INDEXING ON SPHERICAL SURFACES USING SEMI-QUADCODES

Ekow J. Otoo and Hongwen Zhu

TR-217, DECEMBER 1992

School of Computer Science, Carleton University
Ottawa, Canada, KIS 5B6

Indexing on Spherical Surfaces Using Semi-Quadcodes

Ekow J. Otoo and Hongwen Zhu
School of Computer Science
Carleton University
Ottawa, Ontario, Canada, K1S 5B6

November 20, 1992

Abstract

The conventional method of referencing a point on a spherical surface of known radius is by specifying the angular position of ϕ and λ with respect to an origin at the centre. This is akin to the $\langle x, y \rangle$ coordinates system in R^2 cartesian plane. To specify a region in the cartesian plane, two points corresponding to the diagonal points $\langle x_1, y_1 \rangle$ and $\langle x_2, y_2 \rangle$ are sufficient to characterize the region. Given any bounded region, of $2^h \times 2^h$ an alternate form of referencing a square subregion is by the linear quadtree address [9] or quadcode [12]. Corresponding encoding scheme for spherical surfaces is lacking. Recently a method similar to the quadtree recursive decomposition method has been proposed independently by Dutton and Fekete. Namely, the *quaternary triangular mesh (QTM)* [4] and the *spherical quadtree (SDT)* [7]. The addressing method of the triangular regions suggested are very similar. We present a new labeling method for the triangular patches on the sphere that allows for a better and more efficient operation and indexing on spherical surfaces.

1 Introduction

Advances in computer graphics technology now provide capability to model 3-dimensional objects with added manipulative functions such as real-time rotations, etc. In certain disciplines such as in the earth and space science, it is becoming increasing desirable to model spherical surfaces. In geography, the idea of map projections, is effectively a form of modeling the spatial features on spherical surfaces as a flat or planar surfaces, e.g., paper maps, to the extent that actual measurement on the map can be translated, by scaling to the actual physical distances on the ground. The concept of paper maps is an abstract model of the physical world. Given the current state of computer and graphics technology one should be able to visualize information depicted in a map as an actual 3-dimensional model of the physical world. To achieve this, we need a convenient way of representing, storing, retrieving and accessing information relating to spherical regions of the globe.

Techniques of visualizing data of planar regions have benefited considerable from the use of hierarchical data structuring methods such as the quadtrees [19, 18]. To put the problem in perspective, consider a square planar region (A, B, C, D) of figure 1.1. A location in this region may be defined, to some degree of precision, by the Cartesian coordinate system with respect to an origin, say in the lower left hand corner. To specify a region in this system, for example, region (a, b, c, d) of figure 1.1, the coordinates of two diagonal points $a = \langle x_1, y_1 \rangle$ and $c = \langle x_2, y_2 \rangle$, are specified. Alternatively, using a quadtree decomposition, of the space, the subregion (a, b, c, d) can be defined by a linear quadtree code [9] or quadcode [12]. In general, an arbitrary subregion will be defined by a list of quadcodes.

The quadtree like decomposition of planar regions into rectangular cells, and also volumetric space into voxels or cuboids, have had extensive applications in graphics, spatial searching and solid modelling. The address of a cell may be specified by the string of quaternary digits from the root to the terminal node representing the subregion. Such an address embeds the spatial coordinates as well. It is natural then to explore the extension of quadtree-like decomposition method to spherical surfaces. In this respect, we note some properties of the region quadtree decomposition.

- It preserves the rectilinear shapes of the cells at any level.
- The cells, at any level, have equal parametric measures: equal area, perimeter, volume, etc.

The natural extension of the concept of quadtrees to decomposing spherical surfaces so that regions can be appropriately indexed, has been addressed in [2, 20, 14, 15]. Unfortunately this has been found to be inappropriate. Its inappropriateness is exacerbated by the fact that the two desired properties, mentioned above, are not preserved. The shapes of the cells are rectilinear for regions around the equator but triangular near the poles, see figure 1.2. The cells at the same level of decomposition neither have equal shapes or equal areas. The question then is whether there is some other hierarchical decomposition method for spherical surfaces that lends itself to the operations much like the quadtree is to planar surfaces. Such a decomposition and representation can be used for the construction of 3-D model of the globe and for the rendering of both volume and spherical surface data so that distribution over the surfaces can be displayed in vector form or in discrete element of forms as rendered triangles [11].

The answer to this question is in the affirmative and it has been addressed by a number of researchers. Variants of the same idea have been proposed and associated with different names. Dutton [4, 6, 5] proposed a triangular decomposition method which he refers to as *the Quaternary Triangular Mesh (QTM)*. Each triangular cell, in his method, is called *a triacon*. He proposed this as an alternative method for geographically referencing locations on planetary surfaces. He showed that the method has considerable advantages over other conventional methods such as the geographic coordinate system, of longitude and latitude. We briefly elaborate on his approach in the next section. The method has since been studied by Goodchild and Shiren [11] and they conclude that the method serves as a better alternative to the quadtree for global indexing in Geographic Information Systems. A similar approach has been proposed by Fekete [8, 7]. He referred to his approach as *Spherical Quadtree (SQT)* and calls the triangular cells *trixels*.

In either of the two methods, the spherical surface is approximated by a set of triangular patches. Each triangular patch is recursively tessellated by bisecting the edges to form a median point. These median points are then projected to the surface. By joining the new median points, four new triangular patches are generated. One interesting characteristic of the scheme is that the higher the degree of tessellation, the better the triangular patches approximate the shape of the sphere. The two schemes, however, defer with respect to: i) the initial number of triangular patches of the base platonic solid and

ii) the labeling schemes used for addressing the triangular cells.

We observe that the QTM method is to region encoding of spherical surfaces as the linear quadtree [9] is to region encoding of planar surfaces. In proposing the use of QTM, Dutton recommends its use for point location. This requires an arbitrary long string of QTM digits when the depth of tessellation is high. A number of open questions were raised. In particular he for how best to implement the variable resolution aspects of QTM codes on a hardware that is designed to process fixed-precision data. We believe the significance of the QTM encoding is highly diminished when stretched in this manner for point location instead of it being used simply to address regions on spherical surfaces. Given the success and dominant use of the region quadtree and related quadtree-like structures [9, 10, 19, 18] in image processing and spatial indexing, the QTM encoding holds great promise in its use for indexing on spherical surfaces.

In proposing a similar idea called the Spherical Quad-trees (SQT), Fekete [7] presents arguments for its importance in a number of applications. Namely:

- generalization of cartographic visualization;
- efficient access to spherically distributed data;
- development of graphical browser to select metadata of regions.

In this paper, we address some of the problems related to the use of the QTM and SQT codes for spatial indexing and we show how it may be used effectively for indexing regions on spherical surfaces We propose an alternate labeling scheme which we refer to as the *semi-quadcode (SQC)*. Our SQC (not to be confused with the abbreviation (SQT)) is actually a variant of the QTM scheme of Dutton. We would like to emphasize that all three schemes are similar except for the initial base structures inscribed in the sphere. They differ also in the manner in which they generate the address labels and ultimately the algorithms for basic operations such as neighbour finding, adjacency detection, connected component labeling, etc.

We utilize the scheme more for addressing triangular regions only up to the level of precision tolerable by the integer representation of the hardware. The highlights and major contribution of this paper are in:

1. the development of a new labeling scheme termed semi-quadcodes (SQC) in place of QTM methods;

2. the design of simpler algorithms for operations on Spherical surfaces: neighbour finding, union, intersection, difference, rotation, etc.;

3. development of new transformation algorithm from angular coordinates to semi-quadcode addresses;

4. derivation algorithm of the angular coordinates of the three points that define the triangular cell of a given SQC address;

5. application of the new method for indexing QTM cells on spherical surfaces;

We discus the background to the development of the SQC scheme in the next section and show how it relates to the earlier QTM and SQT methods. In section three, we present some properties of the method and its related arithmetic operations. We compare the scheme with the earlier proposed schemes in section four and suggest other applications of it. We conclude in section five, giving an outline of future direction of its use in modeling the globe in Geographic Information Systems.

2 Background

The basic idea of the Quaternary Triangular Mesh (QTM), Spherical Quadtree and Semi-Quadcode is to approximate the surface of a sphere by a set of triangular patches. The method is similar to the digitization of a curve whereby the curve is approximated by a set of line segments. Consider the arc segment, ABC, of figure 2.1a. This may be approximated by the chord AC. Suppose we recursively perform the following procedure. At each level of the recursive process, the midpoint B, of each line segment, say (AC) of the preceding level, is projected radially to intersect the arc at B'. The new approximation to the arc is now $(AB'C)$. Continuing the process to a sufficiently high level of segmentation, the circular arc AC can be approximated by a series of equal line segments as shown in figure 2.1b. This basic idea is carried over to the spherical surface except that a group of three points, forming a triangular patch, is used.

The development of the Quaternary Triangular Mesh (or QTM code) begins by conceiving one of the platonic solids, i.e., the octahedron as inscribed in a sphere such that the vertices touch the surface. The octahedron is chosen because of the property that it can be orientated so that the geographic coordinates (ϕ, λ), of a point on the surface of the earth, consistently and unambiguously, map to a triangular region of a QTM code Q, where $Q = \mathcal{F}(\phi, \lambda)$ and $\mathcal{F}$ is the mapping function. Figures 2.2a and 2.2b show the first two levels of tessellations in such a scheme. We will call the platonic solid inscribed in the sphere (in our case the octahedron), the *base solid* of the QTM scheme and refer to one of the eight triangles of the base solid as *a base triangular facet*. The eight base triangular facets are addressed as 0, 1, ..., 7. In the sequel we will focus on addressing on the spherical surface defined by the representative facet 0.

Like the quadtree, one can construct a hierarchical tessellation of the triangular facet into smaller and smaller regular cells starting from the base triangular fact as shown in figure 2.3a. The QTM region labelling scheme proposed by Dutton is carried out is as follows. Given a triangular facet that is tessellated regularly into four sub-triangular regions, the middle triangle is labelled 0, the one to the left is labelled 1, the one to the right is labelled 2 and the upper triangular cell is labelled 3. Although Dutton proposed the method for point location, we believe the significance of the method is highlighted by its use for addressing regions on spherical surfaces. Consequently, the hierarchical decomposition process terminates when a small enough cell is generated. A triangular cell becomes small enough when either the error introduced by approximating it simply as a flat surface is negligible or the longest code strings occupies the full integer word of the hardware.

Goodchild and Shiren [11] simplify Dutton's cell labelling approach to facilitate transformation from latitude and longitude to the quaternary code of the cell that a point lies and vice versa. In the use of the quaternary codes as a global tessellation method, Goodchild and Shiren consider the projection of a sphere onto the eight distinct planer surfaces of the octahedron. These planer surfaces are then recursively tessellated and labelled. In their subsequent decomposition process, the midpoints of the edges of the triangular cells are not projected to intersect the spherical surfaces for the next level tessellation. One major drawback of this approach is that the error introduced as a result of the first level projection onto one of the eight base planes, remain the same irrespective of the subsequent levels of further tessellations.

The Spherical Quadtree of Fekete [7] begins with an inscribed platonic solid of an icosahedron in the sphere. The sphere is therefor approximated initially by the 20 base planar facets of the icosahedron. The approximation is improved by successively subdividing the planer surfaces into four triangular cells. The new triangular cells for the next level are formed by bisecting the edges of the current triangular cells and projecting the new vertices out onto the surface of the sphere. This process is repeated until any desired resolution is attained. Like the quaternary code, the approximation to the spherical surface is represented as a forest of 20 spherical quadtrees, one for each base planer region of the icosahedron. Each cell within each forest can be labeled by a unique code defined by the path from the root to the leaf cell.

Unlike the method of Goodchild and Shiren, the cell labelling is different from that proposed by Dutton. The SQT, uses the symbols {1, 2, 3, 4} for labelling. Let (A, B, C) be the vertex label of a triangle and let (A', B', C') be the set of midpoints of the sides opposite the vertices A, B and C respectively. The new generated triangular cells are labeled as illustrated in the figure 2.3b.

The two methods described briefly, share a number of common properties. We distinguish between the two methods of labelling by referring to them simply as QTM-code and SQT-code. In the sequel, we will refer simply to the QTM-codes with the understanding that in some cases the discussion applies to both methods. A number of useful operations and functions are necessary for the use of QTM in spatial indexing on spherical surfaces. These include the following:

- given the angular measure of ϕ and λ, compute the QTM code, $Q = \mathcal{F}(\phi, \lambda)$, that contains the point;
- given a QTM address of a triangular cell compute the edge or node adjacent cell in a given direction.
- given the resolution and QTM-code of a cell, compute the angular coordinates of the vertices of the triangular cell.
- given an arbitrary figure on the surface of a sphere of known radius, the level h, of the tessellation, and the set of QTM codes of a connected region, compute the approximate area or perimeter of the defined region.
- *Connected Component Labeling:* assign a unique label to each maximal connected region of cells that share a common property.
- given two lists S_1 and S_2, of QTM codes compute the union S = $S_1 \cup S_2$.
- given two lists S_1 and S_2, of QTM codes compute the intersection S = $S_1 \cap S_2$.
- given set of QTM code representing a region at level h, enhance the resolution of the image by generating the representation at level h+1.

Earlier papers have addressed some but not all of the above operations. We propose a new method, *the semi-quadcode*, as an alternative to the QTM codes that gives a different perspective of the QTM scheme. Due to space limitations this paper focuses on the data structure, the representation of semi-quadcodes and some relevant operations. We present a detailed algorithms for the use of the semi-quadcode in [17]. Some concerns were raised about the size of the index generated when the level of tessellations become high. Specifically, one may ask whether an initial index, after h levels of tessellation, could be allocated on secondary storage and then subsequently extended to h+1 levels of tessellation without reorganizing the previous assignments made.

This is equivalent to the resolution enhancement problem mentioned above and is easily realized by organizing the QTM-codes into a B-tree structure [1, 3]. We utilize a similar idea in constructing the index for spherical surfaces except that semi-quadcodes are used.

The use of the semi-quadcodes greatly facilitates the operations mentioned above. We also address the problem of the use of semi-quadcodes for *Global Spatial Indexing.* In using SQC for global spatial indexing, the triangular cells are conceived as corresponding to data pages or buckets that hold detailed description of vector or raster data of the spatial objects that fall within the cells. The SQC's are used as keys to construct a B^+-tree index whose leaves hold pointers to the locations where detailed information relating to the cells (e.g., the vector map data of the cells) are kept.

The technique achieves $O(log_m N)$ page accesses to retrieve information the features of the spherical surface contained in any one of the N triangular cells. This access complexity is independent of the resolution k, $1 \leq k \leq h$, where h is the maximum resolution of the quadcodes. In this manner both horizontal and vertical navigation through the data is achieved with the same access cost irrespective of the size of the region. This paper highlights the significance of the SQC as a variation of the QTM encoding scheme. We point out its relevance to other applications such as visualization of data on the surface of the globe, rendering of maps as an aid in browsing the metadata of global databases in the geosciences.

3 Geometry of Semi-Quadcode

The semi-quadcode (SQC) is a labeling method for spherical regions of arbitrary resolution based on the subdivision of a base octahedron inscribed in the sphere. The structure formed approximates the sphere by subdividing the 8 triangular planar faces as shown in the figure 2.2b. It is perceived as a hierarchical structure with the planar regions of the inscribed base octahedron forming level 0 of the hierarchy. The hierarchical approximation is formed by recursively tessellating the surface at level h-1 to form the new structure at level h. Each of the edges of the triangular facets at level h-1 are bisected and the midpoints pushed radially to the surface. The new triangular facet formed by joining the midpoints at level h effectively partitions the facet at level h-1 into four new ones. As in [6], the 8 base octants are labelled by $\{0, 1, 2, ..., 7\}$. In the sequel we concentrate on the subdivision of facet 0 since the discussion easily caries over to the other base facets by translation, inversion or both.

3.1 Coordinate System of SQC

The semi-quadcode adopts the main concepts of *Quadcodes* [12, 13] to address QTM cells. Quadcode is a linear quadtree encoding scheme with no *Don't Care* digits. To contrast the SQC with QTM, consider first the QTM labelling of the 0 triangular facet that has been tessellated up to level 2 as shown in figure 2.3a. Some triangular cells are upright while others are inverted. Given a QTM code of a cell computing the adjacency of a cell in a specific direction, requires setting up state transition tables [8, 11]. The algorithm for adjacent cell detection and similar others can be simplified considerably by the use of the semi-quadcode labelling.

The QTM encoding scheme has a number of interesting properties described by Dutton. He observed that:-

- As the level of the tessellation increases, facets grow smaller, the QTM codes grow longer and they tend to be more unique thus allowing for unique address assignment to locations.
- A QTM address at level 16 provide the same order of resolution as LANDSAT pixels.
- By interpreting the QTM location codes as integers, and mapping this onto linear storage, they exhibit the property that numerically similar codes tend to lie in close spatial proximity to one another.
- Facets at the same level having QTM codes terminated by the digits 1, 2 or 3 form hexagonal groups of six triangles with triangles terminated by 1 gravitating towards a vertex labelled 1. Those terminated by 2 gravitate towards a vertex labelled 2 and those terminated by 3 gravitate towards a vertex labelled 3. These nodal points were termed *attractors*.

- Suppose an edge (x, y) at level h is bisected to generate a midpoint z, the value of z is given by the expression $z = 6 - (x + y)$.

Suppose we perceive the QTM of one base plane as a graph G = (V, E) where V is the set of nodes and G is the set of edges. Then we may also add to the above the following property.

Proposition 3.1 *The mesh of a base planer surface constructed by the recursive partitioning in of the quaternary triangular mesh decomposition to any level h, is a graph that always has an Eulerian tour.*

This follows from the fact that the base triangular cell has three nodes each of even degree 2. Subsequent tessellations, to any level, introduce new nodes that are always of even degree.

Consider a 2-dimensional square space oriented so that the bottom and left edges are parallel to the X- and Y-axes. Let the bottom triangular be the half-space formed by the diagonal joining the top-left corner to the bottom-right corner of square space. Let this half-space be recursively partitioned by the lines that constitute the quadtree partitioning of the full square region except that each square region so formed is split into two triangular regions by a line joining the top-left corner to the bottom right corner of each quad cell. This half-space partitioning will be termed the *semi-quadtree.* Figure 3.1a illustrates the tessellation of half-space up to level 2. Let $G' = (V', E')$, denote the graph formed by the set of nodes V' and edges E' of the mesh shown in figure 3.1b.

One key observation of the property of the QTM method that led to the development of the Semi-Quadcode (SQC) is that the graph G, of the Quaternary Triangular Mesh at level h, is isomorphic to the graph G' generated by the quadtree-like tessellation of square grid restricted only to the half-space, complete with the diagonal lines as shown in figure 3.1b. The isomorphism established between the QTM graph G and the semi-quadtree graph G' suggests that a number of the quadtree properties should hold naturally for the semi-quadtree and consequently for the QTM.

3.2 SQC Addressing

To appreciate the significance of the SQC scheme, we show how the linear quadtree labeling [9, 10] is adopted. First, observe that the semi-quadtree tessellation is exactly quadtree tessellation of the space restricted to only one half of it. At each level of recursion, a full quad cell either lies entirely in the subspace below the diagonal line or lies along the diagonal line, i.e., it is a diagonal cell. If it lies entirely below the diagonal line it contains two triangular cells corresponding to the upright and inverted QTM cells respectively. Otherwise it contains only one triangular cell corresponding to an upright QTM cell. We will refer to a "triangular cell" as *tricell.*

Each normal rectangular quad cell contains two triangular cells of the QTM scheme. These are designated as the *u-tricell* for the upright triangular cell, and the *i-tricell* for the inverted triangular cell. Hence we can apply the addressing scheme of the linear quadtree to the quad cells and then distinguish between the u-tricell and i-tricell within the same quad cell by appending the digit 1 and 2 respectively to the corresponding linear quadtree code. The above discussion effectively describes a semi-linear quadtree scheme or what we call *semi-quadcode.*

Let the spherical surface defined by one of the triangular planes of the inscribed octahedron be mapped onto one half space of a square space with the bottom and left edges defined as the X- and Y-axis. A quadtree-like tessellation to level h will partition the space into $2^h \times 2^h$ quad cells. Each quad cell can be addressed by the pair of indices X and Y, i.e., $\langle x, y \rangle$ or by an address formed from the linear quadtree addressing scheme. From now on we will refer to the quadcode addresses instead of linear quadtree labelling. For example in figure 3.1b, where h = 2, the space is tessellated into an 4×4 quad cells. The subspace equivalent to the QTM scheme of figure 2.3a is that triangular region 0AB. The quad cell (S,T,U,V) has the indices $\langle 1, 2, \rangle$. The corresponding quadcode address is $(03)_4$. The u-tricell and i-tricell of the quadcode $(03)_4$ are $(031)_4$ and $(032)_4$ respectively.

Recall that the quadcode label of a node in a quadtree is a string of quaternary digits that defines the path, from the root to the node. More importantly, the coordinate indices

$\langle x, y \rangle$ and the quadcode address are related by the Proposition 3.2 stated below.

Proposition 3.2 *Given the X- Y- coordinates $\langle x, y \rangle$ of a quad cell in a quadtree tessellated square region to level h, denote by $(\beta^x_{h-1}\beta^x_{h-2}\cdots,\beta^x_0)$ and $(\beta^y_{h-1}\beta^y_{h-2}\cdots,\beta^y_0)$ the h binary digit representation of x and y respectively. Then the string of quaternary digits formed by taking pairs of bits of the interlaced binary digits of the form $(\beta^x_{h-1}\beta^y_{h-1}\beta^x_{h-2}\beta^y_{h-2}\cdots\beta^x_0\beta^y_0)$ constitutes the quadcode address of the quad cell.*

Denote the mapping of the $\langle x, y \rangle$ coordinate address to the quadcode Q, by a function $\mathcal{F}$ and conversely denote the inverse mapping by $\mathcal{F}^{-1}$. We have then that $Q = \mathcal{F}(\langle x, y \rangle)$, and $\langle x, y \rangle = \mathcal{F}^{-1}(Q)$. Both the function $\mathcal{F}$ and its inverse $\mathcal{F}^{-1}$ are computed by h bit shifts where h is the length of the quadcode or the highest resolution $(2^h \times 2^h)$ of the tessellated space.

For a black and white image representation, the quadcode encodes only the black regions of the space. Hence in practice, the quadcodes represent the black regions of a black and white image. The quadcode representation is a sorted sequence of the quadcode labels. If four connected square block of the form $\{\ q_{h-1}q_{h-2}\cdots q_1 0,\ q_{h-1}q_{h-2}\cdots q_1 1,\ q_{h-1}q_{h-2}\cdots q_1 2$ and $q_{h-1}q_{h-2}\cdots q_1 3\ \}$ appear in the sequence, these four are replaced by the singly code $q_{h-1}q_{h-2}\cdots q_1$. With the knowledge of the resolution of the space in which the image is embedded, each string of quaternary digits can be interpreted correctly.

The quadcode representation of the tessellated space is what we adopt in the semi-quadcode labelling of the triangular patches of spherical surfaces. The triangular regions are tessellated exactly as in the quaternary triangular mesh but we always count two tricells as being contained within one quad cell except for the diagonal quad cells which contain only one tricell. The semi-quadcode addresses of the two QTM cell in a quad cell are formed by appending to the quadcode the digit 1 or 2 according to whether the tricell is upright or inverted.

3.3 Further Properties of SQC

Further properties of the SQC method are worth taking note of.

- Within any base triangular facet tessellated to level h, each of the X, and Y indices range over the set $\{0, 1, \ldots, 2^h - 1\}$ and the valid coordinate $\langle x, y \rangle$ of quad cells are those for which $(x + y) \leq 2^h - 1$.

- Starting from a perfect sphere and an inscribed octahedron, all cells resulting from tessellating a base triangular plane in the manner described have triangular shapes. However, the triangular patches deviate from equilateral to isosceles triangles and subsequently to cells of unequal edge length and unequal areas.

4 Operations on Semi-Quadcodes

One of the main virtues of the SQC addressing scheme is that it considerably simplifies the computations of neighbours of a cell. Other related operations such as detecting the boundary of SQC cells and connected component labelling are easily computed. Used as a global indexing scheme, the SQC facilitates conversion of latitude and longitude values to the SQC address containing the point. Conversely given and SQC code, one can compute the latitude and longitude values of three vertices of the tricell.

4.1 Neighbour Finding

In discussing neighbours of a SQC cell, we need to define a sense of direction in which the neighbour is required. We discuss edge-adjacent neighbours only in this paper. The same strategy can be extended to compute the addresses of the 5 vertex adjacent neighbours as well, if desired. Relative direction of a cell can be defined in two ways: either i) by the direction of the line joining a vertex to the midpoint of the opposite edge or ii) by referring to the direction in the sense of West, East and North or South. A tricell may have either a northern or southern neighbour depending on whether the cell in question is a u-tricell or an i-tricell. Direction in this paper is defined in the sense of the latter.

The suffixes of the SQC address, indicating the triangular cell type either upright or inverted, are two binary bits even though one bit suffices. Within each quad cell, a u-tricell is always an upright triangular cell, has the suffix code 1 and forms the lower off-diagonal triangle of a quad cell. An i-tricell is always inverted, has suffix digit 2 and forms the upper triangular cell of a quad cell. The edge adjacent cells of a u-tricell are all i-tricells and conversely the edge-adjacent cells of an i-tricell are all u-tricells.

Since the length of the quadcode implicitly defines the level for which the code applies, we will use the suffixes 0 to signify the end of a string representing a quad cell if the level is less than h. The terminal digit 3 in a quadcode of (k+1)-digits implies the presence of both the u-tricell and the i-tricell of the quad cell given by the k-digits. This means that semi-quadcodes at level k, $0 < k < h$, has k+1 quaternary digits. These terminal digits allows us to develop a very simple algorithm for determining the semi-quadcode of an edge adjacent neighbour to a given cell.

Algorithm for determining the edge adjacent neighbours

```
SemiQcode EdgeAdjacent (SemiQcode Q, Direction Dir)
{
        l ← lengthOf(Q) ; //length of SQC string
        tcode ← SuffixCode(Q); //extract the terminal digit.
        Q ← ClearRight(Q, 2); // delete the terminal digit
        ⟨x, y⟩ ← F^{-1}(Q);
        switch (Dir) {
        West:  if (− − x < 0) return (NullSQC) ;
               break ;
        East:  x++;
               if (x + y > 2^h − 1 ) return (NullSQC);
               break;
        North: if (tcode = 1 ∨ (+ + y > 2^h − 1) ) return (NullSQC)
               break
        South  if (tcode = 2 ∨ (− − y < 0 ) return (NullSQC)
        }      AdjToQ ← F(⟨x, y⟩);
        if (tcode = 1)
               AdjToQ ← AppendToQ(AdjToQ, 2);
        else
               AdjToQ ← AppendToQ(AdjToQ, 1);
        return (AdjToQ);
}
```

The value NullSQC is defined as '0' and indicates a possible error condition or an attempt to cross the boundary of the current base facet. The function SuffixCode() returns the suffix digit of the semi-quadcode, ClearRight(Q, n) drops the n suffix bits of the semi-quadcode Q, and the function AppendToQ(Q, d) appends the quadcode "d" as the terminal digit of Q. In a similar discussion of adjacent neighbour finding, detecting when the boundary of a base facet is crossed was not handled. Boundary detection is simple in the SQC scheme. Whenever any of the coordinate values of the pair $\langle x, y\rangle$, become negative or $(x + y) \geq 2^h - 1$ we know a boundary of the base facet is being crossed. In that case we translate our origin to the new origin of the adjacent base planar facet.

4.2 Detecting Sons and Parents of Cells

Related to the problem of neighbour finding is the question of determining the semi-quadcode address of either the sons or the parent of a tricell. For the QTM addressing scheme, the sons of a tricell addressed by $(q_{h-1}q_{h-2}\cdots q_0)$ are determined by appending the digits 0, 1, 2 or 3, to obtain $(q_{h-1}q_{h-2}\cdots q_0 0)$, $(q_{h-1}q_{h-2}\cdots q_0 1)$, $(q_{h-1}q_{h-2}\cdots q_0 2)$ and $(q_{h-1}q_{h-2}\cdots q_0 3)$. The address of the parent of a QTM tricell is obtained by simply discarding the trailing quaternary digit. We assume that in deriving the parent codes, the base facet code is denoted by null (Φ) unless the codes are prefixed by the facet number in which case it will be the base faced number.

In the SQC scheme, computing the address of the children or the parent is not that straight forward. It depends on the suffix digit q_0. If $q_0 = 1$ we have a u-tricell if $q_0 = 2$ we have an i-tricell. If $q_0 = 0$ or $q_0 = 3$ we have the occurrence of both an u- and i-tricells with a common parent. Let $(q_k q_{k-1}\cdots q_1 q_0)$ be the address of a semi-quadcode. If $q_0 = 1$, the children are $(q_k q_{k-1}\cdots q_1 01)$, $(q_k q_{k-1}\cdots q_1 02)$, $(q_k q_{k-1}\cdots q_1 11)$ and $(q_k q_{k-1}\cdots q_1 21)$. These are formed by replacing q_0 by the pairs of quaternary digits {01, 02, 11, 21}. Similarly if $q_0 = 2$ the children are derived by replacing q_0 with the pairs of digits { 12, 21, 31, 32 }.

The algorithm for the parent node of $(q_k q_{k-1}\cdots q_1 q_0)$ is the reverse operation for computing children and the pair of trailing digits "$q_1 q_0$" are replaced by either the digit 1 or 2. If $q_0 = 1$ then if also $q_1 < 3$ the parent address is $(q_k q_{k-1}\ldots 1)$ otherwise the parent address is $(q_k q_{k-1}\ldots 2)$. If $q_0 = 2$ then if also $q_1 > 0$ the parent address is $(q_k q_{k-1}\ldots 2)$

otherwise the parent address is $(q_k q_{k-1} \ldots 1)$.

4.3 Conversions from Latitude/Longitude to SQC Addresses

The angular measures of the location of a point on a sphere with respect to the origin in the center correspond to the geographic coordinates of latitude (ϕ) and longitude (λ). In anticipation of the application domain for which the SQC was developed, we will refer to the geographic coordinates (ϕ, λ) of a point on the globe. Although in reality the globe is an ellipsoid, we will continue to assume that we are dealing with spherical surfaces. This difference will impact our transformation algorithms but it is not essential for now. We will also limit our conversion algorithm to the 0 base plane since this is easily carried over to other base facet planes by changing the origin or the orientation of the X- Y- axes. The base facet 0 is assumed to be the quarter northern hemisphere lying above the equator ($\phi = 0$) and between longitudes $\lambda = 0$ and $\lambda = \pi/2$. Figure 3.1b illustrates the subdivision of the base 0 plane to level 2.

Let h be the highest level of subdivision of the base plane 0. We shall assume that the sense of orientation of triangles is as illustrated in figure 4.1. The vertices of a u-tricell are assumed to be numbered 0, 1 and 2 and those of an i-tricell as being numbered 1, 2 and 3. The X and Y axes in the corresponding $\langle x, y \rangle$ coordinate system are subdivided into 2^h intervals. The coordinate conversion algorithms is performed simply by first converting (ϕ, λ) values into $\langle x, y \rangle$ and then converting from $\langle x, y \rangle$ coordinates to the SQC address. First we state the following propositions. Their proofs are straight forward and are left out.

Proposition 4.1 *Let (ϕ, λ) be the angular coordinate of a point on the surface of a spherical region defined by the lines of longitude $\phi = 0$, $\phi = \pi/2$ and the latitude $\lambda = 0$. Let the degree of tessellation of the region into quaternary triangular mesh be h. Then in the SQC encoding of the space, the corresponding $\langle x, y \rangle$ coordinate of the quad cell that the point lies is defined by*

$$y = \lfloor \frac{2^{h+1} * \phi}{\pi} \rfloor \tag{1}$$

and

$$x = \lfloor \frac{2\lambda(2^h - y)}{\pi} \rfloor. \tag{2}$$

Proposition 4.2 *Let (ϕ, λ) be the angular coordinates of a point in the quarter hemisphere that is defined by the lines of longitudes $\lambda = 0$ and $\lambda = \pi/2$ and let this point lie in the quad cell given by the $\langle x, y \rangle$ in the SQC tessellation method. If the three vertices of the u-tricell in the corresponding quad cell are denoted by (ϕ_0, λ_0), (ϕ_1, λ_1) and (ϕ_2, λ_2), then these angular coordinates are defined by the expressions:*

$$\phi_0 = y * \tfrac{\pi}{2^{h+1}}; \quad \lambda_0 = x * \frac{\pi}{2(2^h - y)} \tag{3}$$

$$\phi_1 = (y+1) * \tfrac{\pi}{2^{h+1}}; \quad \lambda_1 = x * \frac{\pi}{2(2^h - y - 1)} \tag{4}$$

$$\phi_2 = y * \tfrac{\pi}{2^{h+1}}; \quad \lambda_2 = (x+1) * \frac{\pi}{2(2^h - y)} \tag{5}$$

The propositions (4.1) and (4.2) form the bases for the conversion algorithm from angular coordinates to SQC address and vice versa. We assume that within a given cell ϕ varies linear with λ. The algorithm to derive the SQC address given the angular coordinate (ϕ, λ) is stated below.

```
Semi-Qcode Address(integer h, double φ, double λ)
{
        integer x, y ;
        Semi-Qcode Q;
        y ← ⌊(2^(h+1) * φ) / π⌋;
        x ← ⌊2λ(2^h − y) / π⌋;
        φ1 ← (y + 1) * π / 2^(h+1);
        λ1 ← x * π / (2(2^h − y − 1));
        φ2 ← y * π / 2^(h+1);
        λ2 ← (x + 1) * π / (2(2^h − y));
        Q ← F(⟨x, y⟩) ;
        if (φ ≥ ((φ1 − φ2)/(λ1 − λ2)) * (λ − λ1) + φ2 )
                Q ← AppendToQ(Q,2) ;
```

```
    else
        Q ← AppendToQ(Q,1);
    return (Q);
}
```

The above algorithm for computing the SQC address of a point given its location in angular measure is O(1). Note also that it implicitly computes the angular coordinates of two diagonal vertices of the corresponding quad cell. This is easily extended to compute all three vertices of a given SQC address.

4.4 Other Operations

In general an image on the sphere will be given as a set of semi-quadcodes. Other desired operations on the codes are intersection, union and difference of two sets of semi-quadcodes. Although the triangular facets at any given resolution are not all of the same shape, for any given radius and a given resolution, the area and edge lengths of the triangles, depending on the level and address are predefined. For this reason, it is easy to compute the area of a figure inscribed on a spherical surface, to a sufficiently good approximation, given the set of SQC addresses that represent it. Another important operation often desired is the connected component labeling of images on the sphere. All of these operations are easily computed by representing the image as a set of SQC addresses.

5 Applications of SQC

The semi-quadcode was the result of developing data structures for building a model of the globe as an interface to a spatial browser for a global databases. Since its development, it has found a number of applications. The SQC scheme, we believe, is an alternative scheme for applications currently supported by the QTM and SQT schemes. We illustrate this with an application of the SQC to global indexing.

5.1 Global Indexing

One major application of the SQC is for organizing spatial indices into geoscience database. Consider the alternative in which the globe is subdivided along the boundaries of parallels and meridians, at say intervals of 6 degrees. Let each of the quadrangles by further subdivided into regions using the quadtree tessellation method. Traditionally, these subregions correspond to map sheets of increasing map scales. If we now consider that the information related to one of these defined regions are clustered, then we need a consistent method of indexing the locations where each cluster of information is stored.

The problem of global indexing has been previously addressed [2, 20, 14, 16, 15]. One idea suggested is a decomposition of space along lines of meridians and parallel so that regions are assigned their equivalent linear quadtree labels. The quadtree labels serve as the keys to locations where the clusters of information related to the regions are stored. These linear quadtree codes are then organized into B^+-tree index. The main objection to the scheme is the fact the spherical surface subdivided along lines of meridian and parallel, do not maintain the same shape. They are triangular at the poles and rectangular along the 0-parallel.

Since the SQC method maintains the triangular shapes for any level of decomposition, it serves as a better alternative for global indexing. First we note that, given one of the initial 8 base planar facets, the semi-quadcode addresses of regions up to level 13 can be compacted into a 32-bit integer words. The first 4 bits gives the length of the semi-quadcode, the right most 2-bits give the tricell type, i.e., "01" for a u-tricell, "10" for an i-tricell and "11" for the occurrence of two triangular cell of the same parent in quad cell. This representation assumes that the whole spherical surface is represented as a forest of 8 distinct semi-quadtrees.

To represent semi-quadcodes up to 19 levels for the whole spherical surface (equivalent to a resolution of about $6m \times 6m$ for the globe), requires at most a 48-bit (or a 6-byte) words. Within the first byte, the first 3-bits represent the facet number, the next 5-bits represent the length of the semi-quadcode. Of the remaining 40-bits, the last 2-bits designate the cell type. This leaves 38 bits to maintain at most 19 levels (or 19 digits) of the semi-quadcode.

To use the semi-quadcode as a global index, we simply tessellate the surface of the sphere in the manner consistent with the semi-quadcode scheme. For each region that has information to be stored, we associate with it the corresponding semi-quadcode address as its key. Let this location be perceived as a location pointer. The pairs of values (key and pointer) are then organized as the content of the leaf nodes of a B^+-tree.

Irrespective of the level or resolution of the region for which information is maintained, this indexing method gives the same access complexity of $O(\log N)$ for an index of N cells. The main drawback of the scheme is that information must be grouped in units consistent with the triangular subdivision of space. This historically has not been the case. For example map sheets are not triangular in shape. This means that we have to go through the exercise of restructuring and reorganizing information, and consequently map sheets, according to the triangular shapes.

5.2 Seamless Horizontal and Vertical Navigation

Our treatment of the SQC scheme has concentrated on the representation of one of the base triangular facet. We have also illustrated how to detect when facet boundaries are crossed. All our algorithms are easily extended to cover the entire sphere, simply by detecting when a boundary cell is crossed and then making the appropriate change of origin and orientation of the axes.

One immediate requirement in the application of the SQC method to global indexing scheme is the realization of a seamless traversal of cells that are at the same level of tessellation or traversal of the cells at different levels. We refer to the former as a *"horizontal navigation"* of space and to the latter as *"vertical navigation"*. To access information related to an adjacent tricell of some given one we apply the neighbour finding algorithm to locate the address of the neighbour tricell. This address is then used as the search key in the B^+-tree index to locate the location of the data related to the new tricell. In the same manner, we can access information related to one of the children of a given cell by first applying the algorithm for determining the children of a node. The SQC address computed is then used as the search key of the index tree to locate the information related to the higher level cell.

6 Conclusion

We have illustrated how the SQC scheme is used for addressing tricells resulting from the triangular decomposition of the surface of a sphere. By this means we are able to treat region addressing on spherical surface much like quadcode addressing of quadtree subdivision of planer regions.

The SQC scheme is an alternative to the QTM and SQT representation of spherical surfaces. The main virtues of our scheme is that it gives simplified and efficient algorithms for nearly all the operations required to represent an image on a spherical surface. Like the QTM and SQT methods, it has applications for:

- global spatial indexing;
- developing a model of the globe in 3D instead of reliance on projections onto planes;
- improving the speed of rendering maps on the globe;
- rendering information on spherical surfaces;
- developing a spatial browser interface for rapid data selection and analysis;
- developing a new dynamic projection of the globe that varies with change of origins of the planes.

Using 32-bit integer words, with each base planar facet represented independently, we can achieve up to 13 SQC digits, or 13 levels of tessellations. With 48-bit words we can achieve up to 19 SQC digits and still address location on the sphere to sufficient high degree of precision. Note that the errors resulting from projections, diminish by projecting onto triangular facets at high levels of subdivision. Presently, our use of SQC is in modeling the globe. This is the first level of interaction for a spatial browser used to explore the metadata of the Canadian Geomatic databases. Other scheduled works include its use in terrain model building on surfaces of the globe. QTM and SQT form some of the early works in global geographic information systems. The SQC scheme provides an enhanced but simplified scheme that is used to achieve the same objectives.

Acknowledgement

This work is supported in part by a joint grant from the Geographic Information Systems Division, Surveys Mapping and Remote Sensing Sector, Energy Mines Resources, Canada and the Natural Sciences and Engineering Research Council of Canada.

References

[1] R. Bayer and E. McCreight. Organization and maintenance of large ordered indexes. *Acta Informatica*, 1:173 – 189, 1972.

[2] Z. T. Chen. Quad tree spatial spectra — its generation and applications. In *Proc. International Symposium on Special Data Handling*, pages 218 – 237, 1984.

[3] D. Comer. The ubiquitous b-tree. *ACM Comput. Surveys*, 11(2):121 – 137, Jun. 1979.

[4] G. H. Dutton. Geodesic modelling of planetary relief. *Cartographica*, 21(2&3):188 – 207, 1984.

[5] G. H. Dutton. Locational properties of quaternary triangular meshes. In *Proc. 4th Int'l Symp. on Spatial Data Handling*, pages 901 – 910, Zurich, Switzerland, 1990.

[6] G. H. Dutton. Planetary modelling via hierarchical tessellation. In *Proc. AutoCarto 9, ACSM-ASP*, pages 462 – 471, Baltimore, U.S.A., 1989.

[7] G. Fekete. Rendering and managing spherical data with sphere quadtrees. In *Proc. Visualization '90*, pages 176 – 186, San Francisco, 1990.

[8] G. Fekete and L. S. Davis. Property spheres: a new representation for 3-d object recognition. In *Proceedings of the workshop on Computer Vision: Representation and Control*, pages 192 – 201, Annapolis, MD, April 1984. (also University of Maryland Computer Science Tech Report-1355).

[9] I. Gargantini. An effective way to represent quad-trees. *Comm. ACM*, 25(12):905 – 910, Dec. 1982.

[10] I. Gargantini. Linear octtrees for fast processing of three dimensional objects. *Comput. Gr. Image Process.*, 20(4):365 – 374, Dec. 1982.

[11] M. F. Goodchild and Y. Shiren. A hierarchical data structure for global geographical information systems. *Graphical Models and Image Processing*, 54(1):31 – 44, Jan. 1992.

[12] S. X. Li and M. H. Loew. Adjacency detection using quadcodes. *Comm. ACM.*, 30(7):627 – 631, Jul. 1987.

[13] S. X. Li and M. H. Loew. The quadcode and its arithmetic. *Comm. ACM.*, 30(7):621 – 626, Jul. 1987.

[14] F. D. Libera and F. Gosen. Using b-trees to solve geographic range queries. *Comput, Journal*, 29(2):176 – 181, 1986.

[15] D. M. Mark and J. P. Lauzon. Approaches for quadtree-based geographic information systems at a continental or global scales. In *Proc. AutoCarto 7, Digital Representation of Spatial Knowledge*, Amer. Soc. of Photogamm., Falls Church, Virginia, Mar. 1985.

[16] D. M. Mark and J. P. Lazon. Linear quadtrees for geographic information systems. In *Proc. Int'l. Symp. on Spatial Data hasndling*, pages 412 – 430, Zurich, Switzerland, Aug. 1984.

[17] E. J. Otoo. *Semi-Quadcode and Its Operations on Spherical Surfaces.* Technical Report, School of Computer Science, Carleton Univ., Ottawa, Canada, 1992.

[18] H. Samet. *Applications of Spatial Data Structures.* Addison-Wesley, Reading, Mass., 1990.

[19] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, Reading, Mass., 1990.

[20] W. Tobler and Z-T. Chen. A quadtree for global information storage. *Geographical Analysis*, 18(4):360 – 371, Oct. 1986.

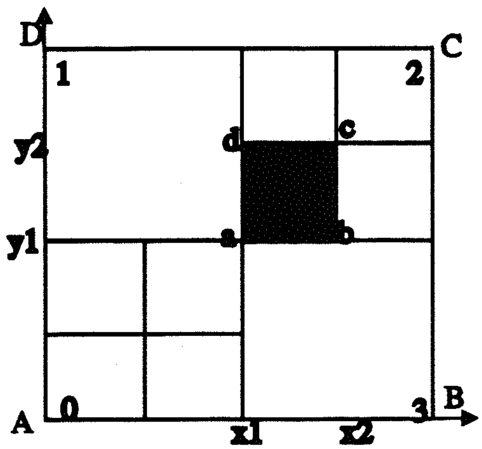

Figure 1.1: A quadtree tessellated space

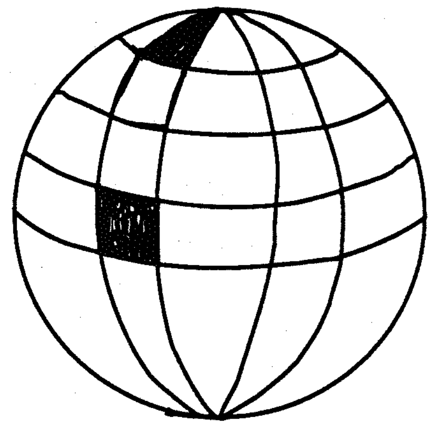
Figure 1.2: Tessellation of Spherical Surface along the lines of meridians and parallels

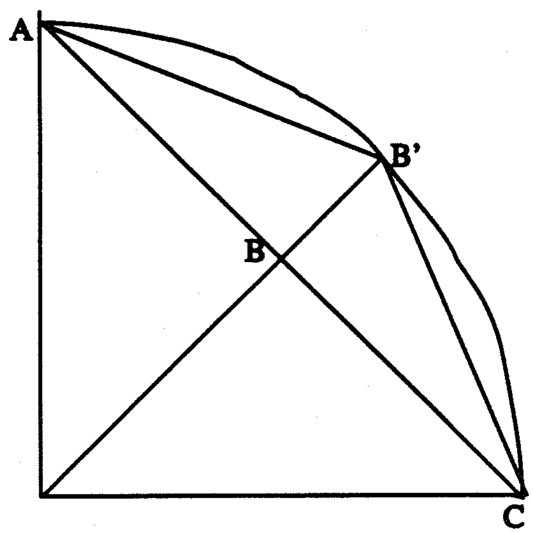

Figure 2.1a: 1st level approx. of the arc AB'C by the line segments AB' and B'C

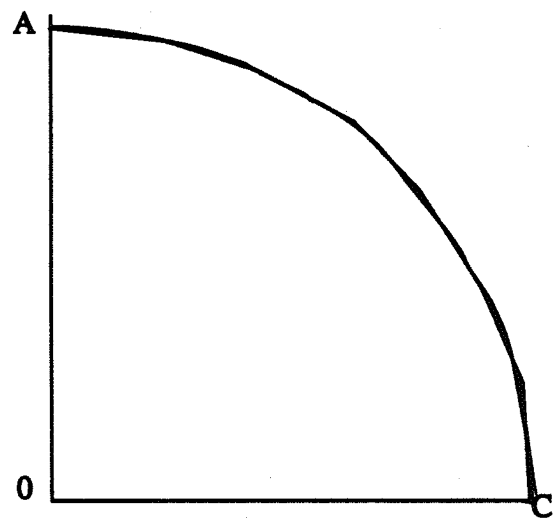

Figure 2.1b: Approx. of the curve AC by a sequence of line segments.

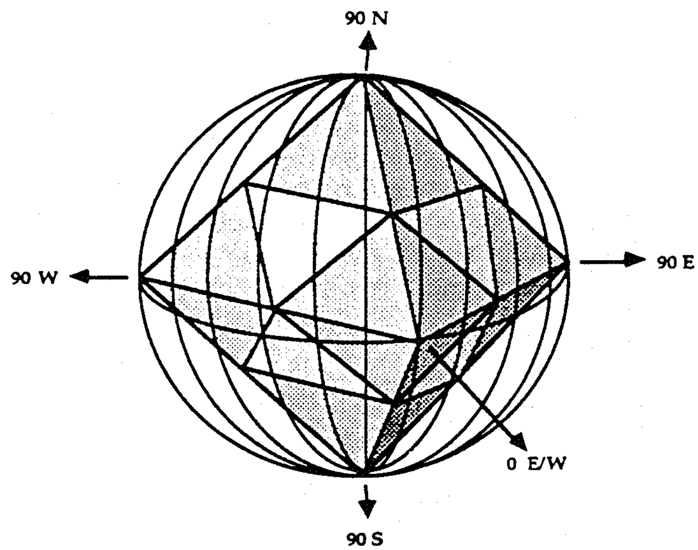

Figure2.2a: The base octahedron inscribed sphere

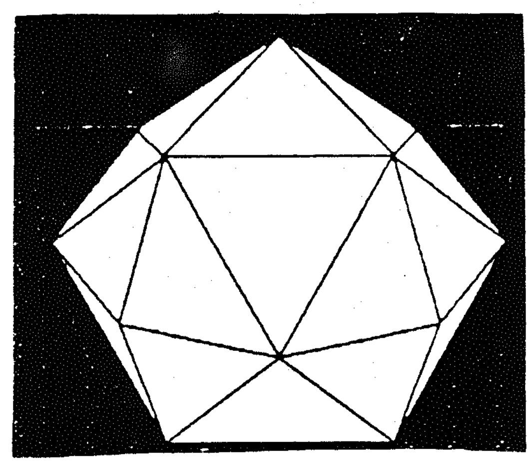

Figure 2.2b: The level 1 development of the Quaternary Triangular Mesh.

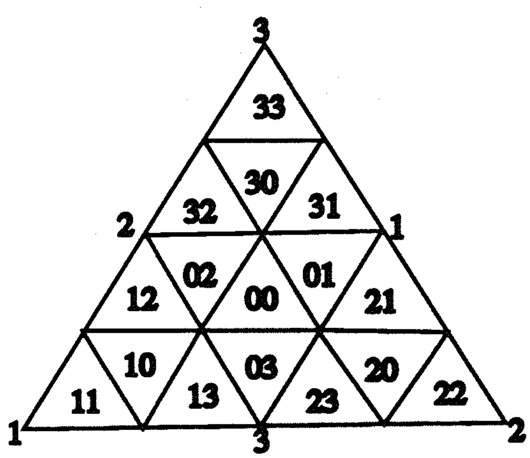

Figure 2.3a: QTM labelling of triacons

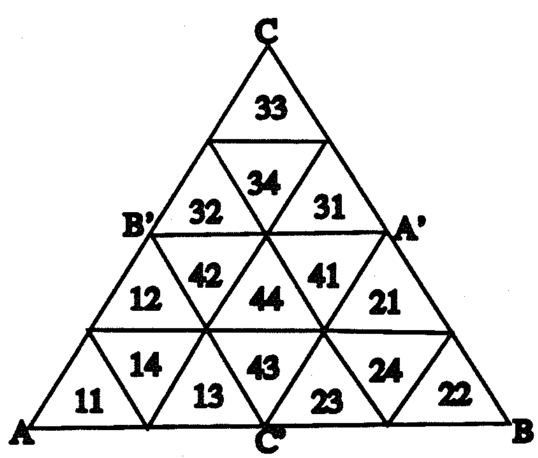

Figure 2.3b: SQT labelling of trixels

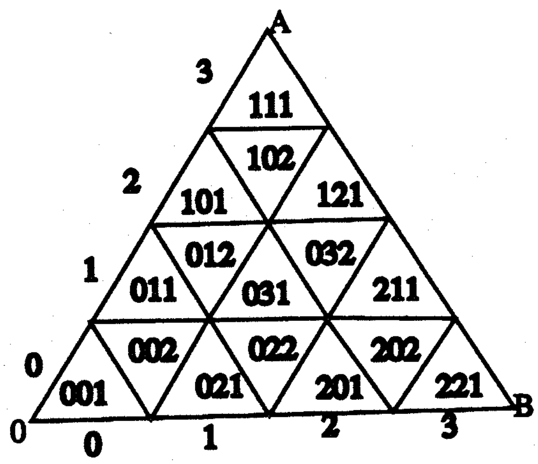

Figure 3.1a: SQC labelling of tricells

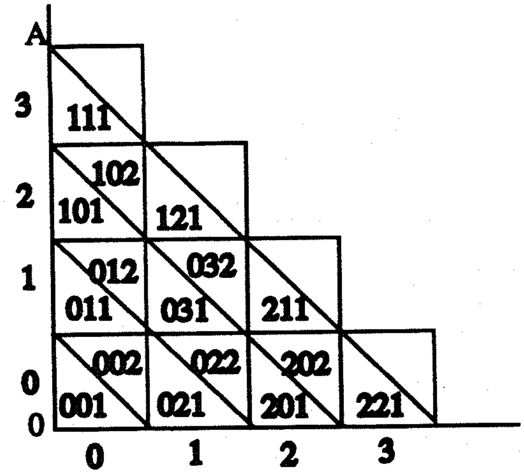

Figure 3.1b: Graph of triangular facets orientated to show correpond with quadtree tessellation.

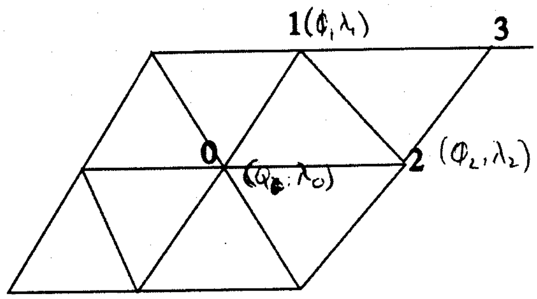

Figure 4.1: Illustration of transformation from angular measures to SQC labels.

School of Computer Science, Carleton University
Recent Technical Reports

TR-176 **Edge Separators of Planar and Outerplanar Graphs with Applications**
Krzystof Diks, Hristo N. Djidjev, Ondrej Sykora and Imrich Vrto, May 1990

TR-177 **Representing Partial Orders by Polygons and Circles in the Plane**
Jeffrey B. Sidney and Stuart J. Sidney, July 1990

TR-178 **Determining Stochastic Dependence for Normally Distributed Vectors Using the Chi-squared Metric**
R.S. Valiveti and B.J. Oommen, July 1990

TR-179 **Parallel Algorithms for Determining K-width-Connectivity in Binary Images**
Frank Dehne and Susanne E. Hambrusch, September 1990

TR-180 **A Workbench for Computational Geometry (WOCG)**
P. Epstein, A. Knight, J. May, T. Nguyen, and J.-R. Sack, September 1990

TR-181 **Adaptive Linear List Reorganization under a Generalized Query System**
R.S. Valiveti, B.J. Oommen and J.R. Zgierski, October 1990

TR-182 **Breaking Substitution Cyphers using Stochastic Automata**
B.J. Oommen and J.R. Zgierski, October 1990

TR-183 **A New Algorithm for Testing the Regularity of a Permutation Group**
V. Acciaro and M.D. Atkinson, November 1990

TR-184 **Generating Binary Trees at Random**
M.D. Atkinson and J.-R. Sack, December 1990

TR-185 **Uniform Generation of Combinatorial Objects in Parallel**
M.D. Atkinson and J.-R. Sack, January 1991

TR-186 **Reduced Constants for Simple Cycle Graph Separation**
Hristo N. Djidjev and Shankar M. Venkatesan, February 1991

TR-187 **Multisearch Techniques for Implementing Data Structures on a Mesh-Connected Computer**
Mikhail J. Atallah, Frank Dehne, Russ Miller, Andrew Rau-Chaplin, and Jyh-Jong Tsay, February 1991

TR-188 **Generating and Sorting Jordan Sequences**
Alan Knight and Jörg-Rüdiger Sack, March 1991

TR-189 **Probabilistic Estimation of Damage from Fire Spread**
Charles C. Colbourn, Louis D. Nel, T.B. Boffey and D.F. Yates, April 1991

TR-190 **Coordinators: A Mechanism for Monitoring and Controlling Interactions Between Groups of Objects**
Wilf R. LaLonde, Paul White, and Kevin McGuire, April 1991

TR-191 **Towards Decomposable, Reusable Smalltalk Windows**
Kevin McGuire, Paul White, and Wilf R. LaLonde, April 1991

TR-192 **PARASOL: A Simulator for Distributed and/or Parallel Systems**
John E. Neilson, May 1991

TR-193 **Realizing a Spatial Topological Data Model in a Relational Database Management System**
Ekow J. Otoo and M.M. Allam, August 1991

TR-194 **String Editing with Substitution, Insertion, Deletion, Squashing and Expansion Operations**
B John Oommen, September 1991

TR-195 **The Expressiveness of Silence: Optimal Algorithms for Synchronous Communication of Information**
Una-May O'Reilly and Nicola Santoro, October 1991

TR-196 **Lights, Walls and Bricks**
J. Czyzowicz, E. Rivera-Campo, N. Santoro, J. Urrutia and J. Zaks, October 1991

TR-197 **A Brief Survey of Art Gallery Problems in Integer Lattice Systems**
Evangelos Kranakis and Michel Pocchiola, November 1991

TR-198 **On Reconfigurability of Systolic Arrays**
Amiya Nayak, Nicola Santoro, and Richard Tan, November 1991

TR-199 **Constrained Tree Editing**
B. John Oommen and William Lee, December 1991

TR-200 **Industry and Academic Links in Local Economic Development: A Tale of Two Cities**
Helen Lawton Smith and Michael Atkinson, January 1992

TR-201 **Computational Geometry on Analog Neural Circuits**
Frank Dehne, Boris Flach, Jörg-Rüdiger Sack, Natana Valiveti, January 1992

TR-202 **Efficient Construction of Catastrophic Patterns for VLSI Reconfigurable Arrays**
Amiya Nayak, Linda Pagli, Nicola Santoro, February 1992

TR-203 **Numeric Similarity and Dissimilarity Measures Between Two Trees**
B. John Oommen and William Lee, February 1992

TR-204 **Recognition of Catastrophic Faults in Reconfigurable Arrays with Arbitrary Link Redundancy**
Amiya Nayak, Linda Pagli, Nicola Santoro, March 1992

TR-205 **The Permutational Power of a Priority Queue**
M.D. Atkinson and Murali Thiyagarajah, April 1992

TR-206 **Enumeration Problems Relating to Dirichlet's Theorem**
Evangelos Kranakis and Michel Pocchiola, April 1992

TR-207 **Distributed Computing on Anonymous Hypercubes with Faulty Components**
Evangelos Kranakis and Nicola Santoro, April 1992

TR-208 **Fast Learning Automaton-Based Image Examination and Retrieval**
B. John Oommen and Chris Fothergill, June 1992

TR-209 **On Generating Random Intervals and Hyperrectangles**
Luc Devroye, Peter Epstein and Jörg-Rüdiger Sack, July 1992

TR-210 **Sorting Permutations with Networks of Stacks**
M.D. Atkinson, August 1992

TR-211 **Generating Triangulations at Random**
Peter Epstein and Jörg-Rüdiger Sack, August 1992

TR-212 **Algorithms for Asymptotically Optimal Contained Rectangles and Triangles**
Evangelos Kranakis and Emran Rafique, September 1992

TR-213 **Parallel Algorithms for Rectilinear Link Distance Problems**
Andrzej Lingas, Anil Maheshwari and Jörg-Rüdiger Sack, September 1992

TR-214 **Camera Placement in Integer Lattices**
Evangelos Kranakis and Michel Pocchiola, October 1992

TR-215 **Labeled Versus Unlabeled Distributed Cayley Networks**
Evangelos Kranakis and Danny Krizanc, November 1992

TR-216 **Scalable Parallel Geometric Algorithms for Coarse Grained Multicomputers**
Frank Dehne, Andreas Fabri and Andrew Rau-Chaplin, November 1992

TR-217 **Indexing on Spherical Surfaces Using Semi-Quadcodes**
Ekow J. Otoo and Hongwen Zhu, December 1992