地理信息科学
Geographic Information Sciences
第一卷第一期 一九九五年六月
Vol. 1, No. 1, June 1995

An Algorithm for Searching Boundary Octants in 3D Geological Subsurface Modeling

Rongxing Li and Changshi Xu*

Department of Geomatics Engineering
The University of Calgary
Calgary, Alberta
Canada T2N 1N4

Abstract

In geological subsurface modeling, large quantities of data are usually involved, which require extensive efforts to process using traditional methods. In addition, 3D representations and spatial operations are essential for computerised processing. With the recent advance in Geographic Information Systems (GIS) and computer technology, a wide range of spatial analysis and processing procedures for geological subsurface modeling are expected to be improved if 2D and 2.5D GIS functions are to be extended to handle 3D objects such as subsurface layers.

This paper gives a brief description of a 3D system for modeling geological subsurface information developed at the Department of Geomatics Engineering, The University of Calgary. In this system, an octree data structure is employed to model 3D information due to its efficiency in Boolean operations, compactness in storing data and capability of intelligent data access. Surface representations are used for visualisation purposes. An algorithm for searching neighbour octants encoded by Peano keys is presented. Instead of dividing large octants into smallest ones at the bottom level, it finds neighbour octants of both the same size and different sizes based on original octants at all levels. In particular, large octants which are partially covered by other smaller octants can be found.

The system has been developed on a Silicon Graphics Workstation and based on the GL (Graphics Library) and "C" programming language.

I. INTRODUCTION

Geological subsurface modeling has been applied in computer-assisted petroleum and mining explorations (Turner 1990 and Hughes 1994). This technology is supported by advanced geological field exploration methods and subsequent data interpretation using geostatistic theory. The accumulatively generated large volume of digital subsurface information, increasingly demanding research, and application practices require geoscientific information systems for efficient management, analysis, visualisation, and decision - making support.

Geographic Information Systems (GIS) have been very successful in handling two dimensional (2D) features such as parcels and road networks (Maguire et al 1991). In addition, objects such as terrain surfaces are modeled using so called 2.5 dimensional representations where for each planimetric point in the x-y plane there is only one corresponding value in the z - direction. This is obviously one of the major barriers for most existing commercial GIS to be applied in geological subsurface modeling in which multiple vertical strata layers must be represented and analysed. A geoscientific information system capable of handling geological subsurface data should have its own data structures supporting sophisticated 3D analysis, strong topological relations between strata layer objects, and attribute queries.

Research efforts have been made to develop such a system since the last decade. For example, elevation information was treated as attributes in order to display and query the third dimensional information in 2D GIS (ESRI 1992). Polyhedra based boundary representations were applied to model subsurface objects and converted into octrees (Tamminen and Samet 1984). Special problems in

* Currently with Multi-Corp Inc., Calgary, Alberta, Canada

1082-4006/95/0101-23$3.00

representing field exploration data in 3D GIS were discussed in Harbaugh and Martinez (1990). A systematic survey of 3D representations was made by Raper and Kelk (1991) and Li (1994) discussed application issues of 3D data structures.

This paper introduces the general concept of a system for geological subsurface modeling developed at the Department of Geomatics Engineering of The University of Calgary. An algorithm for searching boundary octants, one of many octree based functions of this system, is presented in detail.

II. BACKGROUND

A System for Geological Subsurface Modeling

The presented system aims at 3D subsurface modeling to improve oil, gas, and mining explorations. Subsurface data are loaded into the system as 3D multi-layers. For each grid point on the x-y plane, depth values of top and bottom surfaces of each layer are registered in the z direction to form the geometric description of the subsurface layers to which layer attributes are attached. Efficient data structures and modeling tools are provided for visualizing, analysing, and managing the 3D geological subsurface data. The system is developed in C programming language and based on a strong Graphics Library (Silicon Graphics, Inc. 1992) package which takes advantages of hardware of Silicon Graphics.

Two modelers coexist in the system: a surface modeler and an octree modeler. The surface modeler is used to convert the input data into a surface model according to the top-bottom layer-surface information; while the octree modeler transforms the same data into an octree model if necessary. Since the surface model and the octree model depict the same objects by using surface and solid geometric information respectively, the efficiency of modeling functions based on these two models is also different (Li 1994). For example, the surface model gives a relatively realistic shaded surface for visualisation because subtle normal vector changes of the surfaces can be represented. This is especially important when the lighting function is used. On the other hand, octants have six faces which, in turn, give only six normal vectors parallel to three principal axes. If the resolution of the octree model is set as the same as that of the input data, there is no loss of geometric information in the resulting octree model. However, the graphic quality of the octree model display is not comparative to the realistically shaded surface model because of the restriction of normal vector directions. Furthermore, layer-related topology can be constructed in surface models.

One of the major advantages of octree models is efficient Boolean operations because of the simple geometry and topology of octants. If encoded by Peano keys (Laurini and Thompson 1992), some spatial operations can be carried out at the bit level. In this system, octree models are, therefore, used to perform 3D spatial operations for analysis and simulations. Consequently, the system maintains two kinds of models for the same loaded object, namely the surface model for visualisation and the octree model for spatial operations.

Since most spatial operations are based on the octree representation the efficiency of octree operations often determines system responses to users requests. In special cases of geological subsurface modeling with large layer datasets, this is especially true. Among others, one of the critical and frequently used basic spatial operations is finding neighbour octants for a given octant. An application of this basic function in subsurface modeling could be to find all octants on the surface, for example, for a conversion from an octree to a surface model. Surface boundary lines can then be extracted from the boundary octants. The same basic function is also used in the operations to cut a subsurface model by a plane or a half cylinder face so that the intersection profile of the solid surface model is exposed for material queries and geological interpretations. Figure 1 shows one of the examples in geological subsurface modeling. "fences" are interactively defined on a 3D subsurface model (Figure 1(a)). Spatial operations are required to perform the intersection between the "fences" (multi-planes) and the model. The front part of the model is then removed so that the defined profile can be visualized (Figure 1(b)). In light of the above facts, it is necessary to develop an efficient algorithm for finding boundary octants in order to support quick system responses to users' octree-based requests.

Peano Coding in Octree Representations

Octrees are a natural 3D extension from 2D quadtrees. An octree represents a 3D volumetric object by recursively subdividing the object space using a tree structure. At the root level is the object

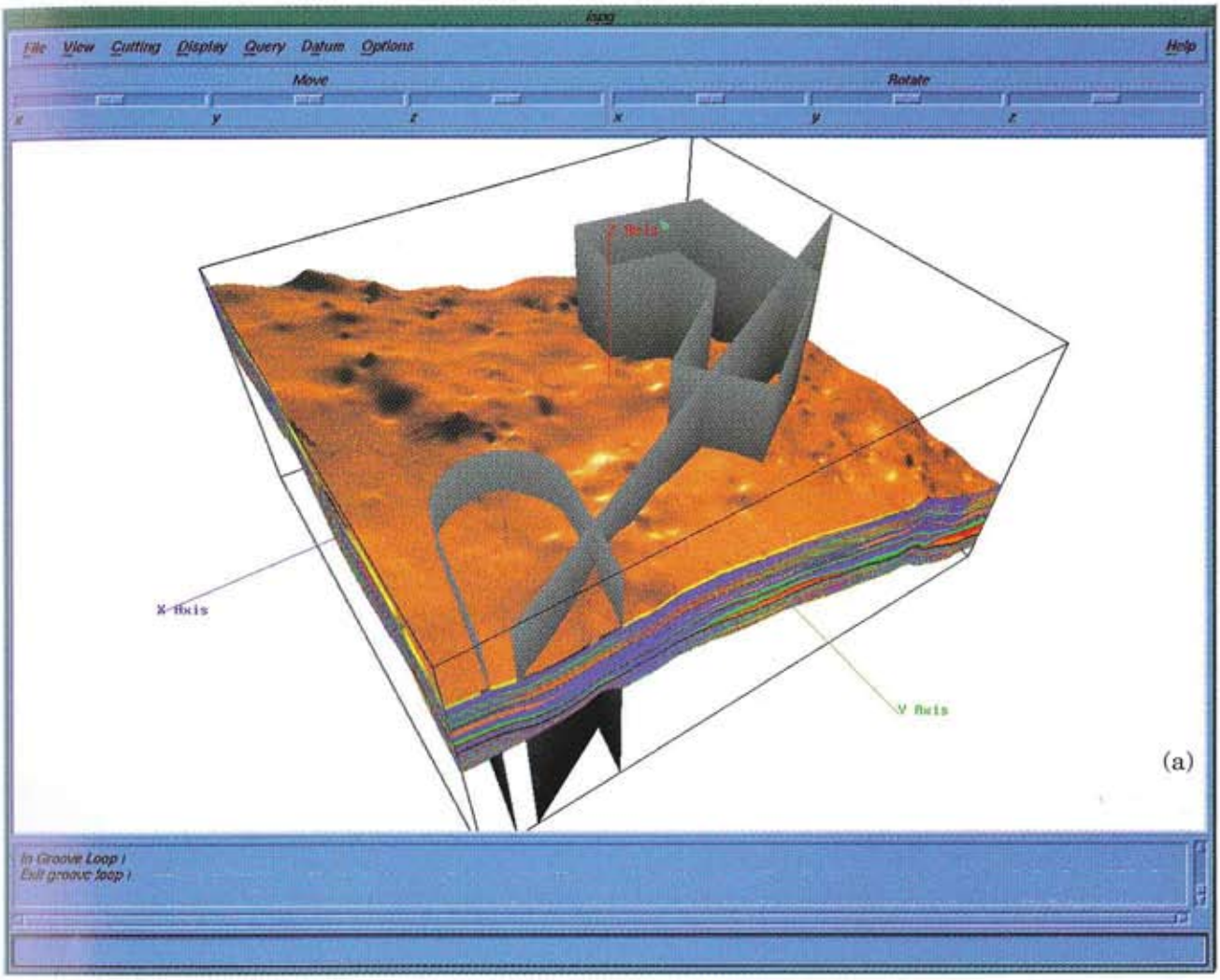

Figure 1. One of examples in geological subsurface modeling: interactive definition of "fences" on a subsurface model (a) and removal of the front part of the model using spatial operations (b).

itself contained in a cube which is called universe. The universe is subdivided into eight octants at the next level. Octants that are not entirely occupied by the object are called partial (P) octants and are further subdivided into smaller octants at the lower level. This partition procedure continues until all suboctants are either empty (E, no object material in the octants) or full (F, octants fully occupied by the object material). The procedure could also stop when the desired resolution is reached. Details of octree representations can be found in Samet (1990) and Li (1994). An extended octree model can be used to describe objects with regularly shaped surfaces with vertices, lines, and faces defined in octants (Laurini and Thompson 1992).

Although there are various octree encoding methods, it is noted that a linear octree, a pointerless structure first proposed by Gargantini (1982), locates any node in an octree by a unique key which can be obtained by interleaving binary coordinates of X, Y and Z. This can be implemented by using Peano encoding. Figure 2 shows an example of filling a 2D space using the Peano N curve and 1D Peano keys. The N pattern of the N curve repeats itself until the 2D space is filled out. Each of the N covers 4 pixels and pixels along the curve are encoded starting from 0. This N curve can be extended in 3D space with its encoding scheme illustrated in Figure 3 (a). Instead of 3D indices, an octree can be encoded using 1D Peano encoding. An octant O is described by its Peano key and octant size: [P(O), S(O)]. For instance, [8, 2] and [33, 1] depict two octants with different sizes (Figure 3 (b)). If the tree

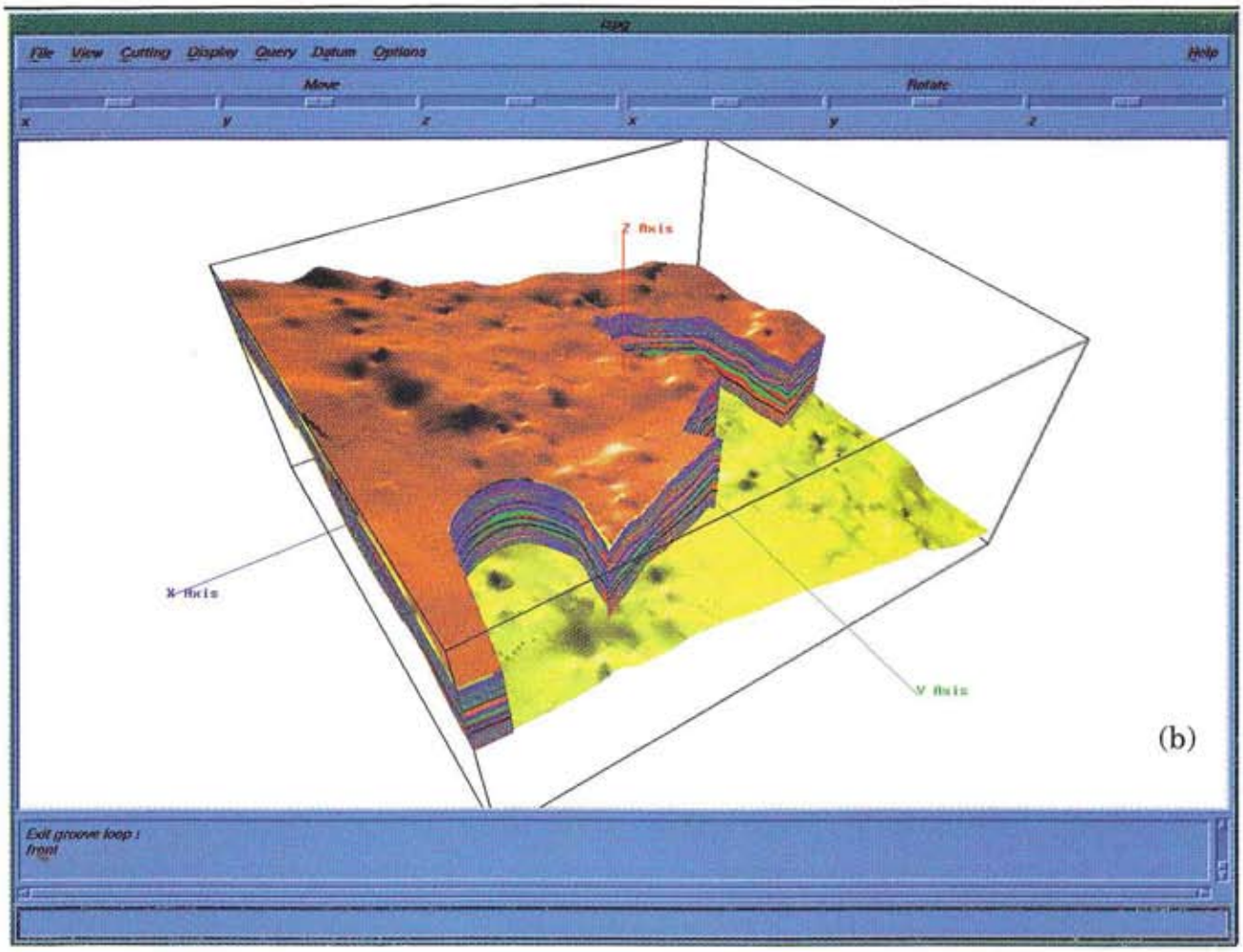

Figure 1. (Continued)

structure (Figure 3 (c)) is used, the second parameter for octant sizes is unnecessary.

The Peano key can be generated by interleaving binary coordinates of (X, Y, Z). For example, an octant with P(O)=14 has decimal coordinates (1, 1, 2). The corresponding binary coordinates are (01, 01, 10). Interleaving the binary coordinates in the order of Z, Y, and X for every bit starting from lower bits results in the binary Peano key 001110 which is 14 in decimal. This nature of the Peano keys makes the storage of Peano keys compact and the conversions between Peano keys and coordinates very efficient. In addition, neighbourhood searching in a Peano encoded octree is efficient because of the characteristics of the N curve (Laurini and Thompson 1992).

III. A PEANO-ENCODING BASED ALGORITHM FOR SEARCHING BOUNDARY OCTANTS

The first step of deriving the 3D surface information from an octree representation of an object can be simplified by finding all boundary octants which can be viewed as a 3D boundary of the object. The detection and analysis of the 3D boundaries were reported by a number of researchers (Liu 1977, Arzty et. al. 1981, and Ibaroudene et. al. 1990). Using repeated elimination of the interior of surfaces, Atkinson et al. (1985) introduced an algorithm by a linked list and followed by a node expansion. These algorithms are based on octants of the minimum size in linear octrees, that is, octants at the lowest level. Any octant larger than that has to be divided recursively until the lowest level is reached. In visualisation and analysis of geological features,

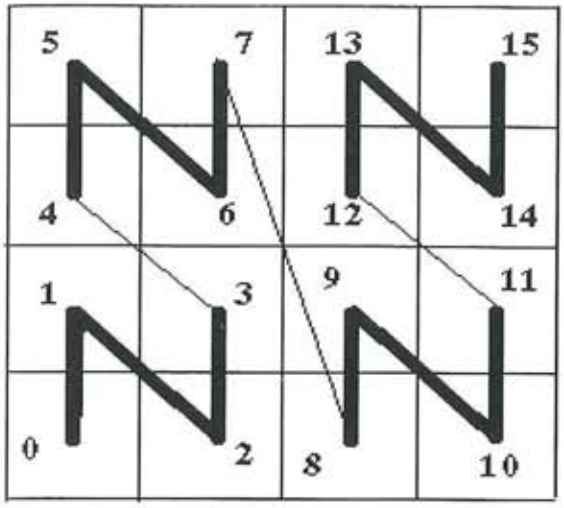

Figure 2. N curve and Peano encoding.

S: South, negative Z direction;
W: West, negative Y direction;
F: Front, positive X direction; and
B: Back, negative X direction.

If d is one of these six directions, the opposite direction is denoted by d'. In an octree with n levels, an octant, denoted as O, generated after i subdivisions has a *grouping factor* $g(O) = (n - i)$, which is an aggregative indicator of the octant.

d_face is defined as the unique face of the octant O in the direction d{N,E,B,S,W,F}. The octant which has the same size as O and shares the d_face of O is called an *exact neighbour octant* of O and denoted as O_d. The shared face is called d'_face of

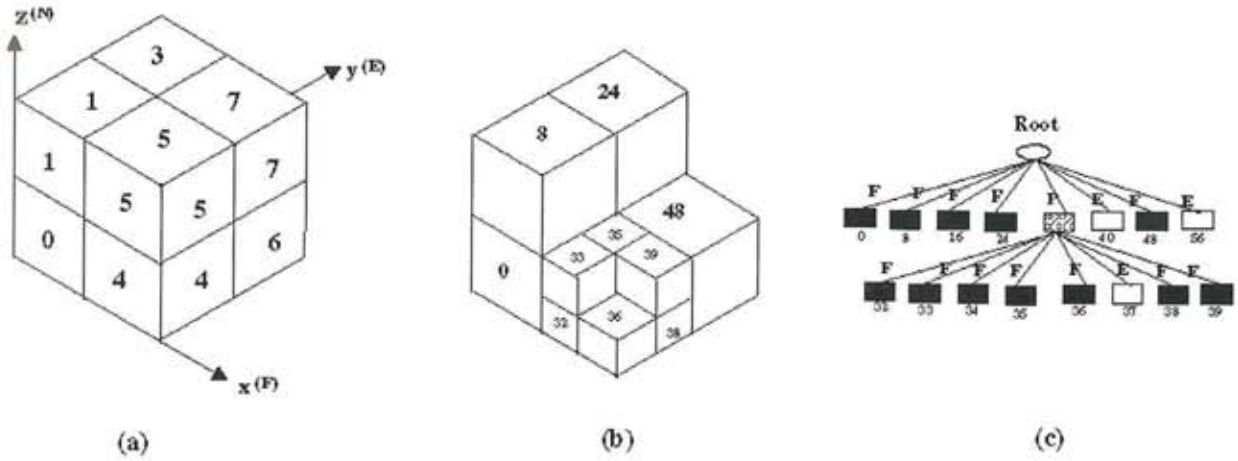

Figure 3. Peano keys for octree encoding and an example.

such as coal, rock, oil and gas layers, 3D boundaries may consist of octants at different levels. It is then natural and efficient to have an algorithm for finding the 3D boundaries based on multi-level octants. The 3D boundaries thus formed contain fewer octants and may result in a decrease of processing time, which is critical in geological applications with large datasets.

Some Definitions

An octant face or its normal vector could point to one of six directions, denoted by

O_d. O_d^L is called a *large neighbour octant* of O if d_face of O is shared by a part of d'_face of O_d^L which has a greater grouping factor and O_1 bigger size.

An octant O is called a *common octant* of O_1 and O_2 if both of them are inside O. The smallest common octant of O_1 and O_2 with a grouping factor k is called the *kth degree common octant*.

Determination of Exact and Large Neighbour Octants

Neighbour octants are searched in each of the six directions. To find neighbour octants of a given octant O in direction d, the first step is to find the grouping factor k of the common octant which contains the octant O and the neighbour octant O_d or O_d^L. Let g be the grouping factor of O, the following iterative formulae can be used to calculate k:

$$\begin{aligned} &P_0(O) = P(O) && \text{(initial value)} \\ &a_i = P_i(O) \bmod(8) && \text{(integer remainder)} \\ &P_{i+1}(O) = P_i(O) / 8 && \text{(integer division)} \\ &i = 1, 2, \ldots, n-1. && (1) \end{aligned}$$

This iterative procedure will be terminated until a_i matches one of the neighbourhood position parameters A for the desired direction d in Table 1. The grouping factor of the common octant is then

$$k = i + g + 1 \qquad (2)$$

Table 1. Neighbourhood position parameters A corresponding to six directions

Direction (d)	A
N	0, 2, 4 or 6
E	0, 1, 4, or 5
B	4, 5, 6 or 7
S	1, 3,5 or 7
W	2, 3, 6 or 7
F	0, 1, 2 or 3

The above grouping factor (implicit information of the size) of the common octant is dependant on a) the position of O and b) the nature of the subdivisions during the octree generation. The common octant is sufficient to contain all possible neighbour octants of O in direction d.

The second step is to find O_d or O_d^L. Since g is smaller than k, O and O_d belong to the kth common octant O_k. The Peano key of O_d can be calculated as

$$P(O_d) = \begin{cases} P(O) + direc \bullet (8^{k-1} - \sum_{i=8}^{k-2} 8^i) & \\ & k-2 \geq g \quad (3) \\ P(O) + direc \bullet 8^{k-1} & otherwise \end{cases}$$

where $direc$ is a function of the direction d: $direc$ (1, 2, 4, -1, -2, -4) for (N, E, B, S, W, F) respectively.

Assume that g' be the grouping factor of the large neighbour octant O_d^L ($g' \leq k$). The Peano key of O_d^L has the following form:

$$P(O_d^L) = P(O_d) - P(O_d) \bmod(8^{g'}) \qquad (4)$$

where $P(O_d)$ is defined in (3).

To demonstrate the above searching scheme, a quadtree is presented in two dimensions in Figure 4, using linear quadtree Peano keys $P(Q)$ for all full quadrants. Equations (1) to (4) are applied to find both exact and large neighbour quadrants in a desired direction. Because the equations are derived for octrees, the number 8 in (1) to (4) has to be changed to 4 in order to make them applicable to quadtrees. Position parameters A in Table 1 have to be restricted to the range of 0, 1, 2, and 3 as well.

Example 1

Figure 4 illustrates a part of a quadtree map. Using the above scheme, the northern neighbour quadrant(s) $P(Q_N)$ or $P(Q_N^L)$ of $P(Q) = 47$ should be found. According to Figure 4, $g = 0$, $P_0(Q) = 47$ and $direc = 1$.

Step 1: determining the grouping factor of the common octant

$i = 0$,
$a_0 = P_0(Q) \bmod(4) = 47 \bmod(4) = 3$,
$a_0 \neq 0$ *or* 2 (position parameters A for N in Table 1)
$P_1(Q) = P_0(Q) / 4 = 47 / 4 = 11$.
$i = 1$

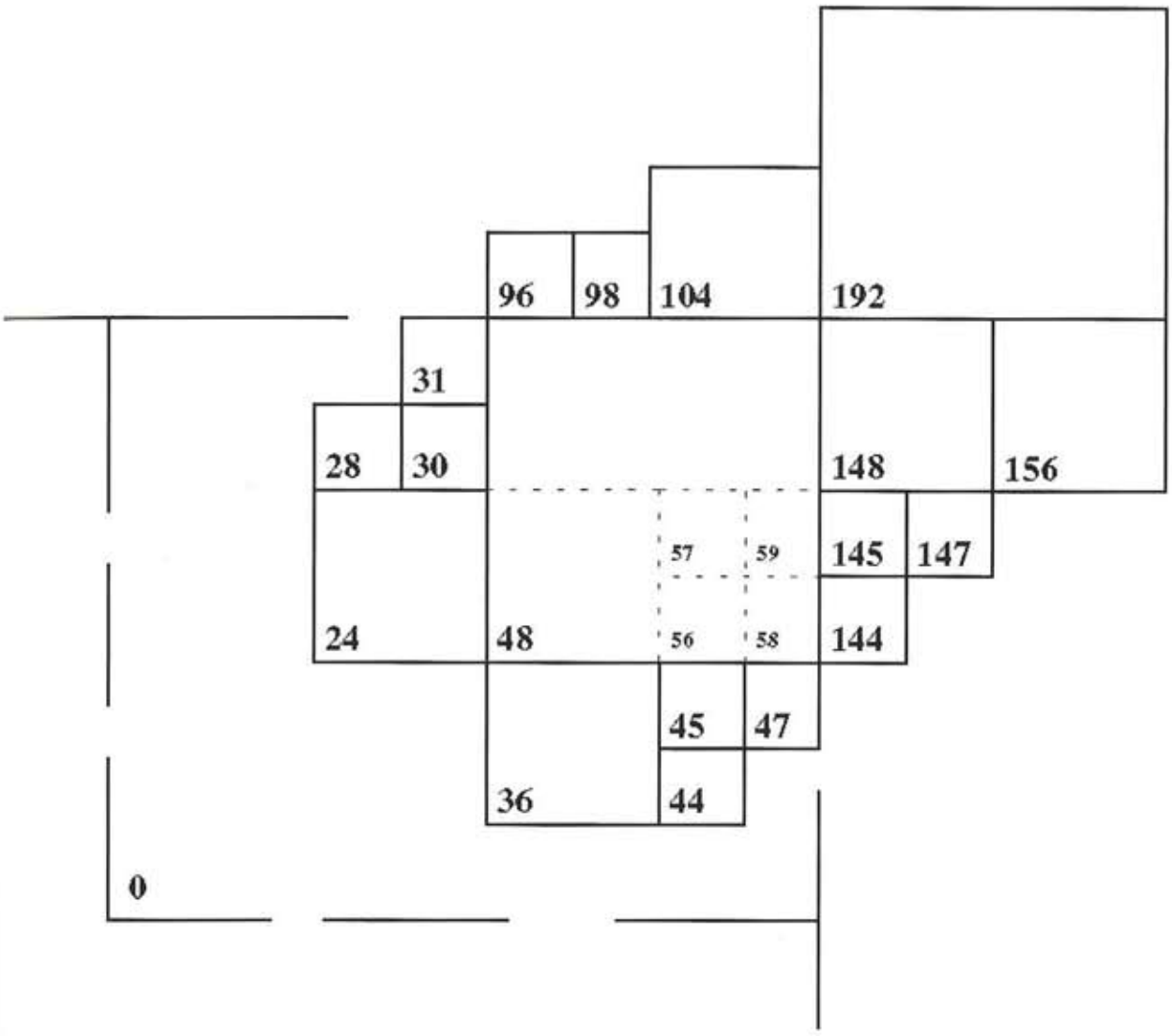

Figure 4. A part of a Peano-encoded quadtree.

$a_1 = P_1(Q) \bmod(4) = 11 \bmod(4) = 3,$

$a_1 \neq 0$ *or* 2,

$P_2(Q) = P_1(Q)/4 = 11/4 = 2.$

$i = 2$

$a_2 = P_2(Q) \bmod(4) = 2 \bmod(4) = 2$

($a_2 = 2$, it is a match and the iterative procedure stops)

$k = 2 + 0 + 1 = 3$

The size of the common quadrant is 8 times ($2^{k-g} = 2^{3-0}$) of the quadrant Q.

Step 2: finding neighbour quadrants

Since $k - 2 = 3 - 2 = 1 \geq g = 0$ the first equation in (3) is used.

$$P(Q_d) = P(Q) + direc \bullet (4^{3-1} - \sum_{i=0}^{1} 4^i)$$

$$= 47 + 1 \bullet (16 - 1 - 4) = 58$$

There is no full quadrant with Peano key 58 in Fig-

ure 4. The large quadrant of Q is to be searched.

$$g' = g + 1 = 1$$

$$P(Q_d^L) = P(Q_d) - P(Q_d) \bmod(4^{g'})$$

$$= 58 - 58 \bmod(4) = 58 - 2 = 56$$

Again, the quadrant with Peano key 56 is not a full quadrant.

$$g' = g + 2 = 2$$

$$P(Q_d^L) = P(Q_d) - P(Q_d) \bmod(4^{g'})$$

$$= 58 - 58 \bmod(16) = 58 - 10 = 48$$

This quadrant ($P(Q_d^L) = 48$) is a full quadrant and the large neighbour quadrant of Q.

A linear octree lists all full octants. If we start from the smallest octants and O_d is found for an octant O in the linear octree, octant O is not on the border in direction d. Otherwise, the large octant has to be searched. In general, large octants O^L exist in three directions (Table 2) depending on the remainder of the Peano key divided by 8 ($P(O)$ mod(8)). Observing the relationship between Peano keys in Figure 3(a) and the definition of directions, it is not hard to understand Table 2. It can be concluded that O is a boundary octant in the checked direction d if neither O_d nor O_d^L is found in the linear octree.

Algorithm for Detecting 3D Boundaries

Each octant of a grouping factor g in the linear octree is initially assigned a border parameter 4^g in each direction ($4^g, 4^g, 4^g, 4^g, 4^g, 4^g$) for (N, E, B, S, W, F). If the border parameter in direction d remains 4^g, it indicates that the octant is not covered by any neighbour octants in this direction; if the parameter is 4^j $(g < j \leq 0)$, the face of the octant in direction d (d_face) has a part as 3D border with an area of 2^{2j}; if the parameter is 0, the octant is covered completely by neighbour octants. Therefore, starting from the initial parameter 4^g, each direction is checked if there is any full octant in the linear octree covering the octant. In cases where a full neighbour octant with a grouping factor j is found, the border parameter in this direction is decreased by 4^j. This procedure is performed for all six directions and all grouping factors. After processing, any octant with border parameters of zero in all directions is considered as interior octant and deleted from the linear octree. The resulted linear octree consists of full boundary octants only. It should be noted that octants on the boundary could be either the smallest octants or octants with higher grouping factors.

Table 2. Possible large neighbour octants in different directions

Peano(O) mod(8)	Directions of possible large neighbor octants
0	SWB
1	NWB
2	SEB
3	NEB
4	SWF
5	NWF
6	SEF
7	NEF

Example 2

Table 3 demonstrates the entire procedure for finding boundary quadrants of the quadtree in Figure 4. The first two columns give the Peano keys and corresponding grouping factors g of all full quadrants of the linear quadtree. Each of quadrants has an initial border parameter 2^g in four directions in the third column.

The searching starts with neighbour quadrants of the lowest level with the grouping factor $j = 0$. If a quadrant is found in a desired direction, the corresponding border parameter in the fourth column is reduced by $2^j = 1$. For example, the quadrant of $P(Q) = 24$ has two neighbour quadrants of $j = 0$ in North. The border parameter in North is reduced from 2 to 0 $(2^1 - 2^0 - 2^0)$.

With $j = 1$ and $j = 2$ in the fifth and sixth column, neighbour quadrants searched become larger. However, the operations on the border parameters remain the same. For instance, at $j = 1$, the quadrant of $P(Q) = 24$ has a large neighbour

($P_E^L(Q) = 48$) in East. The border parameter becomes 0. Please note that if Q_d or Q_d^L is found for Q, border parameters for both Q and Q_d or Q_d^L are updated.

The final status of border parameters are listed in the last column of Table 3. Quadrants with border parameters 0 in all directions are interior quadrants and subject to be deleted. The remaining quadrants with Peano keys and grouping factors describe the border of the quadtree:

$BORDER = \{(24,1), (28,0), (31,0), (36,1), (44,0),$
$(47,0), (96,0), (98,0), (104,1), (144,0),$
$(147,0), (156,1), (192,2)\}$

IV. CONCLUSIONS

3D data structures and associated spatial operations have become important issues as GIS are increasingly applied in extended areas where the third dimension information is critical, such as geological subsurface modeling. In many cases multi-representation-based systems are unavoidable in order to take advantage of different data structures. The border search algorithm introduced in this paper is based on octrees using Peano encoding, which support spatial operations and are complementary to surface models for data visualisation.

The presented algorithm finds neighbour octants of both the same size and different sizes. Without this, large octants would have to be divided into smallest ones at the bottom level in order to search neighbour octants and may not be computationally efficient, especially when partial boundary octants are searched. The general discussions about the neighbourhood relations can also be used for other

Table 3. The entire procedure for searching boundary quadrants of the quadtree in Figure 4

		Initial Value	j = 0	j = 1	j = 2
		Border Parameter 2^g	Border Parameter	Border Parameter	Border Parameter
P_code	Grouping Factor g	N E S W	N E S W	N E S W	N E S W
24	1	2 2 2 2	0 2 2 2	0 0 2 2	0 0 2 2
28	0	1 1 1 1	1 0 0 1	1 0 0 1	1 0 0 1
30	0	1 1 1 1	0 0 0 0	0 0 0 0	Deleted
31	0	1 1 1 1	1 0 0 1	1 0 0 1	1 0 0 1
36	1	2 2 2 2	2 0 2 2	0 0 2 2	0 0 2 2
44	0	1 1 1 1	0 1 1 0	0 1 1 0	0 1 1 0
45	0	1 1 1 1	0 0 0 0	0 0 0 0	Deleted
47	0	1 1 1 1	0 1 1 0	0 1 1 0	0 1 1 0
48	2	4 4 4 4	2 2 2 2	0 0 0 0	Deleted
96	0	1 1 1 1	1 0 0 1	1 0 0 1	1 0 0 1
98	0	1 1 1 1	1 0 0 0	1 0 0 0	1 0 0 0
104	1	2 2 2 2	2 2 2 1	2 0 0 1	2 0 0 1
144	0	1 1 1 1	0 1 1 0	0 1 1 0	0 1 1 0
145	0	1 1 1 1	0 0 0 0	0 0 0 0	Deleted
147	0	1 1 1 1	0 1 1 0	0 1 1 0	0 1 1 0
148	1	2 2 2 2	2 2 0 2	0 0 0 0	Deleted
156	1	2 2 2 2	2 2 2 2	0 2 2 0	0 2 2 0
192	2	4 4 4 4	4 4 4 4	4 4 0 2	4 4 0 2

spatial octree-based spatial operations.

ACKNOWLEDGMENT

Programming efforts made by Mr. L. Qian, Ms. Y. Chen and Mr. F. Dong and Figures drawn by Mrs. J. Tian are appreciated. The authors thank Mr. D.J. Hughes for his constructive discussions and comments. The research is sponsored by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Geological Survey of Canada.

REFERENCES

[1] Arzty, K., G. Frieder and G.T. Herman, 1981. The Theory, Design, Implementation and Evaluation of a Three-dimensional Surface Detection Algorithm, *Computer Graphics and Image Processing*, 15:1-24.

[2] Atkinson, H.H., I. Gargantini and M.V.S. Ramanath, 1985. Improvements to a Recent 3D-border Algorithm, *Pattern Recognition*, 18:215-226.

[3] ESRI 1992. ArcCAD GIS for AutoCAD Provides Full Complement of GIS Tools within AutoCAD Environment. *ARC News Spring* 1992.

[4] Gargantini, I., 1982. Linear Octrees for Fast Processing of Three Dimensional Objects, *Computer Graphics and Image Processing*, 20:365-374.

[5] Harbaugh, J.W. and P.A. Martinez 1990. Two Major Problems in Representing Geological Well and Seismic Data in Petroleum - Bearing Regions via 3-D Geographic Information Systems. in *Three Dimensional Modeling with Geoscientific Information Systems*. Kluwer, Dortrecht, the Netherlands, pp.291-320.

[6] Hughes, J.D. 1994. Interpretive Three-Dimensional Modeling and Geological/economic Analysis of Layered Sequences. *Proceedings of International Association of Mathematical Geologists*, Mt. Tremblant, Quebec, October, 1994.

[7] Ibaroudene, D., V. Demjanenko and R.S. Acharya, 1990. Adjacency Algorithms for Linear Octree Nodes, *Computer Vision, Graphics and Image Processing*, 8:115-123.

[8] Jones, C.B., 1989. Data Structures for Three-dimensional Spatial Information Systems in Geology, *International Journal of Geographical Information Systems*, 3(1):15-31.

[9] Laurini, R. and D. Thompson, 1992. *Fundamentals of Spatial Information Systems*. Academic Press, pp.507-511.

[10] Li, R., 1994. Data Structures and Application Issues in 3-D Geographic Information Systems, *Geomatica*, 48(3):209-224.

[11] Liu, H.K., 1977. Two- and Three-dimensional Boundary Detection, *Computer Graphics and Image Processing*, pp.123-134.

[12] Maguire, D.J., M.F. Goodchild and D.W. Rhind 1991. *Geographical Information Systems: Principles and Applications*. Longman Scientific & Technical, England.

[13] Meagher, D., 1982. Geometric Modeling Using Octree Encoding, *Computer Graphics and Image Processing*, 19:129-147.

[14] Meier, A., 1986. Applying Relational Database Techniques to Solid Modeling. *CAD*,18(6):319-324.

[15] Raper, J.F. and B.Kelk 1991. Three Dimensional GIS. In *Geographical Information Systems: Principles and Applications*, edited by Maguire et al., Longman Scientific & Technical, England, pp.299-317.

[16] Samet, H. 1990. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 493p.

[17] Silicon Graphics, Inc. 1992. *Graphics Library Programming Guide*. SGI, Mountain View, CA.

[18] Tamminen, M. and H. Samet, 1984. Efficient Octree Conversion by Connectivity Labeling. *Computer Graphics*, 18(3):312-318.

[19] Turner, A.K. (ed.) 1990. *Three Dimensional Modeling with Geoscientific Information Systems*. Kluwer, Dortrecht, the Netherlands.