Highly Efficient Computer Oriented Octree Data Structure and Neighbors Search in 3D GIS Spatial

Noraidah Keling[1], Izham Mohamad Yusoff[1], Uznir Ujang[2], Habibah Lateh[1]

[1] *School of Distance Education, Universiti Sains Malaysia*
[2] *Faculty of Geoinformation & Real Estate, Universiti Teknologi Malaysia*

Corresponding Author: Noraidah Keling
E-mail: noraidahkeling@gmail.com

Abstract:

Three Dimensional (3D) have given new perspective in various field such as urban planning, hydrology, infrastructure modeling, geology etc due to its capability of handling real world object in more realistic manners, rather than two-dimensional (2D) approach. However, implementation of 3D spatial analysis in the real world has proven difficult due to the complexity of algorithm, computational power and time consuming. Existing GIS system enables 2D and two-and-a-half dimensional (2.5D) spatial datasets, but less capable of supporting 3D data structures. Recent development in Octree see more effort to improve weakness of octree in finding neighbor node by using various address encoding scheme with specific rule to eliminate the need of tree traversal. This paper proposed a new method to speed up neighbor searching and eliminating the needs of complex operation to extract spatial information from octree by preserving 3D spatial information directly from Octree data structure. This new method able to achieve O(1) complexity and utilizing Bit Manipulation Instruction 2 (BMI2) to speedup address encoding, extraction and voxel search 700% compared with generic implementation.

1 Introduction

Recent developments in the field of three-dimensional (3D) data spatial representation have led to a renewed interest in Geographic Information System (GIS). Application of 3D technologies in GIS developments is one of the issue that have special attention in this platform and has come to the limelight among various geo-related profesionals such as land practitioners and environmentalists [1][2]. Perspective in third dimension really helpful in some field likes urban planning and management [3], geological sub surface modelling [4], mining and oil exploration [5][6], hydrology [7], infrastructure modeling, geology etc [8][9]. Apart from visualization task, 3D have given new perspective in this field due to its capability of handling complex real world phenomena in more realistic manners to recognize the geographical and geological space better and master the inherent laws in earth-science and engineering application to provide facilities to analyze and thus, help decision makers explore and understand the complexities'[10]. Most of the direct and indirect approaches use GIS for effectively by incorporating GIS-functions such as a spatial database for storing, displaying, and updating the input data based on site-specific measurement and experiment to understand the role of the interaction conditioning factor [11].

However, a major problem with this kind of application is existing commercial GIS software enables 2D and two-and-a-half dimensional (2.5D) spatial datasets which is a very useful to visualize adjacent surfaces with variations of height such as terrain but the major limitation is that height values are represented as a function of x and y coordinates and less capable of supporting 3D data structures because they use a 2.5D approach for their 3D functionality; only focusing on 3D visualization [12]. The usage of 2.5D for 3D data representation is just approximation and no more satisfy the recent need [13] [14]. Therefore, the demand on a fully functional 3D GIS is increasing and became more and more important to geological analysis and engineering application to understand geological phenomena and complex structure any kind of information related with their 3D environment in a more realistic way and to carry out the complicated 3D spatial analysis [15].

Representation of numerous real-world phenomena such as flood, geology, water pollution, air pollution in spatial data model are not same [16]. In term of geometric characteristics, existing 3D spatial representation data model in Geographical Information System (GIS) can be described by three representation which are surface, volumetric, and hybrid (mixed of both model; surface and volumetric model [4][14]. Table 1 show various data model representation [12] [17] [18][19] [20] [21] [22].
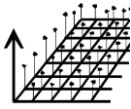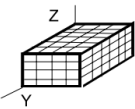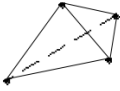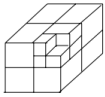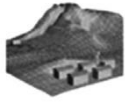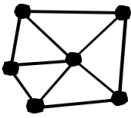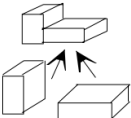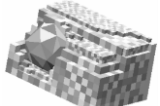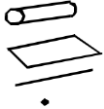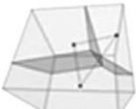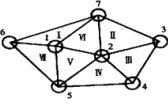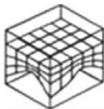
Surface Model	Volumetric Model		Hybrid Model
	Regular	Irregular	
Grid	3D array	TEN	TIN-Octree
Shape	Octree	Pyramid	TIN-CSG
Facet	CSG	TP	Octree-TEN
B-Rep	Voxel	3D Voronoi	
TIN		Geocellular	

Table 1. Various Data Model Representation

2   Methods

2.1   Related Research

Octree is a method where 3D space had been represented in a hierarchical tree structure as visualized in Figure 1. Classical octree originate from Klinger [23] as quadtree and subsequently expanded to cover 3D space as Octree. The earlier implementation of octree use tree traversal like suggested by Samet [24], Ballard and Brown [25] and Besancon [26]. Recent research development in Octree see more effort to improve weakness of octree in finding neighbor node which is required for spatial analysis by using various address encoding scheme with specific rule like matrix [27], lookup table [28] and arithmetic [29] to eliminate the need of tree traversal.
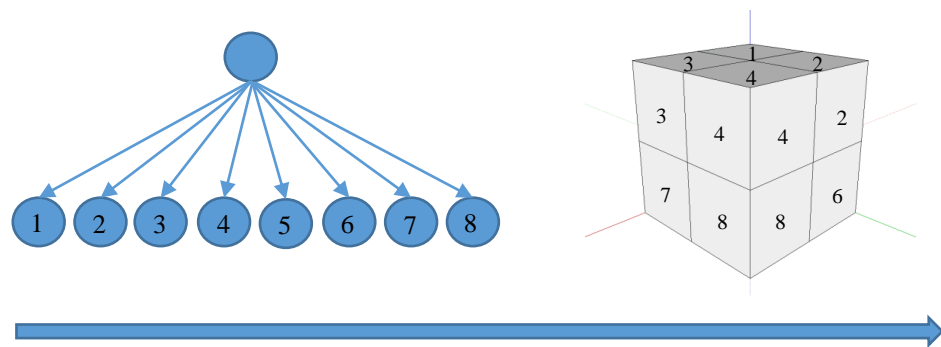
Figure 1 : Octree Node to 3D Space Mapping

2.2   Proposed Octree Structure

In this paper, the author proposing another address encoding scheme and data structure which bases on binary ordering which efficiently preserve 3D spatial information from Octree structure and carefully designed in such way to fully take advantage of the advancement of computer CPU accelerator instruction to speed up neighbor search and eliminating the need of complex operation to derive voxel spatial information unlike suggested by Kim and Lee [29] or the matrix solution like Voros [27].

2.2.1 Binary Address Encoding

The basis of the proposed octree encoding scheme is to assign each node of Octree a unique address in binary form with 3D spatial information embedded. Octree in 3D space is binary by nature. Voxel progression along with the axis can be represent by only 1 bit as shown in Figure 2. For example, the voxel which is close to axis X, the address is set as 0 and for the voxel located further from the axis the address is set as 1. By incorporating this nature into address encoding, we can preserve all 8 v

o

x

eX

l

s

patial data information by using only 3 bits.
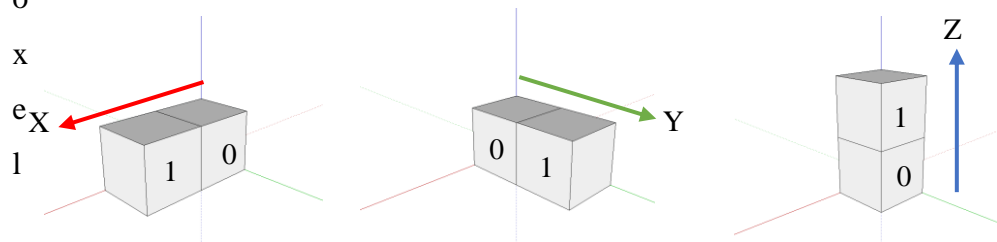
Figure 2 : Voxel Numbering Progression

These 3 bits can be arrange using $X_pY_pZ_p$ standard, where P is octree level or depth. For example, for the voxel which located in coordinate x=0, y=0. z=0, the voxel address is 0b000 (0d0) and voxel located at coordinate x=1,y=0,z=1 then the voxel address will be 0b111 (0d7). The addressing for the voxel is straight forward and easy to identify voxel physical location just based on octree address. Figure 3 show the complete address for proposed octree in binary form.
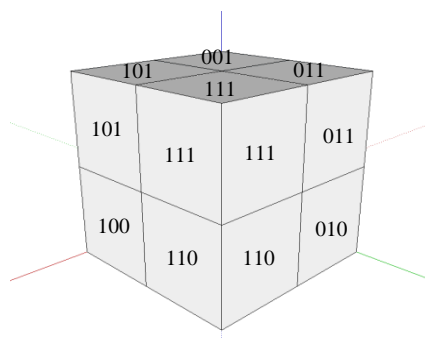
Figure 3 : Proposed Octree Addressing Scheme

2.2.2   Subdivision Address Encoding

Voxel Address $= (X_p Y_p Z_p) \cap (X_{p+1} Y_{p+1} Z_{p+1}) \cap \dots \cap (X_{p+n} Y_{p+n} Z_{p+n})$

(1)

The address encoding for subdivision octree is similar with parent voxel
except the parent address is directly concatenating into child voxel address
as shown in   (1). The main different between proposed encoding with
previous methods that introduced by Voros [27], Payeur [28] and Kim &
Lee [29] is this method stricly using 3 bits binary encoding per level as
shown in Figure 4 and Figure 5 while others are using decimal based as
shown in Figure 6 and Figure 7. Decimal representation is make sense for
human readable and manual processing because parent address can be
easily identified concatenated with child address, but to represent those
number in decimal will require 25% more space and without any
advantage for computer processing.
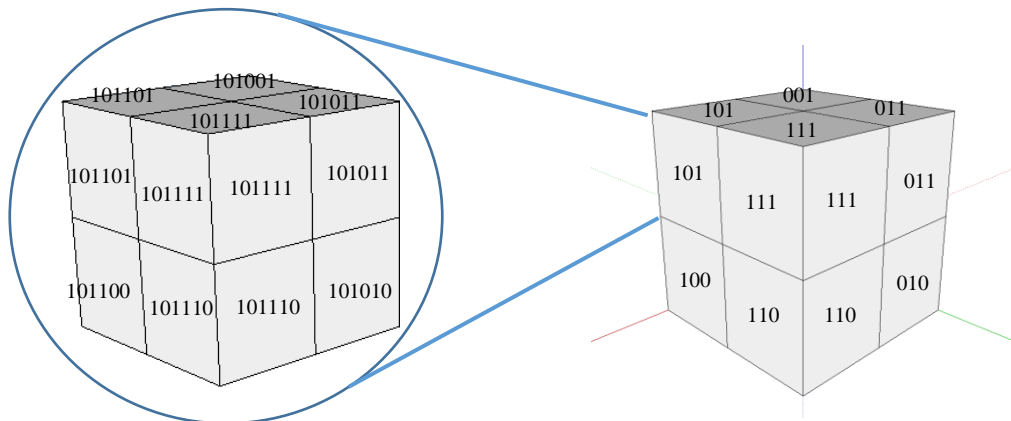
Figure 4 : Proposed Octree Subdivision Addressing Scheme (binary)
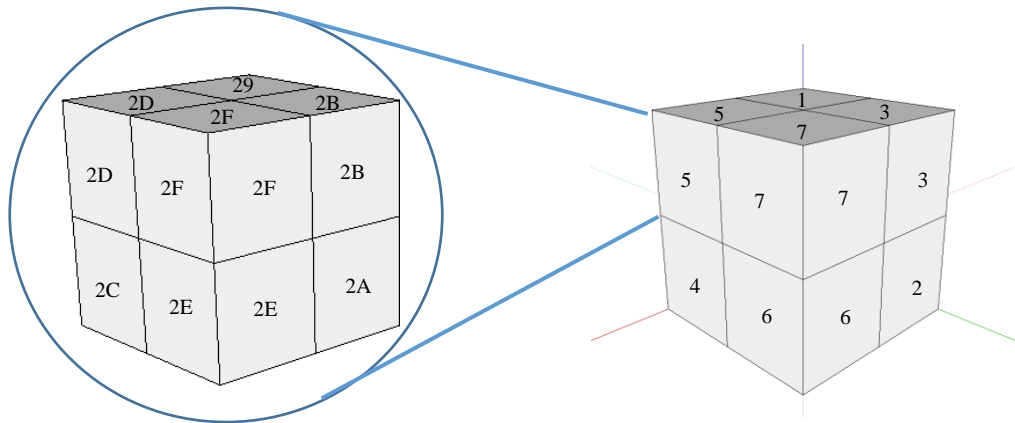
Figure 5 : Proposed Octree Subdivision Addressing Scheme (hexadecimal)
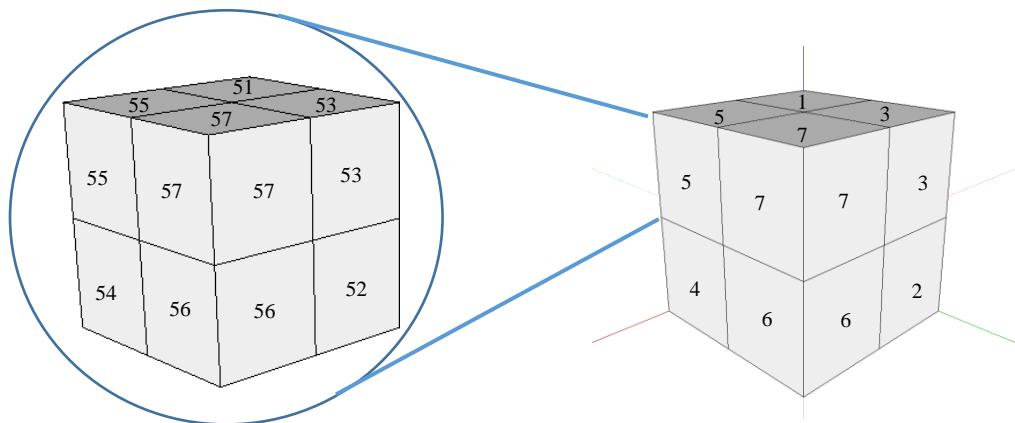
Figure 6 : Payeur and Voros Octree Subdivision Addressing Scheme
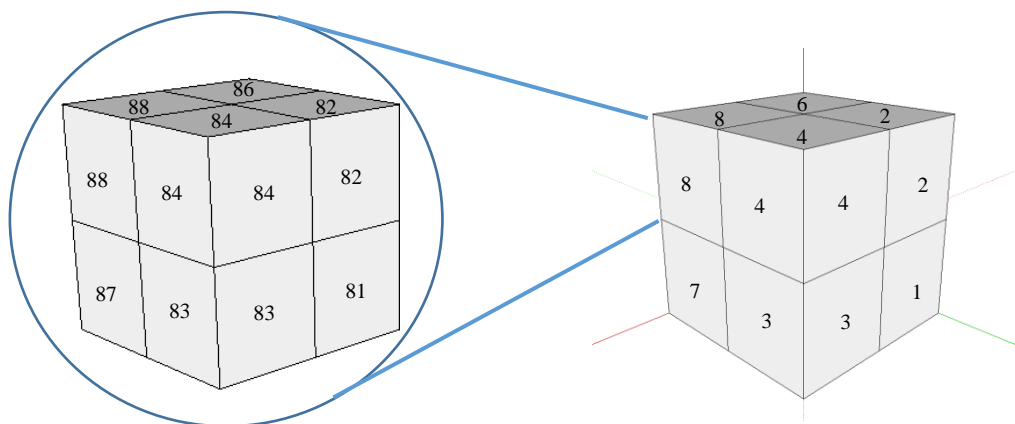
Figure 7 : Kim and Lee Octree Subdivision Addressing Scheme

2.3 Data Structure

In order to take advantage of modern computer (explained in section 2.4), each voxel should be kept either in 32 bits or 64 bits. By using equation (2), 32 bits data structure able to hold maximum of 10 levels of octree and for 64 bits data structure will to be able to hold up to 21 levels of octree.

$$Maximum\ Octree\ Maximum\ level\ (P)\ = \lfloor\frac{Data\ Stucture\ Bits}{3}\rfloor$$

(2)

Using equation (3), with the assumption of minimum resolution of surface area that we want to cover is 1 m^2, 32bits data structure only able to cover up to 1,048.576 km^2 but 64 bits data structure able to cover up to 4,398,046,511.104km^2. With the total earth surface area is estimated only around 510,072,000 km^2, 64 bits data structure practically more than enough to cover whole earth thus there is no need more than 64 bits data structure.

$$Covered\ Area = \ 2^{P*2} * Resolution$$

(3)

For this paper, we focus on 64 bits data structure but the same method also can be applied for 32 bits data structure. The octree node address $X_p Y_p Z_p$ for each level should be start concatenated from MSB to LSB with 0b1 appended after last 3 bits address as terminator as shown in Figure 8 and Figure 9.

MSB										LSB		
X_1	Y_1	Z_1	X_2	Y_2	Z_2	X_3	Y_3	Z_3	...	X_{21}	Y_{21}	Z_{21}	1
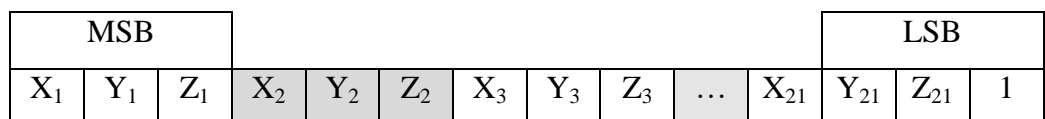
Figure 8 : Proposed Data Structure

MSB										LSB		
X_1	Y_1	Z_1	X_2	Y_2	Z_2	1	0	0	...	0	0	0	0
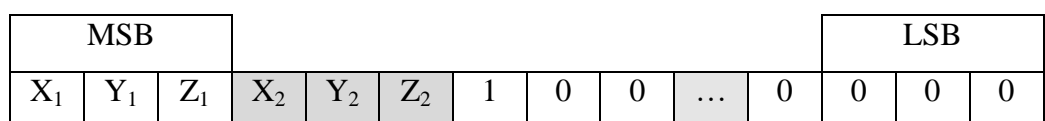
Figure 9 : Proposed Data Structure with Partially Fill Up Bits

Numbers of octree level can be determine by (4) and since each level require
3 bits of information embedded into the data structure, the total of bits need
to be used is calculated according to (5).

$$\text{Numbers of Octree level (P)} = \frac{\log_2\left(\left(\frac{Area\ Size\ to\ Cover}{\text{Smallest unit@resolution}}\right)\right)}{2}$$

(4)

$$Bits\ Required = 3P + 1$$

(5)

2.4 CPU Accelerated Instruction

Bit Manipulation Instruction 2 (BMI2) is series of new instructions
introduced by Intel as part of Advance Vector Extensions 2 (AVX 2) in 2013
[30]. The same instruction and also supported by AMD in Excavator 2015.
Intel and AMD is market leader in computing with 80% of the total computer
sold is based on Intel or AMD CPU. There are 3 instructions from BMI2 that
can be used to accelerate octree address generation and manipulation;
TZCNT, PDEP and PEXT.

2.4.1 Count the Number of Trailing Zero Bits (TZCNT)

TZCNT is a single instruction to count trailing 0 in the registers. This
instruction eliminating the need of bit extraction-compare loops as shown
in Figure 10. TZCNT can be used to identify octree level from voxel
address as demonstrate in section 2.5.1

```
temp ← 0
DEST ← 0
DO WHILE ( (temp < OperandSize) and (SRC[ temp] = 0) )
      temp ← temp +1
      DEST ← DEST+ 1
OD
IF DEST = OperandSize
```

```
        CF ← 1
ELSE
        CF ← 0
FI
IF DEST = 0
        ZF ← 1
ELSE
        ZF ← 0
FI
```

Figure 10 : TZCNT Pseudo Code

2.4.2   Parallel bits Extract (PEXT)

PEXT is a single instruction to insert bits into different position from a
sequences as shown in Figure 11. PEXT use in proposed method to
reconstruct voxel address as demonstrate in section 2.5.2 and 2.5.4.

```
TEMP ← SRC1;
MASK ← SRC2;
DEST ← 0 ;
m← 0, k← 0;
DO WHILE m< OperandSize
     IF MASK[ m] = 1 THEN
            DEST[ k] ← TEMP[ m];
                 k ← k+ 1;
     FI
m ← m+ 1;
OD
```

Figure 11 : PEXT Pseudo Code

2.4.3   Instructions, Parallel Bits Deposit (PDEP)

PDEP is a single instruction to gather bits from different position into a
sequences as shown in Figure 12. PDEP use in proposed method to extract
X Y and Z 3D spatial coordinate from voxel address as demonstrate in
section 0 and 2.5.4

```
TEMP ← SRC1;
MASK ← SRC2;
```

```
DEST ← 0 ;
m← 0, k← 0;
DO WHILE m< OperandSize
      IF MASK[ m] = 1 THEN
      DEST[ m] ← TEMP[ k];
            k ← k+ 1;
      FI
      m ← m+ 1;
OD
```

Figure 12 : PDEP Pseudo Code

2.5 Manipulation

2.5.1 Determine Octree Level from Voxel Address

Using proposed voxel address encoding and data structures, we can easily
determine octree level from voxel address alone without requiring whole
octree traversal. Here TZCNT can help to determine location of
termination bit as shown in Figure 13.
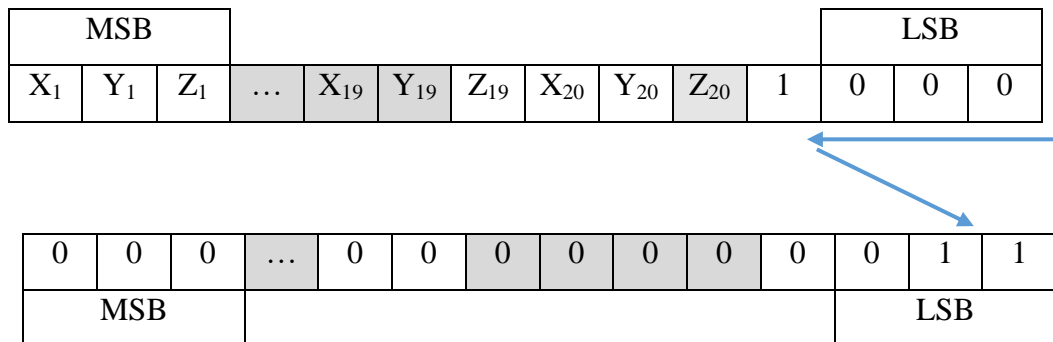
Figure 13 : Counting Trailing Zeros

After getting termination bit location, octree level can be determine by using (6)

$$\text{Voxel Octree Level} = 21 - \frac{\text{TZCNT}((\text{Voxel Address Register}))}{3}$$

(6)

2.5.2 Voxel Address to 3D Spatial Coordinate

Our proposed method make the process to determine 3D spatial coordinate
from voxel address simple and fast. With PEXT instruction, the extraction
much faster. The coordinate can be easily extracted using (7)

$$Coordinate = (PEXT(X\ mask), PEXT(Y\ mask), PEXT(Z\ mask))$$

(7)

Figure 14 visualize how X axis coordinate can be extracted from voxel
address. The same process also true for Y and Z axis as shown in Figure
15 and Figure 16.

MSB										LSB		
X_0	Y_0	Z_0	X_1	Y_1	Z_1	X_2	Y_2	Z_2	...	X_p	Y_p	Z_p

1	0	0	1	0	0	1	0	0	...	0	0	0

0	0	0	...	0	0	0	0	0	0	X_0	X_1	X_2
MSB										LSB		

Figure 14 : Extracting Voxel X Coordinate

MSB										LSB		
X_0	Y_0	Z_0	X_1	Y_1	Z_1	X_2	Y_2	Z_2	...	X_p	Y_p	Z_p

0	1	0	0	1	0	0	1	0	...	0	0	0

0	0	0	...	0	0	0	0	0	0	Y_0	Y_1	Y_2
MSB										LSB		
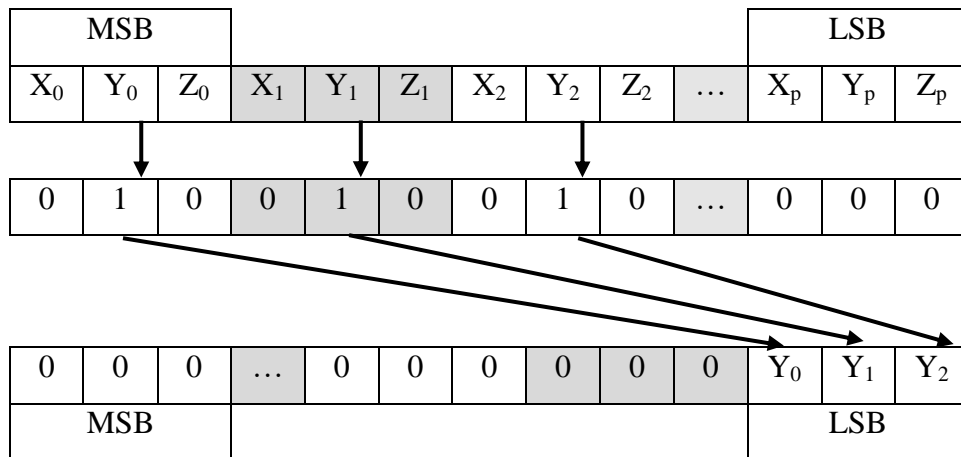
Figure 15 : Extracting Voxel Y Coordinate

MSB										LSB		
X_0	Y_0	Z_0	X_1	Y_1	Z_1	X_2	Y_2	Z_2	...	X_p	Y_p	Z_p

0	0	1	0	0	1	0	0	1	...	0	0	0

0	0	0	...	0	0	0	0	0	0	Z_0	Z_1	Z_2
MSB										LSB		
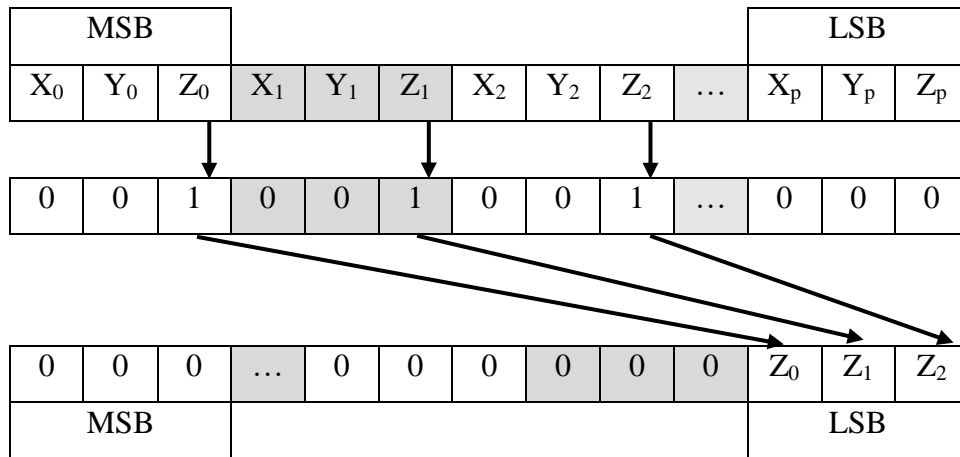
Figure 16 : Extracting Voxel Z Coordinate

2.5.3 3D Spatial Coordinate to Voxel Address

For application that need voxel address generation for certain voxel coordinate like voxel neighbor discovery, PDEP can be used efficiently to reconstruct voxel address as shown in (8).

$$Voxel\ address = PDEP(X\ mask) \cup PDEP\ (Y\ mask) \cup PDEP(Z\ mask)$$

(8)

Figure 17 visualize how X axis coordinate can be converted into voxel address. The same process also true for Y and Z axis as shown in Figure 18 and Figure 19.
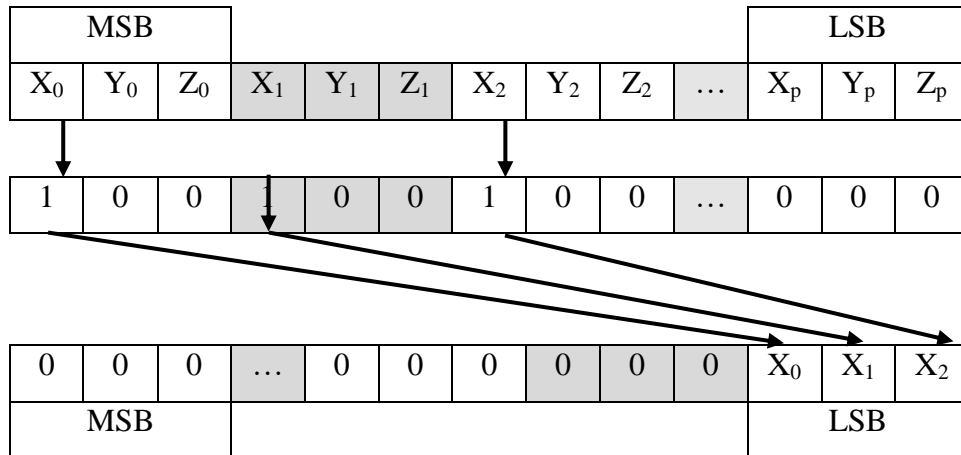
MSB										LSB		
0	0	0	...	0	0	0	0	0	0	X_0	X_1	X_2

1	0	0	1	0	0	1	0	0	...	0	0	0

X_0	0	0	X_1	0	0	X_2	0	0	...	0	0	0
MSB										LSB		
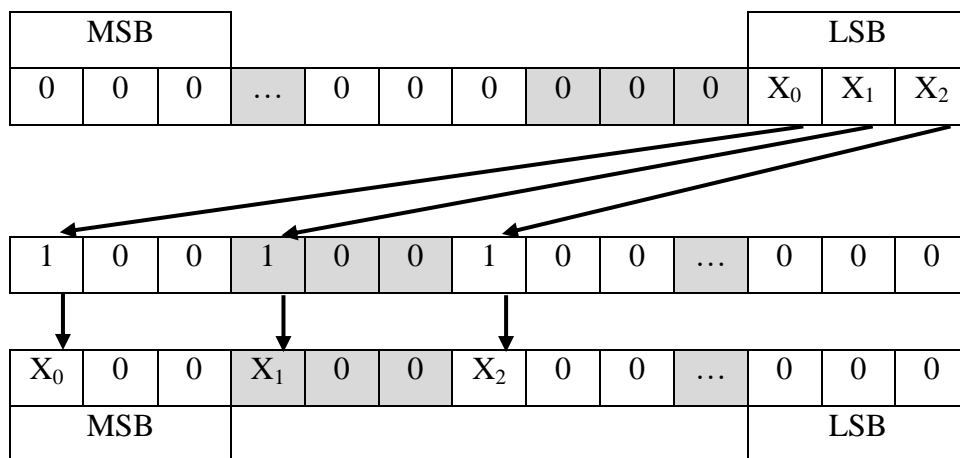
Figure 17 : Voxel Address Generation for X Axis

MSB										LSB		
0	0	0	…	0	0	0	0	0	0	Y_0	Y_1	Y_2

0	1	0	0	1	0	0	1	0	…	0	0	0

X_0	Y_0	0	X_1	Y_1	0	X_2	Y_2	0	…	0	0	0
MSB										LSB		
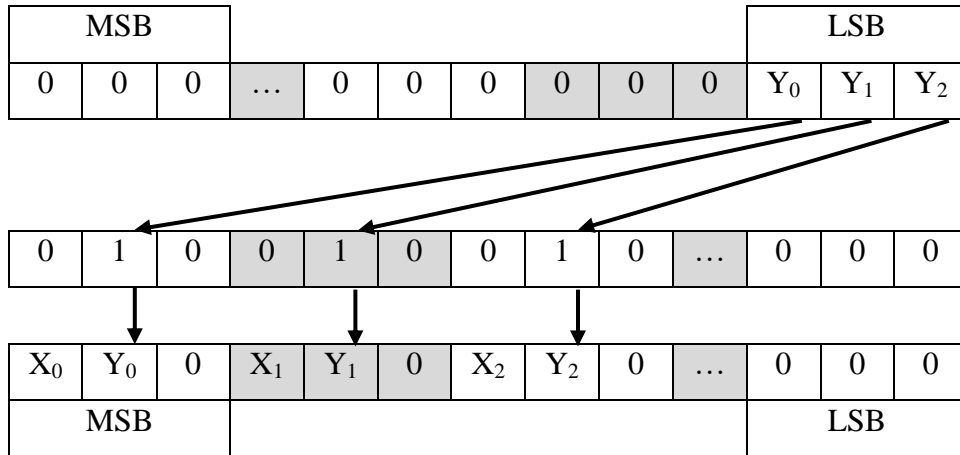
Figure 18 : Voxel Address Generation for Y Axis

MSB										LSB		
0	0	0	…	0	0	0	0	0	0	Z_0	Z_1	Z_2

0	0	1	0	0	1	0	0	1	…	0	0	0

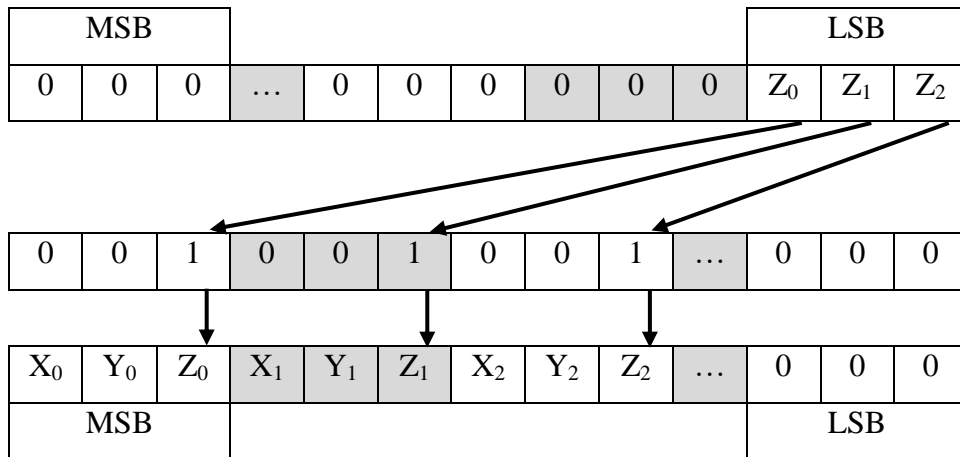| X_0	Y_0	Z_0	X_1	Y_1	Z_1	X_2	Y_2	Z_2	…	0	0	0
MSB										LSB		

Figure 19 : Voxel Address Generation for Z Axis
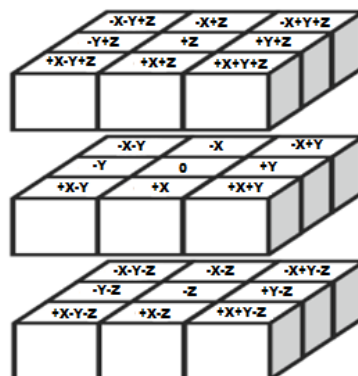
2.5.4   Finding Neighbor Voxel

Figure 20 : 26 Neighbors Voxels

Neighbor search is commonly used for spatial navigation and analysis. Finding neighbor search in classical Octree very resource extensive since it require full node traverse to find each neighbor so in order to practically

use octree in those situation, address encoding like by proposed Voros [27], Payeur [28] and Kim & Lee [29] is favored.

There are total of 26 neighbor voxels in an Octree and the offset of XYZ coordinate is shown in Figure 20. Our proposed method able to find each of them with 3 simple generic step without complicated math or lookup table. For example, to get neighbor address at offset +X+Y+Z:

1) Extract voxel reference coordinate
   X_{ref}= PEXT (X mask, $Voxel_{ref}$)
   Y_{ref}= PEXT (Y mask, $Voxel_{ref}$)
   Z_{ref}= PEXT (Z mask, $Voxel_{ref}$)

2) Add neighbor offset
   X_{+X+Y+Z} = Xref + 1
   Y_{+X+Y+Z} = Yref + 1
   Z_{+X+Y+Z} = Zref + 1

3) Reconstruct $Voxel_{+X+Y+Z}$ address
 $Voxel_{+X+Y+Z}$ = PDEP(X Mask, X_{+X+Y+Z}) ∪ PDEP(X Mask, Y_{+X+Y+Z}) ∪ PDEP(X Mask, Z_{+X+Y+Z})

Our proposed method also having fixed complexity regardless octree level so it suit for simple and huge octree structure.

3   Results

Classical octree is inefficient to find neighbor address and having complexity $O(2^{3(p-1)})$ where p represents the maximum number of resolution levels in octree. Payuer's method [31] have maximum complexity of $O(26(p-1))$ if there is no direction information given. Kim and Lee [29] method have worst case complexity $O(6(p-1))$. Gargantini [32] and Schrack [33] both have $O(p)$ complexity. Our proposed method is independent of octree level and have fixed $O(1)$ complexity. Figure 21 show comparison of the complexity

neighbor search for those alternative implementation and our proposed
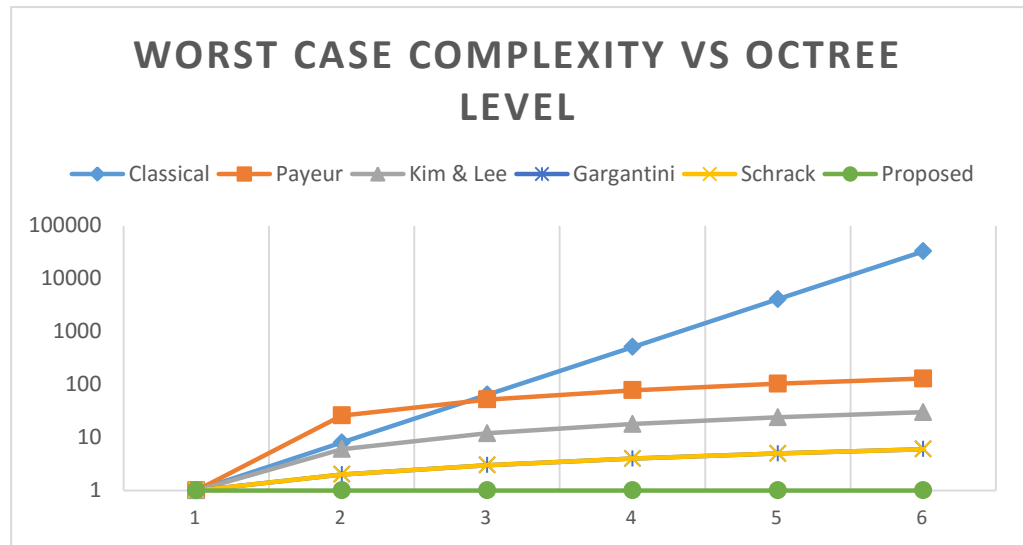
method.

Figure 21: Worst Case Complexity of Neighbor Search

Another advantage to for having data structure and encoding that computer
friendly is the ability to speedup address encoding, extraction and neighbor
search with BMI2 instruction which is very fast and simpler to implement
compared with generic implementation. Figure 22 show comparison of
performance between generic implementation and BMI2 accelerated
instruction. By average, BMI2 implementation 700% faster than generic
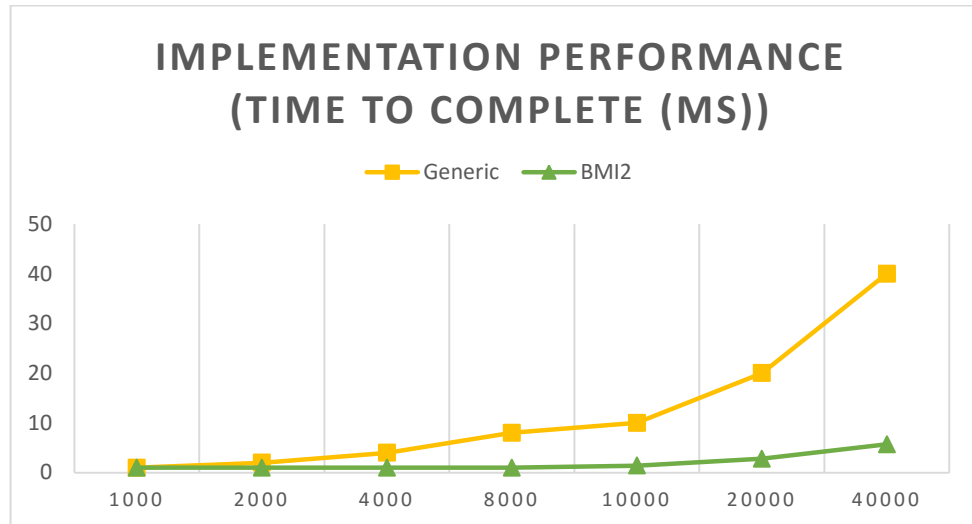implementation.

Figure 22 : Octree Neighbor Finding Performance Improvement with BMI2

4 Discussion

The key for faster and simple octree address encoding scheme is to use binary form. Octree in 3D space is binary compatible and by using clever rule, 3D spatial information easily embed inside the address itself and can be constructed, extracted and manipulated without using complex post processing. We also proposing new 64 bits data structure which is designed with BMI2 compatible in mind and proposing simple and fast method for neighbor search which is having constant complexity regardless octree level. This method can be expanded in the future to cover the neighbor search for intra octree level.

References

[1]   M. F. Goodchild, "Geographic information systems and science: today and tomorrow," *Annals of GIS*, vol. 15. pp. 3–9, 2009.

[2]   B. J. L. Berry, D. A. Griffith, and M. R. Tiefelsdorf, "From Spatial Analysis to Geospatial Science," vol. 40, pp. 229–238, 2008.

[3]   U. Ujang, A. A. Rahman, and F. Anton, "An Approach of Instigating 3D City Model s i n Urban Air Pollution Modeling for Sustainable Urban Development in Malaysia An Approach of Instigating 3D City Model s i n Urban Air Pollution Modeling f or Sustainability Urban Development in Malaysia," no. June 2014, pp. 1–22.

[4]   L. Q. R. Li, Y. Chen, F. Dong, "3D Data Structures and Applications in Geolofical Subsurface Modeling," in *International Archives of Photogrammetrry and Remote Sensing. Vol. XXXI, Part B4. Vienna*, 1996, pp. 508–513.

[5]   J. Pouliot, K. Bédard, D. Kirkwood, and B. Lachance, "Reasoning about geological space: Coupling 3D GeoModels and topological queries as an aid to spatial data selection," *Comput. Geosci.*, vol. 34, pp. 529–541, 2008.

[6]   B. Jin, Y. Fang, and W. Song, "3D visualization model and key techniques for digital mine," *Trans. Nonferrous Met. Soc. China*, vol. 21, pp. s748–s752, 2011.

[7]   M. Y. Izham, U. Muhamad Uznir, A. R. Alias, K. Ayob, and I. Wan Ruslan, "Influence of georeference for saturated excess overland flow modelling using 3D volumetric soft geo-objects," *Comput. Geosci.*, vol. 37, no. 4, pp. 598–609, Apr. 2011.

[8]   L. F. Zhu, M. J. Li, C. L. Li, J. G. Shang, G. L. Chen, B. Zhang, and X. F. Wang, "Coupled modeling between geological structure fields and property parameter fields in 3D engineering geological space," *Eng. Geol.*, vol. 167, pp. 105–116, 2013.

[9]   L. Zhu, C. Zhang, M. Li, X. Pan, and J. Sun, "Building 3D solid models of sedimentary stratigraphic systems from borehole data: An automatic method and case studies," *Eng. Geol.*, vol. 127, pp. 1–13, 2012.

[10]   H. Tianding, "3D GIS interactive editing method: Research and application in glaciology," *… Sci. Eng. (ICISE), 2010 2nd …*, pp. 1–4, 2010.

[11]   R. a de By, "Principles of Geographic Information Systems---An introductory textbook," vol. 1, 100AD.

[12]   A. Abdul-Rahman and M. Pilouk, *Spatial Data Modelling for 3D GIS*. Springer, 2008.

[13]   J. D. Rogers and R. Luna, "IMPACT OF GEOGRAPHICAL INFORMATION SYSTEMS ON GEOTECHNICAL ENGINEERING," pp. 1–23, 2004.

[14]   W. Lixin, "Topological relations embodied in a generalized tri-prism (GTP) model for a 3D geoscience modeling system," *Comput. Geosci.*, vol. 30, pp. 405–418, 2004.

[15]   S. H. I. Wenzhong, "DEVELOPMENT OF A HY BRID MODEL FOR THREE-DIMENSIONAL GIS," *Geo-Spatial Inf. Sci.*, vol. 3, pp. 6–12, 2000.

[16]   M. Breunig and S. Zlatanova, "3D geo-database research: Retrospective and future directions," *Comput. Geosci.*, vol. 37, no. 7, pp. 791–803, Jul. 2011.

[17]   D. Y. Shen, A. N. Ma, H. Lin, X. H. Nie, S. J. Mao, B. Zhang, and J. J. Shi, "A new approach for simulating water erosion on hillslopes," *International Journal of Remote Sensing*, vol. 24. pp. 2819–2835, 2003.

[18]   D. Y. Shen, K. Takara, Y. Tachikawa, and Y. L. Liu, "3D simulation of soft geo-objects," *Int. J. Geogr. Inf. Sci.*, vol. 20, no. 3, pp. 261–271, Mar. 2006.

[19]   D. Shen and K. Takara, "A modelling and 3-D simulation system for water erosion on hillslopes—M3DSSWEH," *J. Hydraul. Res.*, vol. 44, no. 5, pp. 674–681, Sep. 2006.

[20]   W. U. Huixin and X. U. E. Huifeng, "A New Hybrid Data Structure for 3D GIS," *First Int. Conf. Innov. Comput. Inf. Control - Vol. I*, vol. 1, pp. 162–166, 2006.

[21]   S. Wenzhong, "Development of a hybrid model for three-dimensional GIS," *Geo-Spatial Inf. Sci.*, vol. 3, pp. 6–12, 2000.

[22]   D. Shen, D. W. Wong, F. Camelli, and Y. Liu, "An ArcScene plug-in for volumetric data conversion, modeling and spatial analysis," *Comput. Geosci.*, vol. 61, pp. 104–115, 2013.

[23]   A. Klinger, "Patterns and search statistics," *Optim. methods Stat.*, 1971.

[24]   H. Samet, "Neighbor finding in images represented by octrees," *Comput. Vision, Graph. Image Process.*, vol. 45, p. 400, 1989.

[25]   D. Ballard and C. Brown, "Computer vision, 1982," *Prenice-Hall, Englewood Cliffs, NJ*.

[26]   J. Besançon and O. Faugeras, "Vision par ordinateur en deux et trois dimensions," 1988.

[27]   J. Vörös, "Strategy for repetitive neighbor finding in octree representations," *Image Vis. Comput.*, vol. 18, pp. 1085–1091, 2000.

[28]   P. Payeur, "A computational technique for free space localization in 3-D multiresolution probabilistic environment models," *IEEE Trans. Instrum. Meas.*, vol. 55, no. 5, pp. 1734–1746, 2006.

[29]   J. Kim and S. Lee, "Fast neighbor cells finding method for multiple octree representation," *2009 IEEE Int. Symp. Comput. Intell. Robot. Autom. -*, pp. 540–545, Dec. 2009.

[30]   I. Intel, "Advanced vector extensions programming reference," *Intel Corp.*, 2011.

[31]   P. Payeur, "An optimized computational technique for free space localization in 3-D virtual representations of complex environments," *2004 IEEE Symp. Virtual Environ. Human-Computer Interfaces Meas. Syst. 2004. (VCIMS).*, no. AUGUST 2004, pp. 1–7, 2004.

[32]   I. Gargantini, "Linear octtrees for fast processing of three-dimensional objects," *Comput. Graph. Image Process.*, 1982.

[33]   G. Schrack, "Finding neighbors of equal size in linear quadtrees and octrees in constant time," *CVGIP Image Underst.*, 1992.