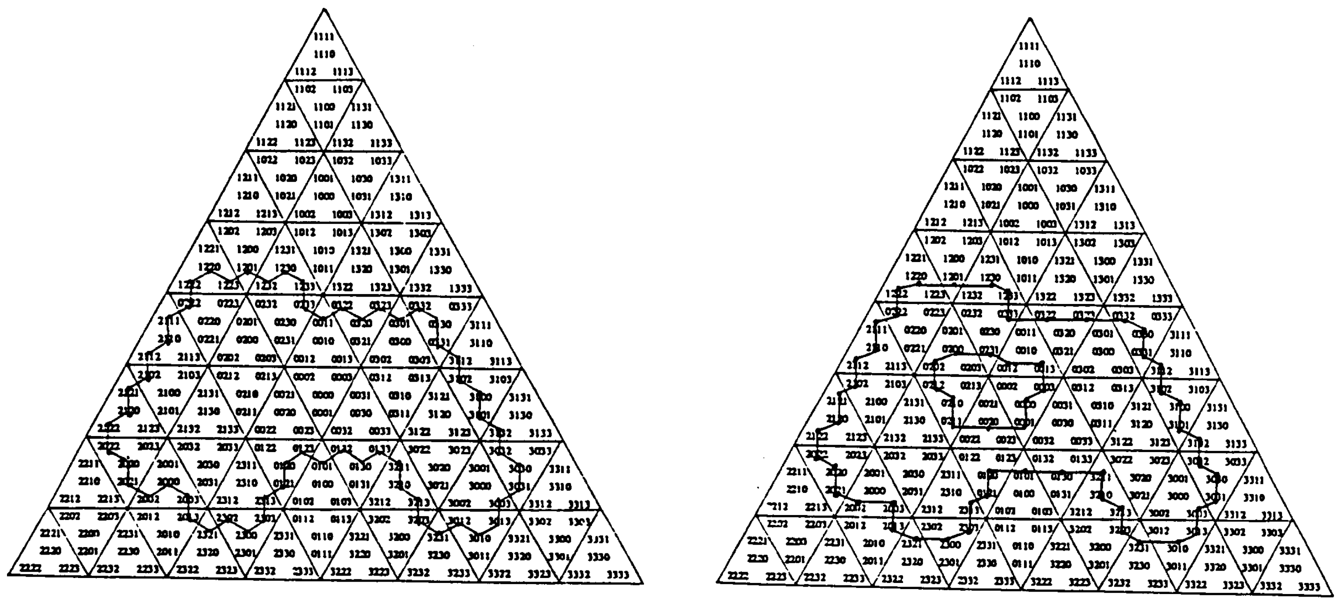
Figure 3. The neighbors of a triangle when it is in different locations



(a). Sixth level

(b). Tenth level

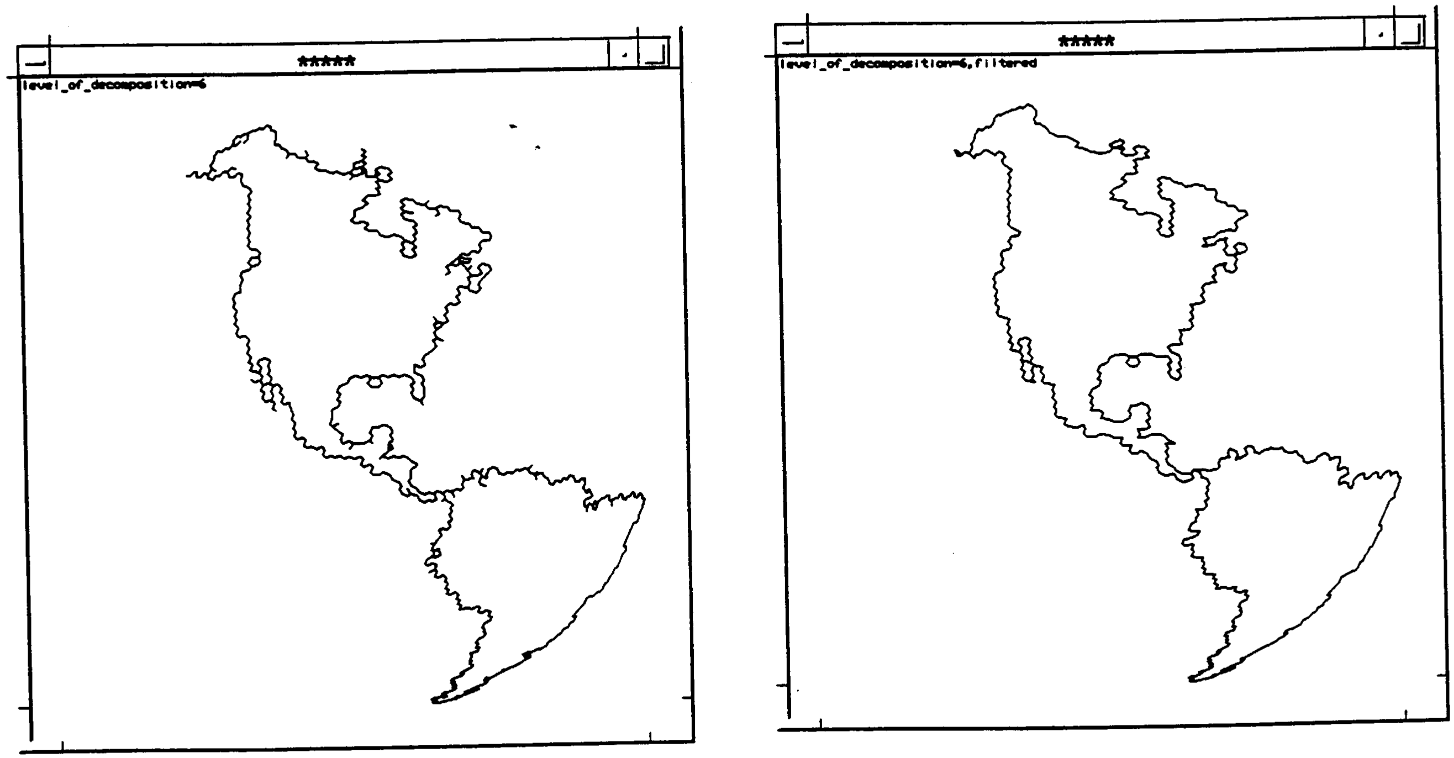
Figure 4. The longitudes and latitudes of the globe are interpolated to sixth and tenth levels



(a). The 3-code chain code representation of a border line starting in triangle 2013 and moving clockwise, the boundary is represented as: *tllltltrtrtrtrtrrrrrrtrrrrrrrtrtrtrtltllltltllllltltllll*

(b). The 8-code chain code representation of border line starting in triangle 2013 and moving clockwise, the boundary is represented as: *twltltrtrtrtreerteeertrtrtrtrtltlwltltwwwtltlwl*

Figure 5. The border line of a region represented by chain codes in THDS



(a). Generalized coastline of North and South America illustrating "zig-zag"

(b). Coastline of North and South America after smoothing

Figure 6. Th coastline of North and South America before and after smoothing

Figure 7. The coastline of North and South America interpolated to level 8, converted to 8-code chain representation and smoothed

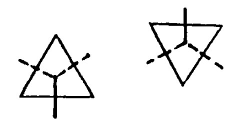
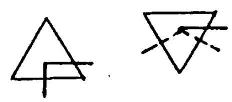
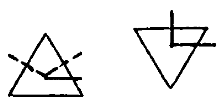
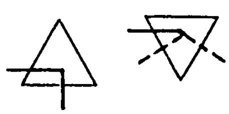
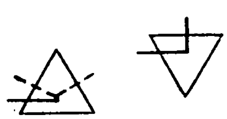
pattern	possible point types	criteria	
		Triangle upward	triangle downward
a 	single intersection point	$t_{i-1} \cap (l_{i+1} \cup r_{i+1}) \cap \overline{e}_{i+1} \cap \overline{w}_{i+1}$	*or* $t_{i+1} \cap (l_{i-1} \cup r_{i-1}) \cap \overline{e}_{i+1} \cap \overline{w}_{i+1}$
b 	upward tangent entry point	$t_{i-1} \cap \overline{l}_{i+1} \cap \overline{r}_{i+1} \cap e_{i+1} \cap \overline{w}_{i+1}$	$\overline{t}_{i-1} \cap (l_{i-1} \cup {}_{i-1}) \cap e_{i+1} \cap \overline{w}_{i+1}$
c 	downward tangent entry point	$t_{i-1} \cap (\overline{l}_{i+1} \cup \overline{r}_{i+1}) \cap \overline{e}_{i+1} \cap w_{i+1}$	$\overline{t}_{i-1} \cap l_{i-1} \cap r_{i+1} \cap \overline{e}_{i+1} \cap w_{i+1}$
d 	downward tangent exit point	$t_{i+1} \cap \overline{l}_{i+1} \cap \overline{r}_{i+1} \cap \overline{e}_{i-1} \cap w_{i+1}$	$\overline{t}_{i-1} \cap (l_{i+1} \cup r_{i+1}) \cap \overline{e}_{i-1} \cap w_{i-1}$
e 	upward tangent exit point	$\overline{t}_{i-1} \cap (l_{i+1} \cup r_{i+1}) \cap \overline{e}_{i-1} \cap w_{i-1}$	$t_{i+1} \cap \overline{l}_{i+1} \cap \overline{r}_{i+1} \cap \overline{e}_{i-1} \cap w_{i-1}$

Figure 8. Flags assigned to boundary triangles in the region filling algorithm. In the notation used to represent the region boundary using 8-code, t_i is true if the i-th entry in the chain is a move to the top neighbor, etc.

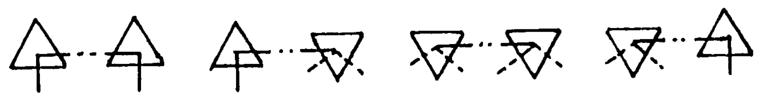
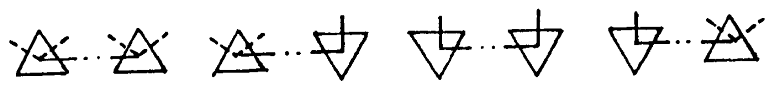
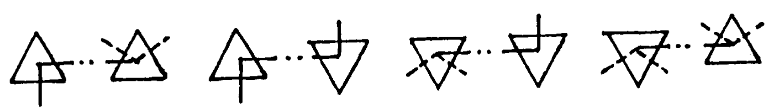
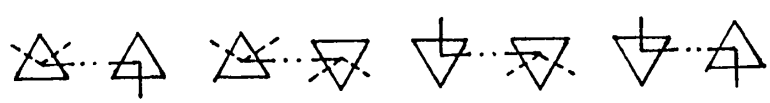
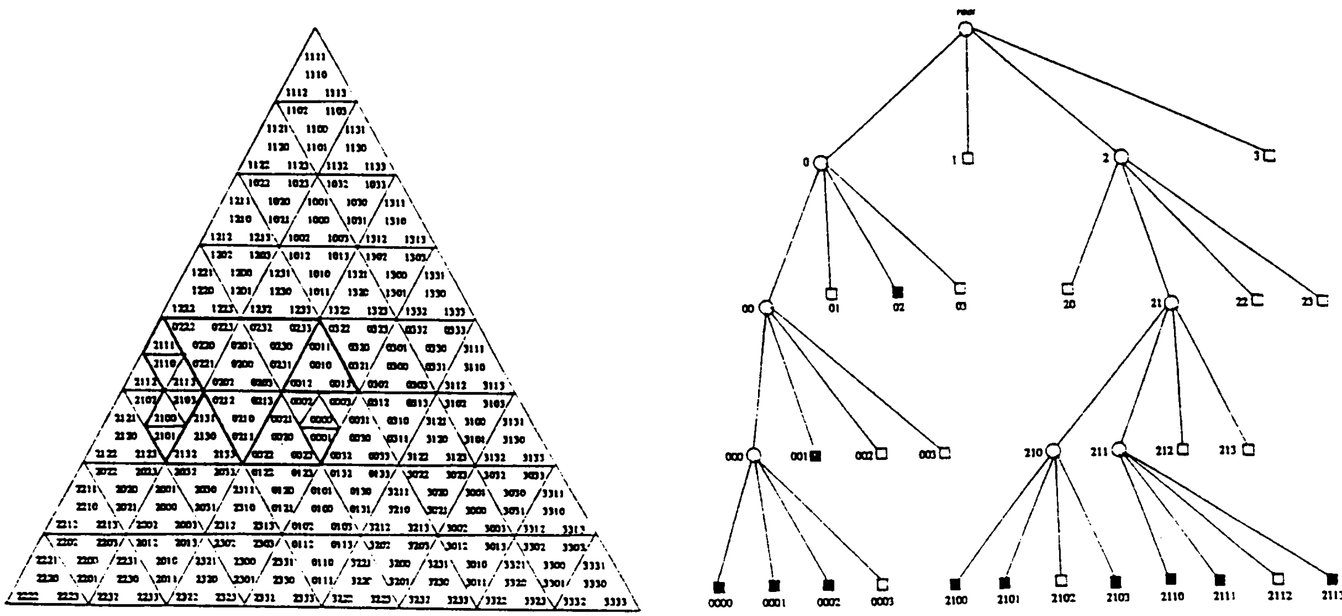
type of points	Patterns of border line traveling through points for region filling
simple points (not an intersection point)	
upward tangential entry downward tangential exit (not an intersection point)	
downward tangential entry upward tangential exit (not an intersection point)	
single point intersection entry or exit (possible intersection point)	
upward tangential entry upward tangential exit (possible intersection point)	
downward tangential entry downward tangential exit (possible intersection point)	

Figure 9. Types of boundary triangles

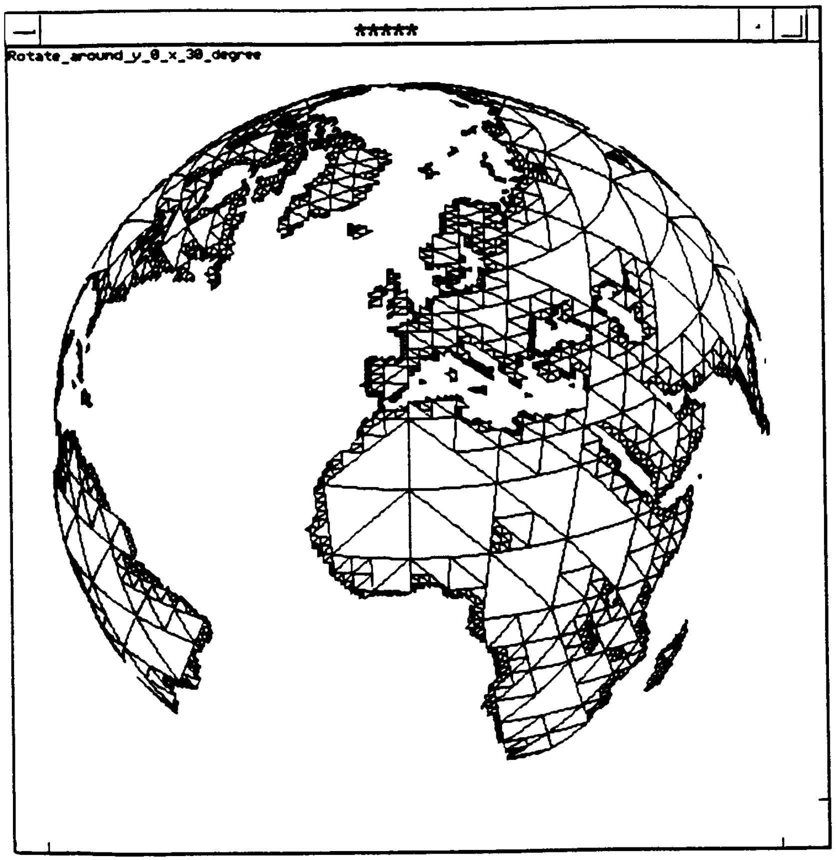
Figure 10. The North and South American Continents shown in Figure 7 filled with triangles at level 8. The lakes in the continents are excluded as holes



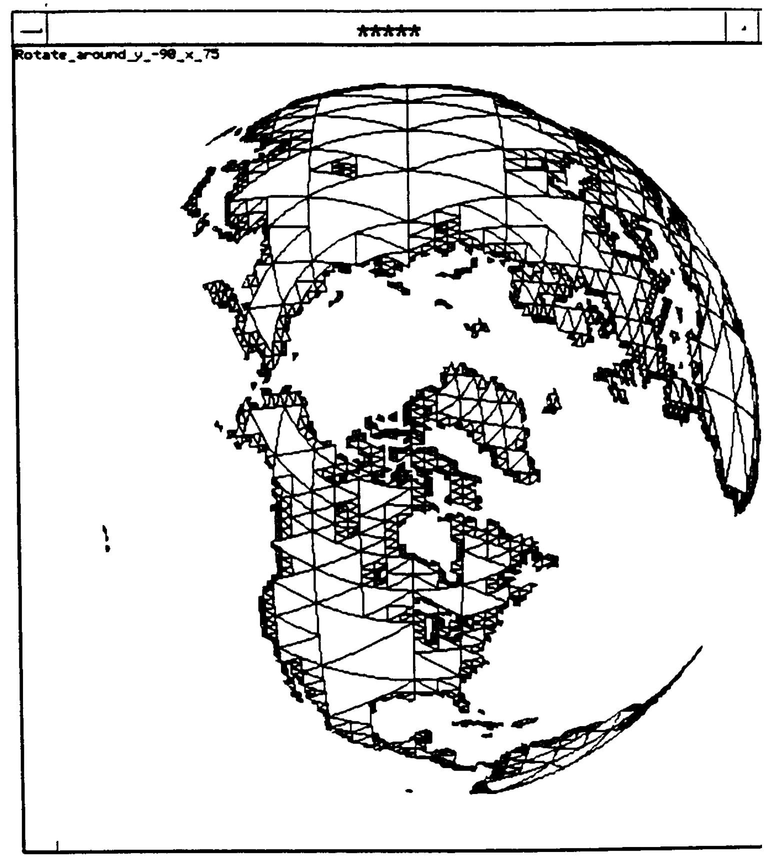
(a). Divide the region to triangles of different levels

(b). The region is represented by a tree with nodes and leaves

Figure 11. Node representation of a region

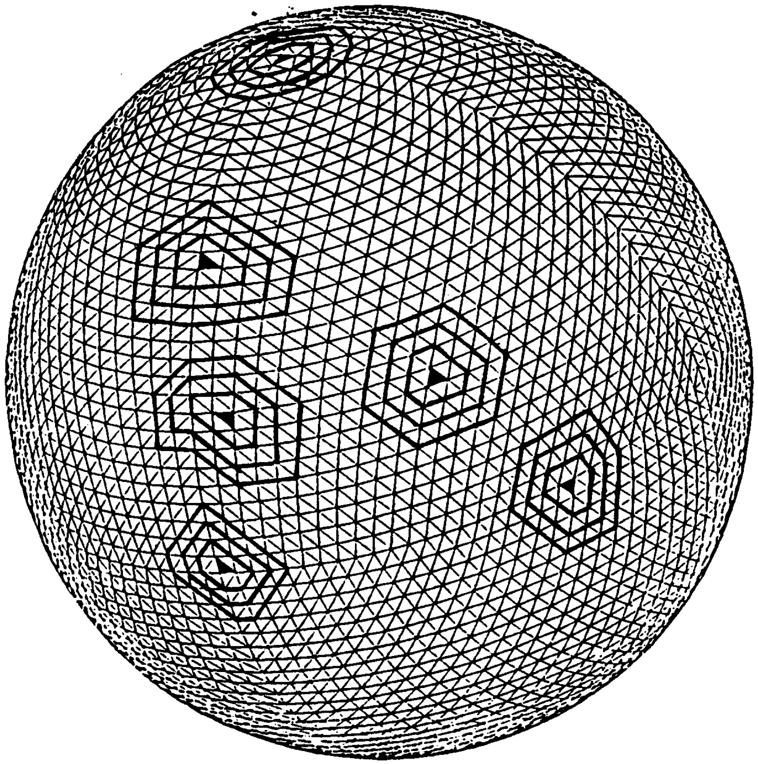


(a). 0 degree rotation around the y axis and 30 degree around the x axis

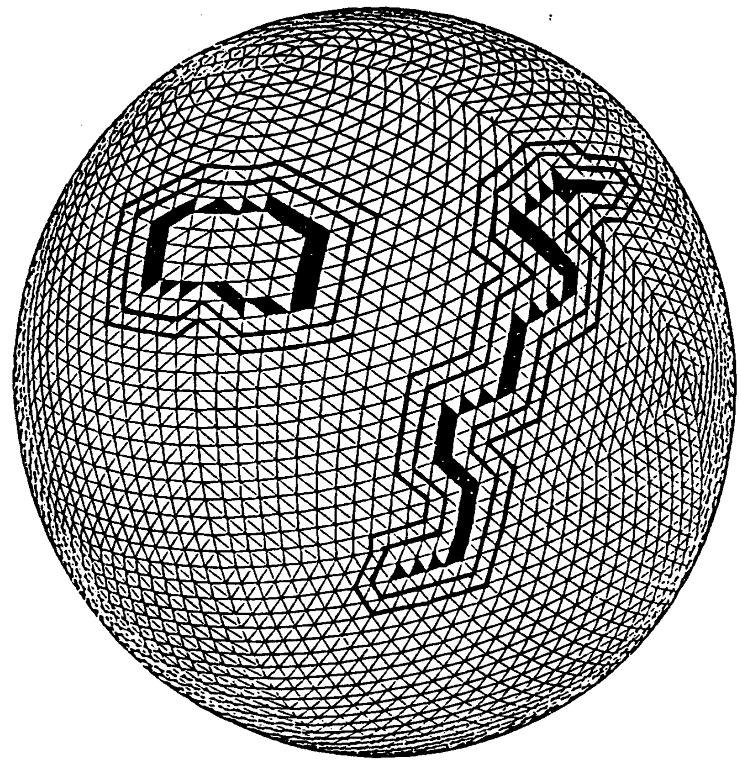


(b). -90 degree rotation around the y axis and 75 degree around the x a:

Figure 12. Quadtree representation of the Earth land surface with different projection angles



(a). Dilation of points



(b). Dilation of line and region

Figure 13. Dilation of point, line and region

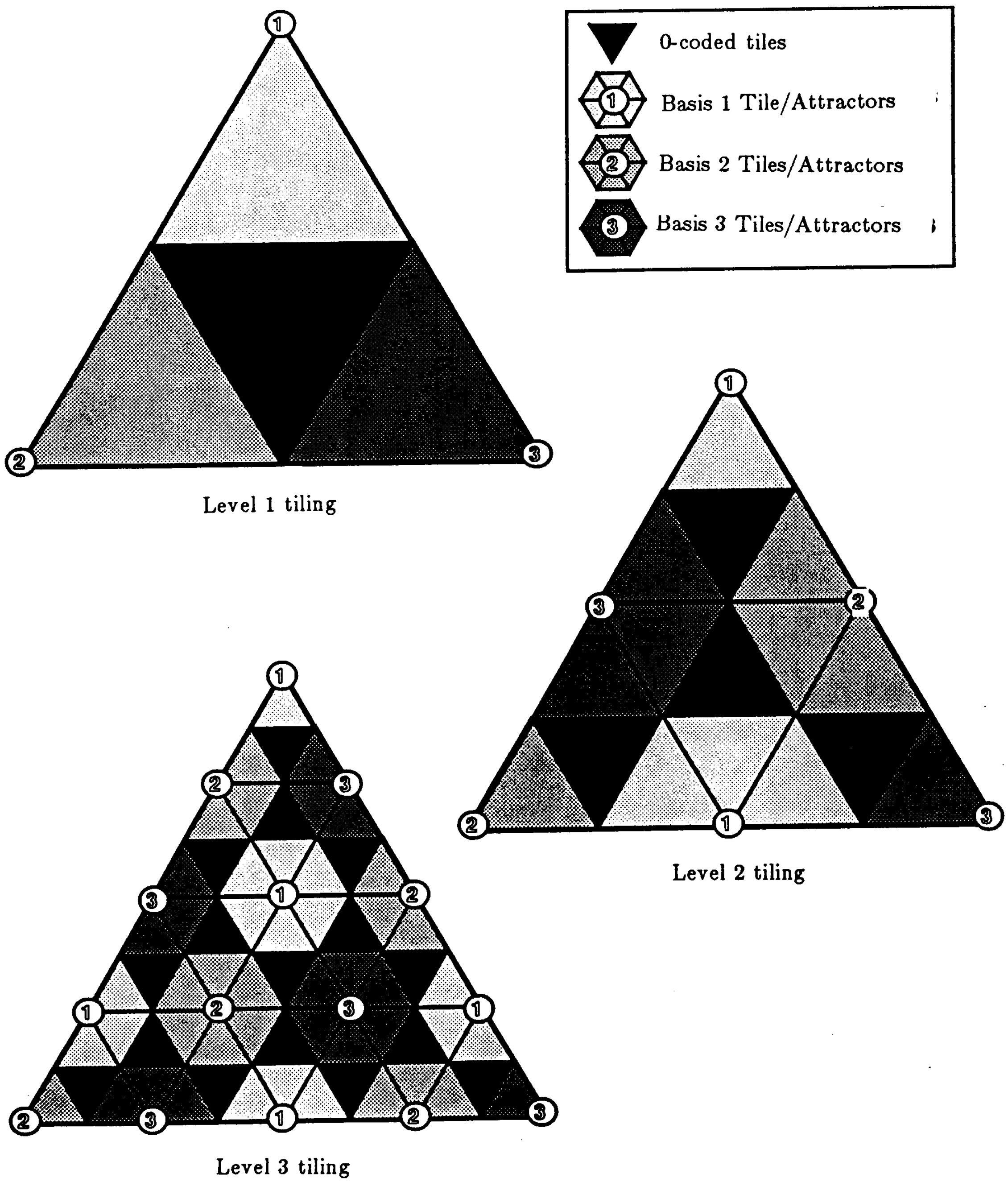


Figure 14. Development of attractors relating triangular grid vertices and tiles

ϕ	λ	ϕ	λ	ϕ	λ	ϕ	λ
61.17	-117.00	60.83	-116.00	60.83	-115.25	61.00	-114.42
60.92	-113.92	61.25	-113.75	61.42	-112.92	61.58	-112.17
62.08	-111.75	62.33	-110.92	62.42	-109.83	62.67	-110.17
62.67	-109.00	62.83	-110.00	62.83	-110.75	62.67	-111.58
62.42	-111.92	62.08	-112.42	62.00	-113.08	62.17	-113.83
62.42	-114.17	62.42	-115.33	62.17	-115.08	61.83	-114.50
61.75	-115.25	61.42	-115.33	61.17	-115.83	61.17	-117.00

(a). Input data, 28 pairs of longitude λ and latitude ϕ

210310103	210310131	210313210	210313213	210313002	210313012
210313010	210313011	210313322	210303322	210303320	210303321
210303302	210303312	210303003	210303013	210303010	210303011
210303233	210303230	210303231	210303203	210303202	210303221
210303220	210303223	210303232	210313232	210313201	210313200
210313202	210313212	210310103			

Initial address $A_0 =$ 210310103
chain code: *reretrtrtrtrtwtllltlwltretllltw*

(b). After smoothing and converting to 8-code. The number of triangular addresses reduces to 33

Table 1. Conversion of input longitude latitude pairs to triangular addresses and smoothed 8-code

Appendix. Areas of Region on Spherical and Ellipsoidal Earths

If the earth was assumed a sphere, the earth surface area A between latitude ϕ_1 and ϕ_2 covered by a level 0 triangle is

$$A = \frac{\pi R^2}{2} (\sin\phi_2 - \sin\phi_1) \tag{A-1}$$

At level n, the total number of triangles in the belt between ϕ and $\phi + \frac{\pi}{2^{n+1}}$ is

$$N_\phi = \frac{2^{n+2}}{\pi} (\frac{\pi}{2} - \phi) - 1$$

and the earth surface area of a triangle at level n is

$$\Delta A_\phi = \frac{A}{N_\phi} = \frac{\pi^2 R^2}{2^{n+3}} \frac{\sin(\phi + 2^{-n-1}\pi) - \sin\phi}{\frac{\pi}{2} - \phi - 2^{-n-2}\pi} \tag{A-2}$$

When $2^{n+1} \gg 1$, we have

$$\Delta A_\phi \approx k \frac{\cos\phi}{\frac{\pi}{2} - \phi} = k \frac{\sin\chi}{\chi} = k \; sinc \; \chi \tag{A-3}$$

where

$$k = \frac{\pi^3 R^2}{2^{2n+4}}, \qquad \chi = \frac{\pi}{2} - \phi.$$

Expression (A-3) implies that at high level of decomposition, the area of a decomposed triangle is a function of latitude ϕ which equals to

$$\phi = \frac{\pi}{2^n \sqrt{3}} y \tag{A-4}$$

When the ellipsoid earth is considered, we use the equation of ellipsoid

$$x = R\cos\phi, \qquad y = R'\sin\phi$$

where R and R' are the radius of the earth in the equator and pole directions respectively. The earth surface area A between ϕ_1 and ϕ_2 is

$$A = 2\pi \int_{\phi_1}^{\phi_2} x\sqrt{x^2 + y^2} \, d\phi \tag{A-5}$$

where

$$x^2 + y^2 = R^2\cos^2\phi + R'^2\sin^2\phi = R^2 - (R^2 - R'^2)\sin^2\phi = R^2 - r^2\sin^2\phi$$

$$x\,d\phi = R\cos\phi\,d\phi = R d\sin\phi$$

where $r = \sqrt{R^2 - R'^2}$, and

$$A = 2\pi R \int_{\sin\phi_1}^{\sin\phi_2} \sqrt{R^2 - r^2\sin^2\phi}\; d\sin\phi = 2\pi R\, r \int_{\sin\phi_1}^{\sin\phi_2} \sqrt{(\frac{R}{r})^2 - \sin^2\phi}\; d\sin\phi$$

$$= \pi R\, r\, [(\sin\phi_2\sqrt{\frac{R^2}{r^2} - \sin^2\phi_2} - \sin\phi_1\sqrt{\frac{R^2}{r^2} - \sin^2\phi_1}) + \frac{R^2}{r^2}(arcsin(\frac{r\sin\phi_2}{R}) - arcsin(\frac{r\sin\phi_1}{R}))]$$

Since we have the relation that

$$\sin\phi_2\sqrt{R^2 - r^2\sin^2\phi_2} - \sin\phi_1\sqrt{R^2 - r^2\sin^2\phi_1}$$

$$= R\,[\sin\phi_2[1 - \frac{1}{2}(\frac{r}{R}\sin\phi_2)^2 - \frac{1}{8}(\frac{r}{R}\sin\phi_2)^4 - \frac{1}{16}(\frac{r}{R}\sin\phi_2)^6 - \ldots -] -$$

$$- \sin\phi_1[1 - \frac{1}{2}(\frac{r}{R}\sin\phi_1)^2 - \frac{1}{8}(\frac{r}{R}\sin\phi_1)^4 - \frac{1}{16}(\frac{r}{R}\sin\phi_1)^6 - . - \ldots]]$$

$$\approx R\,[(\sin\phi_2 - \sin\phi_1) - \frac{r^2}{2R^2}(\sin^3\phi_2 - \sin^3\phi_1)]$$

In this expression we neglect terms of 4 -th and higher orders (they are less than 0.000006 of the first term). With the relation

$$arcsinx = x + \frac{1}{2}\frac{x^3}{3} + \frac{1}{2}\frac{3}{4}\frac{x^5}{5} + \ldots$$

and neglecting terms of 5 -th and higher order, we have

$$\frac{R^2}{r}[arcsin(\frac{r}{R}\sin\phi_2) - arcsin(\frac{r}{R}\sin\phi_1)]$$

$$\approx \frac{R^2}{r}[\frac{r}{R}\sin\phi_2 + \frac{1}{6}(\frac{r}{R}\sin\phi_2)^3 - \frac{r}{R}\sin\phi_1 - \frac{1}{6}(\frac{r}{R}\sin\phi_1)^3]$$

$$= R\,[\sin\phi_2 - \sin\phi_1 + \frac{r^2}{6R^2}(\sin^3\phi_2 - \sin^3\phi_1)]$$

Furthermore let $\phi_2 = \phi_1 + \Delta\phi$ and assume $\Delta\phi = \frac{\pi}{2^{n+1}} \ll 1$, we have

$$\sin^3(\phi_1 + \Delta\phi) - \sin^3(\phi_1) = \frac{3}{4}[\sin(\phi_1 + \Delta\phi) - \sin\phi_1] - \frac{1}{4}[\sin 3(\phi_1 + \Delta\phi) - \sin 3\phi_1]$$

$$= \frac{3\pi}{2^{n+3}}[\cos\phi - \cos 3\phi]$$

Then

$$A = 2\pi R^2[\sin\phi_2 - \sin\phi_1 + \frac{r^2}{6R^2}(\sin^3\phi_2 - \sin^3\phi_1)]$$

$$= k_1[\cos\phi_1 + \frac{r^2}{8R^2}(\cos\phi_1 - \cos 3\phi_1)]$$

where

$$k_1 = \frac{\pi^2 R^2}{2^n}.$$

Then

$$\Delta A = \frac{A}{\frac{2^{n+4}}{\pi}(\frac{\pi}{2} - \phi_1) - 1} = k\frac{\cos\phi_1 + \frac{r^2}{8R^2}(\cos\phi_1 - \cos 3\phi_1)}{\frac{\pi}{2} - \phi_1 - \frac{\pi}{2^{n+2}}}$$

$$\approx k\frac{\cos\phi_1 + \frac{r^2}{8R}(\cos\phi_1 - \cos 3\phi_1)}{\frac{\pi}{2} - \phi_1} = k[\frac{\sin\chi}{\chi} + \frac{r^2}{8R^2}\frac{\sin\chi}{\chi} + \frac{r^2}{8R}\frac{\sin 3\chi}{\chi}]$$

$$= k[sinc\,\chi + \frac{r^2}{8R^2}sinc\,\chi + \frac{r^2}{8R^2}\frac{\sin 3\chi}{\chi}] \quad (A\text{–}6)$$

where

$$k = \frac{\pi^3 R^2}{2^{2n+4}}.$$

For $\phi_1 = 0$, we have $\chi = \frac{\pi}{2}$, $sinc\,\chi = \frac{2}{\pi}$, $\frac{\sin 3\chi}{\chi} = \frac{-2}{\pi}$, and

$$\Delta A_0 = k[\frac{2}{\pi} + \frac{r^2}{8R^2}\frac{2}{\pi} - \frac{r^2}{8R^2}\frac{2}{\pi}] = \frac{2k}{\pi}$$

For $\phi_1 = \frac{\pi}{2,}$ we have $\chi = 0$, $sinc\,\chi = 1$, $\frac{\sin 3\chi}{\chi} \approx 3$, we have

$$\Delta A_{\frac{\pi}{2}} = k[1 + \frac{r^2}{2R^2}]$$

The ratio of triangle area at $\phi = \frac{\pi}{2}$ to the area at $\phi = 0$ is

$$\frac{\Delta A_{\frac{\pi}{2}}}{\Delta A_0} = \frac{\pi}{2} [1 + \frac{r^2}{2R^2}]$$

As a comparison, the ratio is $\frac{\pi}{2}$ on a spherical earth. The triangle area at the poles of an ellipsoidal earth is about 0.313 percent larger than that calculated for a spherical earth.