Proposing Storage Structures and Interpolation Algorithms of 3D Spatial Data

Dang Van Pham[1,2(✉)] and Vinh Cong Phan[1]

[1] Faculty of Information Technology, Nguyen Tat Thanh University, Hochiminh City, Vietnam
pvdang.tps@gmail.com, {pvdang,pcvinh}@ntt.edu.vn
[2] GUST, Vietnam Academy of Science and Technology, Hanoi, Vietnam

Abstract. The rapidly growth urbanization and using high-intensity land in urban areas recently are hot topics interested in researchers of 2-3-4D GIS (2-dimensional, 3-dimensional, and 4-dimensional geographic information system) more and more by the shapes of high-rise buildings (buildings) of diversity and abundance located on limited land funds. This problem is a big challenge for researchers of 2-3-4D GIS, how they can visually represent it into the 2D computer screen. This paper proposes to build space-data storage structures and to develop 3D space data interpolation algorithms (IA and NCA) for visual representation of buildings located on limited land funds in 3D geographical scientific space. This paper also presents experiments by these 3D bodies to represent visualization of buildings. Through these experimental results show able to support the authorities apply for management stages of urban techniques infrastructures in the near future.

Keywords: Limited land funds · 2-3-4D GIS · Spatial database
IA and NCA

1 Introduction

The increase in population, the demand for education as well as employment, and the policies of receiving immigrants in large cities are always expanding. Therefore, the demand of using high-intensity land [1] requires that we need to develop data structures three-dimensional (3D) space to manage the space object 2-3-4D located on the land fund limited. This management is a major challenge for GIS researchers 2-3-4D, because the shape of the buildings is very rich and diverse in the code samples.

Due to the increasing population and the influx of immigrants into the major cities, leading to more and more land in the urban area, which had limited far more limited. The problem set for the authorities was to develop the planning of buildings, taking the space above the land fund to make up for limited. This is the basis for this paper proposed the structures of spatial data storage and spatial data interpolation algorithms for 3D space.

To build a high-rise building that is the combination of the components as point, line, surface, and blocks. This paper applies the method B-REP (Boundary

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2019
Published by Springer Nature Switzerland AG 2019. All Rights Reserved
P. Cong Vinh and V. Alagar (Eds.): ICCASA 2018/ICTCC 2018, LNICST 266, pp. 81–97, 2019.
https://doi.org/10.1007/978-3-030-06152-4_8

Representations) to represent the object 0-1-2-2.5-3D based on the elements already defined in advance, including: point, line, surface, and blocks. In which, the Line could be the line segment, the arch, the circle; Surface may be flat, the surface polygons created by the circle, cone, the surface of the cylinder; Solid is the expansion of the surfaces to gig the 3D blocks, the blocks can: box, cone, cylinder, the combination of this block or a block of any [2, 3]. Therefore, appropriate B-REP method to gig for the space object has a shape often, artificial, and scalar. B-REP focuses on building objects and relationships between them [4].

The objective of this paper is to propose building the data storage structures and developed the 3D space data interpolation algorithms to visual representation of the high-rise building in 3D geography science space. The construction of the data storage structures in this space in order to minimize the storage of spatial data with the data interpolation algorithms for 3D space, to support to solve some part of land fund management problem limited.

The next section of this paper is organized as follows. Section 2 presents overview of studies, reviews and suggestions. Section 3 performs to develop data storage structures and interpolation algorithms of 3D space data. Section 4 illustrates experiments with the spatial components to build a high-rise building in 3D geography science space. Section 5 presents the results achieved and the development direction in the future. The final section is reference documents.

2 The Data Models of 3D and 4D GIS

2.1 Overview of the Data Models of 3D and 4D GIS

Over the years, researchers have proposed various models of data space, time, and semantics 3-4D. These models have great contribution in the development history of the 0-1-2-2.5-3-4D GIS. This contribution was evidenced through the GIS model is the author of proposals in the past, as follows.

2.1.1 3D Cadastral Model

In 2017, author Yuan Ding and associates [1] proposed the extrusion approach based on non-overlapping footprints (EABNOF) to build the geometry model and topology in the 3D Cadastral Model. EABNOF handles the case of complex 3D blocks. To reach EABNOF, the overlap between the overlapping traces of the input data files will be removed, including the division of the extrusion and processing traces of cadastral objects to fit together. The trace against new cross was born will be extruded to produce the original copies. To build the geometry model and topology for cadastral objects, there are 3 proposed evaluation criteria to identify and remove the excess from the original and then the original version of the composition space the same 3D or the features of the link structure will be incorporated. Special sections of EABNOF approach that groups of authors have applied to 2D data sets. The author team has tested two types of structures on the Pozi verifies EABNOF approach: a complicated

building and the furniture. EABNOF based on the traces of the 2D cadastral data sets and especially consistent with areas of the sets of 2D cadastral data to setting 3D cadastral system with lower costs.

2.1.2 TUDM Model

TUDM was a model of 4D spatial - temporal data proposed by N.G.T. Anh and coworkers in 2012 [5]. These authors have focused on developing a time dimension to integrate into the known 3D GIS space model. The time dimension in TUDM can be a time or a time period. The birth and extinction time of an object in TUDM can either be in the real world or be recorded in the database. TUDM can represent and store not only the evolutionary history of 0D, 1D, 2D objects but also the life cycle of 3D objects.

2.1.3 VRO-DLOD3D Model

The VRO-DLOD3D model was proposed by P.V. Dang and colleagues in 2017 [6]. The author group has researched and developed a visual representation of geographic features (people, buildings and geospatial space) along with relationships (blood relations, social relations, previous conviction relations, previous offence relations and vital relations) in three dimensions at different levels of detail, serving the protection of security and social order and safety in the area. The author group also presents data in forms through a number of typical queries at different levels of detail.

2.1.4 UDM Model

UDM (Urban Data Model) was a model of spatial data proposed by Coors in 2003 [7] based on four basic objects POINT, LINE, SURFACE, BODY. UDM uses two elemental objects NODE, FACE. ARC isn't proposed in this model. Each FACE is defined by 3 NODEs, so the model reduces some NODE-ARC, ARC-FACE relationships. Some topology relationships such as NODE are on FACE, NODE in BODY is not described. The obvious advantage of the UDM is the efficient data storage, the object-oriented analysis which is used urban management applications and representation of faces and blocks based on triangulation.

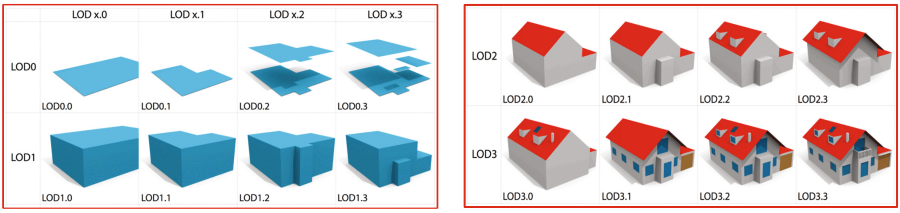
2.1.5 CityGML Model

CityGML model by Groger and colleagues proposed in 2007 [8]. The idea behind this model is to build a 3D city model as a model platform and open XML. The main purpose of CityGML is to develop common definitions related to the entities, attributes, and relationships in the model 3D cities to the different applications can share a common data source. CityGML is represented by objects of geometry GML3. This model is based on ISO 19107. Standard CityGML has an attribute not only space but also a semantic attribute. CityGML model supports 5 levels of detail (LOD): LOD0 is the most rugged, mostly 2.5D digital terrain model; LOD1 a block model, the buildings are represented as blocks with flat roofs; in LOD2 more complex buildings can be modeled, complex roof, settings such as stairs and balconies are available; LOD3 allow architectural models, may have detailed the walls, roof, doors gates, etc.; LOD4 completed LOD3 and include internal structure, such as guest rooms, doors, stairs, furniture, etc. It can be shown the same object in different LOD.

Author Kolbe and colleagues in 2009 [9] was used CityGML model and combined building new models for application in a number of German cities. These cities include, buildings, furniture, vegetation, land use, water areas, transportation (streets, rails, etc.) are defined in modules subject matter may be open wide in the future.

2.1.6 Improved the CityGML Model

The author group Biljecki and colleagues [10] have improved CityGML model by refining LODs. This improvement increases the level of detail of the detailed CityGML from 5 to 16 the level of detail. This improvement is made using a geometric display. Each level of CityGML is smoothed over 4 times, as illustrated in Fig. 1 (see left and right).



(left) (right)

Fig. 1. Visual illustration of 16 LODs smoother for a residential building [10]

2.1.7 ELUDM Model for 2.5D Objects

ELUDM (Enhanced Levels of detail Urban Data Model) was a model proposed by P.V. Dang and coworkers in 2011 [11]. These authors have proposed the addition of LOD (Levels of Detail) and complex links between Surface, Line, Point and LOD to the ELUDM to serve visual representation of 2.5D objects at multiple different levels. User defines the number of levels. The diversity of the visualizing will respond the requirements of different applications. This approach can extend for LOD of objects 3D that not depend on semantic.

2.1.8 3D Array Model

Array 3D model was proposed by Rahman in 2005 [2, 3]. Models with simple data structures used to represent most of the 3D object. The 3D elements in the Array have the value 0, 1. In the description that the base value of 0, 1 describes the value that each element in the 3D Array occupied by 3D objects. If a 3D object is being scanned in a 3D array which element of the array is initialized with the initial value 0. After scanning to 3D objects, elements worth 1 present information for 3D objects. If scanning 3D objects with high resolution, the size of the array in each direction will increase as volumetric 3D data should describe also increased and require large storage space.

2.1.9 Octree Model

Octree model is an extension of the quadtree into the octal tree. Octree representation is a 3D model based on volume [3, 15]. Octal tree gives us the picture, and this is a method represented by the data structure tree. Generally, an octal tree is defined based on a cube that contains the smallest 3D objects needs performing. Original cube will be divided into 8 cube offspring. An octal tree is based on the decomposition of recursive algorithm follow. Each Node in the tree is or leaf, or 8 seedlings. Each seedling will be checked before being divided into 8 different seedlings.

2.1.10 CSG Model

The model CSG was proposed by Rahman in 2008 [2, 3]. CSG performed a 3D object by combining the 3D element has been defined before. The basic 3D blocks such as: cube, cylinder, and sphere. The relationship between the figures includes: transformation and the mathematical treatise storage class. These transformations include translation, rotation, allowed to measure change. The comment class storages include union, intersect and except. CSG is often used in CAD. CSG is very convenient in the calculation of the volume of the object, so the CSG does not conform to the performance for the objects have unusual geometric shapes.

2.1.11 Combined Model Between B-REP and CSG

Model B-REP + CSG by Chokri and colleagues were suggested in 2009 [12]. The model is based on the idea of performing 3D objects by combining the two methods of B-REP and CSG. In B-REP models using 4 basic object point, line, surface, and solid. A line is defined by the first and last points 2. A surface is defined from a closed string with or without direction. A surface may be full or empty. Empty_surface to describe the gap. Many surfaces in the same plane make up Composit_surface. A solid is represented by a set of surrounding surfaces. Solid may be full or empty. FULL_-Solid is created by a set of scalar surfaces, so the interior of Solid is not modeled. In CSG element objects include cylinders, spheres, and prisms. A CSG_Composit is the union, intersect, and except of 2 Solids. The inside of Solids is not modeled. The cubes, triangular prisms, polygonal prisms are the derivative objects of the prisms. The advantage of this approach is based on the advantages of the B-REP method, which demonstrates very well the external boundaries that make up the object and the advantage of the CSG approach is the minimization of storage data (Table 1).

Table 1. Classification of the models

Model types	Name of the models
B-REP (B)	3D Cadastral Model, TUDM Model, VRO-DLOD3D Model, UDM Model, CityGML Model, Improved the CityGML Model, ELUDM Model
VOXEL (V)	3D Array Model, Octree Model
CSG (C)	CSG Model
COMBINING B-V-C	B-REP and CSG

2.2 Classification of the Models

The selection 3D data models to represent 3D GIS objects for a specific application will determine the methods of storing, retrieving, managing, treating ways when the show and the obvious required data is also different. A model can combine all areas is not practical. The data models of the authors have proposed a paper analyzes aggregate, and is divided into four main categories as follows:

Type 1: *Representation of 3D objects by B-REP. B-REP method to represent a 3D object-based element has been defined, including: Point, Line, and Surface, Solid. B-REP suitable for representing 3D objects shaped conventional and scalar. B-REP focuses on building objects and the relationships between them [12].*

Type 2: *Representation of 3D objects by voxel elements, such as pixels in 2D GIS. Voxel method represents a 3D object based on ideas split an object into sub-elements, each element is called a voxel child [13]. One element to be considered as a geographical space and is assigned by an integer [14].*

Type 3: *Representation of 3D objects by combining the basic 3D block (CSG) [2].*

Type 4: *Representation of 3D objects by combining three types above*

2.3 Proposing Storage Structures and Interpolation Algorithms of 3D Space Data

Through the summarizing and classification models of the authors represent proposed above. We have the following comments below. Through these comments, we propose to build the structures of spatial data storage for the components of a residential building and the interpolation algorithms of 3D space data will be presented in Sect. 3 of this paper.

Comment 1: *The models proposed by authors are most representative of type B-REP.*

Comment 2: *The models proposed by authors are not to mention structure installed in a database management system.*

Comment 3: *The data interpolation algorithms of 3D space has not been mentioned.*

Comment 4: *The combination of the spatial component into a residential building also has not been mentioned.*

3 Developing Data Structures and Interpolation Algorithms of 3D Space Data

3.1 Developing 3D Space-Data Structures

Spatial data records shapes, sizes, and locations of 3D objects [6]. To manage spatial data is a major challenge for researchers, including insert spatial data, update spatial data, delete spatial data, query spatial data, and query spatial data over time.

However, these are now multiple database management system support the spatial data management, including Oracle database management system, but requires us to use and implement combination the spatial data structure inherent in Oracle to build our own structures effectively.

The EABNOF [1] is a reach extrusion based on non-overlapping traces to build the model geometry and topology in the 3D Cadastral Model. EABNOF handles the case of complex 3D blocks and the overlap between the overlapping traces of the input data will be removed. There are three proposed evaluation criteria to identify and remove the excess from the original and then the original versions of the composition space the same 3D or the features of the link structure will be incorporated. CSG model [2, 3] represents 3D objects by combining 3D elements (cube, cylinder, and sphere) have been defined before. The relationship between the figures includes: transformation (translation, rotation, measure change) and logical operations (union, intersect, and except). B-REP + CSG model [12] represents 3D objects by combining the two methods of B-REP and CSG. B-REP uses four basic objects point, line, surface, and solid to representing 3D objects. The advantage of this approach is based on the advantages of the B-REP method (which demonstrates very well the external boundaries that make up the 2D and 3D object.). Through these three approaches above, we have seen that, the authors have not been mentioned construction of data structures as well as construction of interpolation algorithms of 3D spatial data. These are the premise for this paper proposing development of data structures as well as 3D spatial data interpolation algorithms.

Developing the data structures for the 3D space component and rich diversity of designs is a huge challenge. Within the limits of the paper, we will illustrate the spatial components of a typical building, in order to satisfy the following criteria:

Criterion 1: *Spatial components can be defined arbitrarily by user.*

Criterion 2: *Data structures are simple but still are able to govern space components.*

Criterion 3: *Data structures must conform to spatial data interpolation algorithms.*

In practice, some urban areas are always crowded in all aspects such as housing, people, goods, vehicles, etc. We proposed scenario simulates a number of residential areas, but only set topics centered on residential housing and grab space area on high ground compensate for limited land funds. Figure 2 describes some residential areas today which there are different types of housing and land built on limited funds. The types of buildings have structural diversity, richness of design and challenge us to represent them in 2D computer screen. There are a number of buildings in residential areas are now built based on criteria overhead space taken up for the limited land fund, which see Fig. 2 with Sunny and Bee villa. Bee Villa is built by combining the basic components, including prismatic stand, prismatic triangle, prism quadrilateral, rectangular, square, lines, and points (see Figs. 3 and 4). Figure 5 performs a combination of basic components to form a residential building, these components including object types need management space.

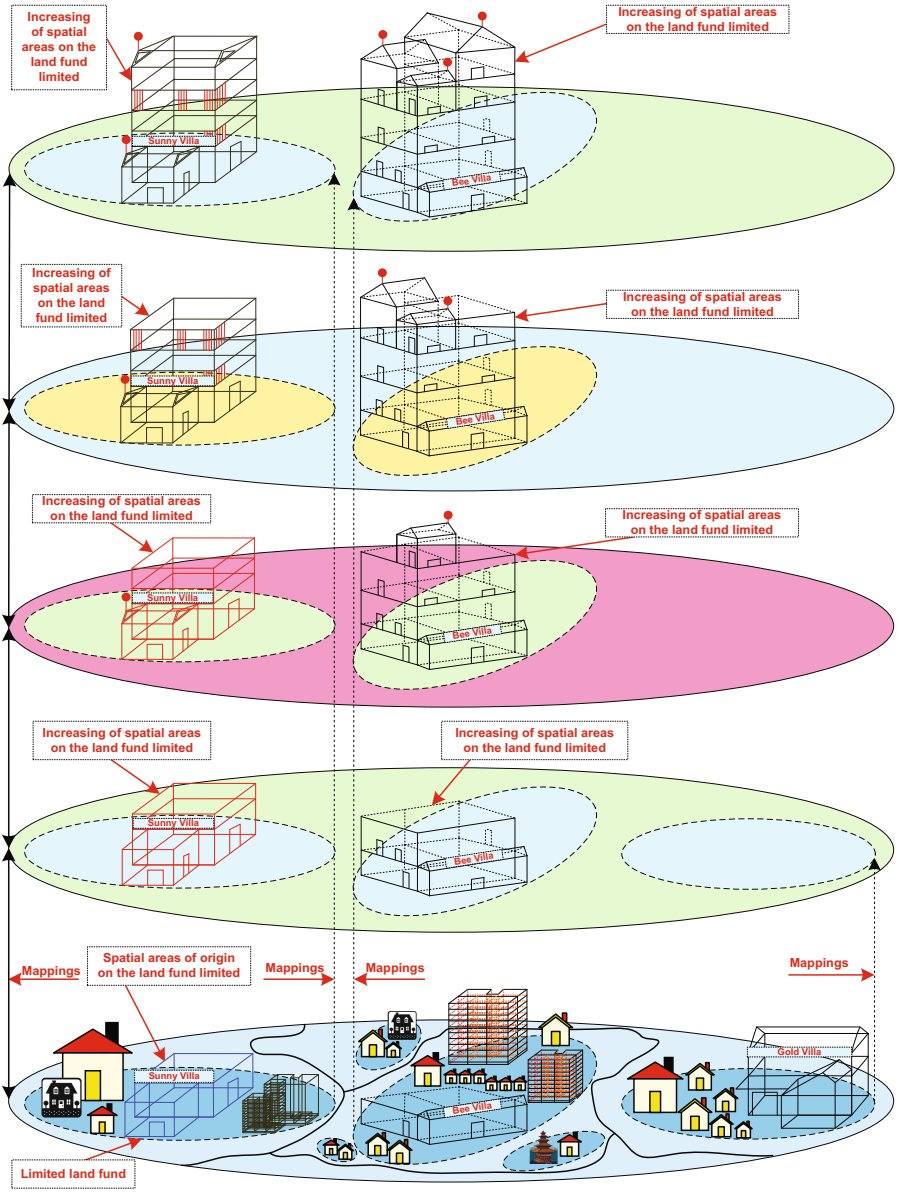


Fig. 2. Illustration of some residential area with residential buildings built on high space taken up for the limited land funds

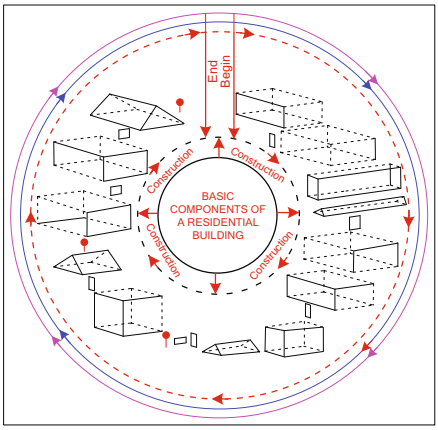


Fig. 3. The process of combining the basic components of a residential building on the limited funds

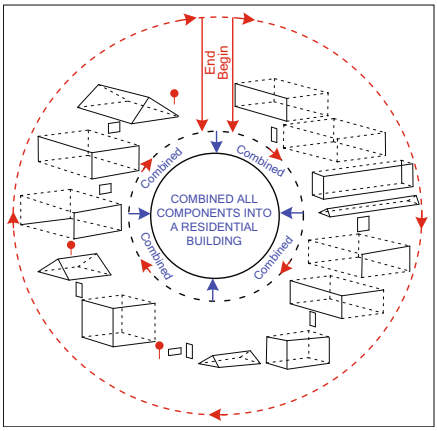


Fig. 4. Making combinations of basic components to become a residential building

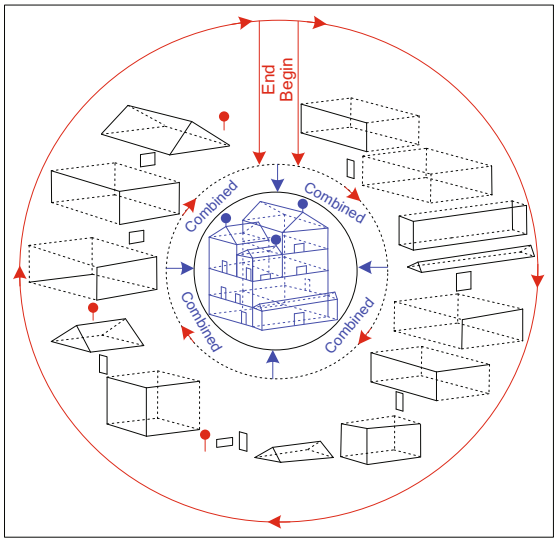


Fig. 5. Processing of the basic components of a residential building on limited funds

We categorized the space objects are as follows: Type 1: vertical or horizontal prisms; Type 2: vertical or horizontal equilateral triangular prisms; Type 3: vertical or horizontal equilateral rectangular prisms; Type 4: vertical or horizontal rectangles; Type 5: vertical or horizontal squares; Type 6: vertical or horizontal lines; Type 7: Points. Therefore, we propose the representation of spatial objects by combining the data structures available in the Oracle database management system to build the data structures for the object types need to management takes illustrated in Fig. 5. Construction general structure, including creating structure of nodes array contains nodes: create type nodexyz as object (idn char (10 byte), x float, y float, z float), meanings is object support to single data types; create or replace type arrnode is array of nodexyz, meanings creating 2D object contains nodes. Objects in blocks: Structural representation of the object space is vertical or horizontal prisms; Vertical or horizontal equilateral rectangular prisms; Vertical or horizontal equilateral rectangular prisms; Vertical or horizontal pentagonal prisms by requiring users to provide data nodes bottom, height h, and the type of the object space.

3.1.1 Creating Spatial Data Structure of Blocks Form

```
Table [blocks] (
    Idvl          char(10 byte)          not null  constraint fkbl references villa(Idvl),
    Idb           char(10 byte)          not null  constraint pk_blocks  primary key,
    Height        float                  not null, --Height is attribute height indentify of object
    Typeshape     nvarchar2(50 byte)     not null, --Vertical:VY or Horizontal: HX or HZ
    Description   nvarchar2(100 byte)    not null, --Describling object of solid form
    Arraynode     arrnode                not null  --Arraynode is 2D array
);
```

3.1.2 Creating Data Sets of Spatial Objects into Structural Blocks

Insert into [blocks] values('VL1', 'B1', 15, 'VY', N 'This is a vertical prism.', arraynode(nodexyz('N1', x1, y1, z1), nodexyz('N2', x2, y2, z2), nodexyz('N3', x3, y3, z3), nodexyz('N4', x4, y4, z4)));

Insert into [blocks] values('VL1', 'B2', 7, 'HX', N 'This is a rectangular prism.', araynode(nodexyz('N5', x1, y1, z1), nodexyz('N6', x2, y2, z2), nodexyz('N7', x3, y3, z3)));

Insert into [blocks] values('VL1', 'B3', 6, 'VY', N 'This is vertical equilateral rectangular prisms.', arraynode(nodexyz('N8', x1, y1, z1), nodexyz('N9', x2, y2, z2), nodexyz('N10', x3, y3, z3), nodexyz('N11', x4, y4, z4)));

Insert into [blocks] values('VL1', 'B4', 7, 'HX', N 'This is a horizontal rectangular prisms.', arraynode(nodexyz('N12', x1, y1, z1), nodexyz('N13', x2, y2, z2), nodexyz ('N14', x3, y3, z3), nodexyz('N15', x4, y4, z4), nodexyz('N16', x5, y5, z5)));

3.1.3 Creating Spatial Database Structure of Flatbed Form

```
Table [flats] (
    Idvl          char(10 byte)          not null  constraint fk references villa(idvl),
    Idf           char(10 byte)          not  null  constraint pkflats  primary key,
    Typeshape     nvarchar2(50 byte)     null, --T:Triangle,R:Rectangle,S:Square, Surface
    Description   nvarchar2(100 byte)    not  null,    --Describling object of flats form
    Arraynode     arrnode                not  null     --Arraynode contains nodes
);
```

3.1.4 Creating Data Sets of Spatial Objects into Structural Flats

Insert into [flats] values('VL1', 'F1, 'R', N 'This is rectangle.', arraynode (nodexyz ('N17', x1, y1, z1), nodexyz('N18', x2, y2, z2), nodexyz('N19', x3, y3, z3), nodexyz ('N20', x4, y4, z4)));

Insert into [flats] values('VL1', 'F2, 'S', N 'This is square.', arraynode (nodexyz ('N21', x1, y1, z1), nodexyz('N22', x2, y2, z2), nodexyz('N23', x3, y3, z3), nodexyz ('N24', x4, y4, z4)));

Insert into [flats] values('VL1', 'F3, 'L', N 'This is string line.', arraynode (nodexyz('N25', x1, y1, z1), nodexyz('N26', x2, y2, z2)));

Insert into [flats] values('VL1', 'F4', 'P', arraynode(nodexyz('N27', x1, y1, z1)), null, N 'This is point.');

3.2 Developing 3D Spatial Data Interpolation Algorithms

Spatial data describes the shapes, sizes, and locations of spatial objects [6]. To represent objects in 3D, we have to provide them with positions in nodes in 3D, and then take into account their shape and size. This paper applies the B-REP [12] approach to represent 3D objects by boundaries. We propose to develop two algorithms: Algorithm 1 is capable of interpolating 3D data and is named Interpolation Algorithm (IA); Algorithm 2 is capable of connecting nodes to form Lines, which represent 3D space objects and is named Nodes Concatenation Algorithm (NCA). We see in Fig. 6 an illustration of the composition of a residential building on a limited ground, and in this context the role of the IA algorithm must be able to interpolation of 3D spatial data and the role of the NCA algorithm must be able to connect the Nodes together to form Lines, to visualize 3D objects. From Fig. 6, we detail the processing of IA and NCA algorithms for visual representation of vertical prisms and horizontal triangular prisms, as follows, we use two IA and NCA algorithms to represent the vertical prism (see Fig. 7), next, we use two IA and NCA algorithms to represent the horizontal triangular prism (see Fig. 8). Two ways of representation above (see Figs. 7 and 8) will illustrate by two IA and NCA algorithms the following:

- **IA (Interpolation Algorithm)**

Input : $Type_1$ is pa method, including: A_2D_1Nodes contains 3D spatial datasets originally; H is height; Shape is picture type, nNodes of A_2DNodes, flags, C_2, C_3; $Type_2$ is public variables: A_2D_2Nodes, A_2D_3Nodes, C_1=0, flag=true

Output: A_2D_1Nodes contains datasets of 3D space interpolated.

```
01: IA(ref A2D1Nodes, h, shape, nNodes, flags)
02:   A2D3Nodes = new string[A2D1Nodes.Len]; //A2D3Nodes contains Nodes
03:   C2 = 1;
04:   C3 = 0;
05:   if(flags ==true)
06:      if(flag==true)      //Test first picture
07:         A2D3Nodes =A2D2Nodes = A2D1Nodes;
08:         C1 = C3 = nNodes;
09:         flag=flase;
10:      else if (flag==false)     //Test the next picture
11:         for(i=0;i<=A2D1Nodes.Upper(0)&&A2D1Nodes[i,0]!=null;i++,C1++,C3++)
12:            for(j=0; i<= A2D1Nodes.Upper(1); j++)
13:               switch(j)
14:                  case 0: //--IDN
15:                        A2D2Nodes[C1, j] = "N"+ riseIndexNodes;
16:                        A2D3Nodes[C3, j] = A2D2Nodes[C1, j]; break;
17:                  case 1: //--x
18:                        A2D2Nodes[C1, j] = A2D1Nodes[i, j];
19:                        A2D3Nodes[C3, j] = A2D2Nodes[C1, j]; break;
20:                  case 2: //--y
21:                        A2D2Nodes[C1, j] = A2D1Nodes[i, j];
22:                        A2D3Nodes[C3, j] = A2D2Nodes[C1, j]; break;
23:                  case 3: //--z
24:                        A2D2Nodes[C1, j] = A2D1Nodes[i, j];
25:                        A2D3Nodes[C3, j] = A2D2Nodes[C1, j]; break;
26:               end switch
27:            end for j
28:         end for i
29:      end if
30:   end if
31:   if(shape == "VY" || shape == "HX" || shape == "HZ")  //3D spatial data interpolation
32:      for(i=0; i<=A2D2Nodes.Upper(0) && C2++<=nNodes; i++, C1++, C3++)
33:         for(j=0; j<=A2D2Nodes.Upper(1); j++)
34:            switch(j)
35:               case 0: //--IDN
36:                     A2D2Nodes[C1, j] = "N" + riseIndexNodes;
37:                     A2D3Nodes[C3, j] = A2D2Nodes[C1, j];    break;
38:               case 1: //--x
39:                     A2D2Nodes[C1, j] = A2D1Nodes[i, j] + (shape=="HX"?h:0);
40:                     A2D3Nodes[C3, j] = A2D2Nodes[C1, j];    break;
```

```
41:                 case 2: //--y
42:                       A2D2Nodes[C1, j] = A2D1Nodes[i, j] + (shape==”VY”?h:0);
43:                       A2D3Nodes[C3, j] = A2D2Nodes[C1, j];    break;
44:                 case 3: //--z
45:                       A2D2Nodes[C1, j] = A2D1Nodes[i, j] + (shape==”HZ”?h:0);
46:                       A2D3Nodes[C3, j] = A2D2Nodes[C1, j];    break;
47:              end switch
48:           end for j
49:        end for i
50:    end if
51:    A2D1Nodes=A2D3Nodes; //Assign data after 3D spatial data interpolation
52:    End IA;
```

- **NCA (Nodes Concatenation Algorithm)**

Input : A_2DNodes contains datasets of 3D spatial data; Shape is picture type.
Output: A_2DLines contains datasets of Lines connected by Nodes

```
01: NCA(A2DNodes, out A2DLines, Shape)
02:    CNodes = 0;                                  //Summarizing of nodes current
03:    A1DNodes = new string[A2DNodes.Len];  //A1DNodes contains all Nodes of A2DNodes
04:    A2DLines  = new string[A2DNodes.Len];  //Containing datasets of linesconnect by Nodes
05:    for(i = 0; i<= A2DNodes.Upper (0); i++)   //A2DNodes copy IDN each Node
06:       if(A2DNodes[i, 0) == null) then break;  //Exit for when meets IDN = null
07:       A1DNodes[i] = A2DNodes[i, 0);          //Copy IDN from A2DNodes into A1DNodes
08:       CNodes++;                                 //Count Nodes
09:    end for
10:    col = midI = rowI = 0;
11:    for(i=0; i<CNodes; i++)                     //Plan 1: concatenation bottom prism
12:       midI = i;
13:       A2DLines[i, col++] = “L” + riseIndexLines; //Create Index for Line
14:       A2DLines[i, col++] = A1DNodes[midI];     //Copy IDN of A1DNodes in each row i
15:       if(midI==CNodes)   //Rear array append with array middle element
16:          A2DLines[i, col++] = (Shape is true)?A1DNodes[0]:A1DNodes[midI/2];
17:          col =0;
18:          rowI = i;          //Save row order to i of A2Dlines, to update next for row i
19:          break;             //Exit for
20:       end if
21:       if (Shape is true)     //Comment: T:Triangle, R:Rectangle, S:Square, S:Surface, L:Line
22:          A2DLines[i, col++] = A1DNodes[midI];
23:       else                   //Comment: Blocks
24:          A2DLines[i, col++]=(midI==(CNodes/2))?A1DNodes[0]:A1DNodes[midI];
25:       end if
26:       col=0;                 //Reset for col of rowI
27:    end for
28:    for(i=0; i< CNodes/2 && Shape not true; i++) //Plan 2: concatenationheight of prism
29:       A2DLines[rowI + i + 1, col++]="L" + riseIndexLines; //Create Index for Line
30:       A2DLines[rowI + i + 1, col++]=A1DNodes[i];   //Copy IDN first A1DNodes in A2DLines
31:       A2DLines[rowI + i + 1, col++]=A1DNodes[(CNodes/2)+i]; //Copy IDN
32:       col=0; //Reset for col of rowI
33:    end for
34: End NCA;
```

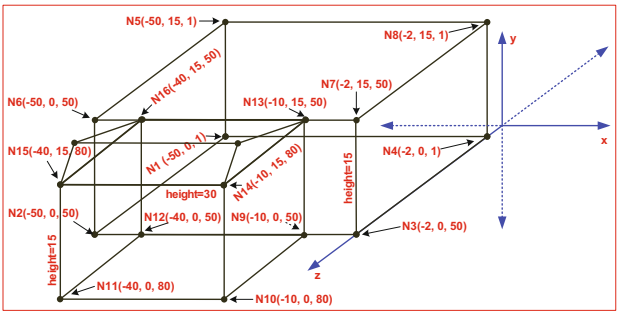


Fig. 6. A combination of a residential building or villa on a narrow ground

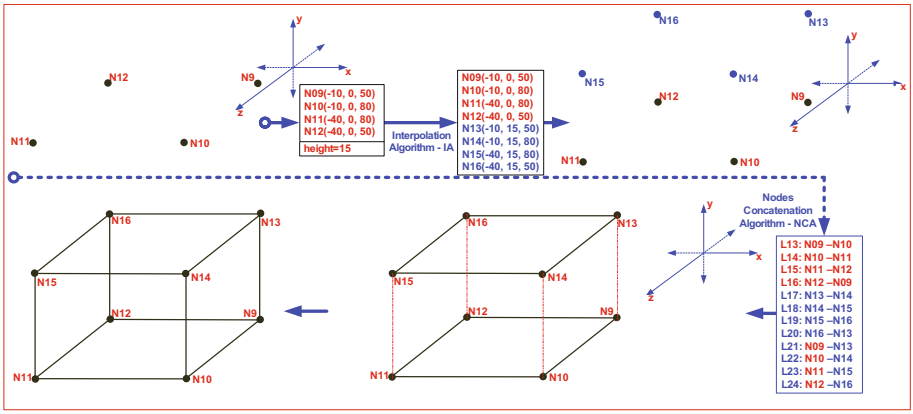


Fig. 7. Representation of vertical prisms by IA and NCA algorithms

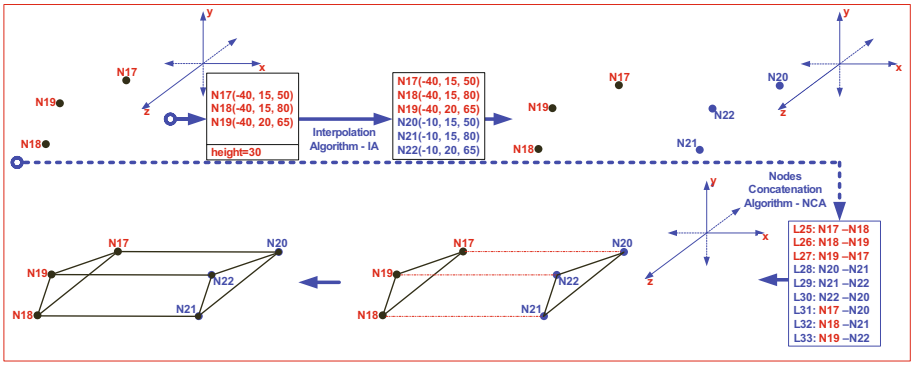


Fig. 8. Representation of horizontal triangular prisms by IA and NCA algorithms

4 Experiments

In this section, we have detailed the data storage structures and interpolation algorithms of 3D spatial data, and have seen the challenges and difficulties of 3D spatial data processing. To illustrate the results of this study in Sect. 3, we perform an empirical installation on Oracle database administration and C# programming language [16–18]: two IA and NCA algorithms; Data structures include: structure representing spatial objects that are prismatic or vertical; The triangular prism is either vertical or horizontal, by the users providing the bottom data, height h, and type of shape of the object. Experimental results are visual representations of spatial components of a residential building located on a narrow ground (see Figs. 9 and 10).

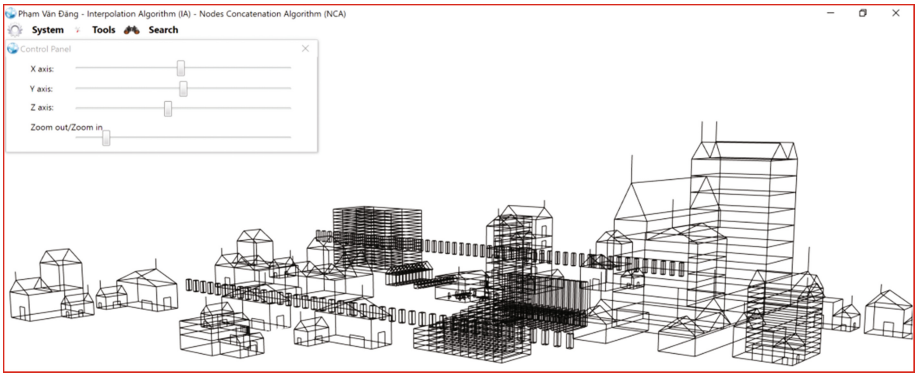


Fig. 9. A representation of urban areas including residential buildings and villas

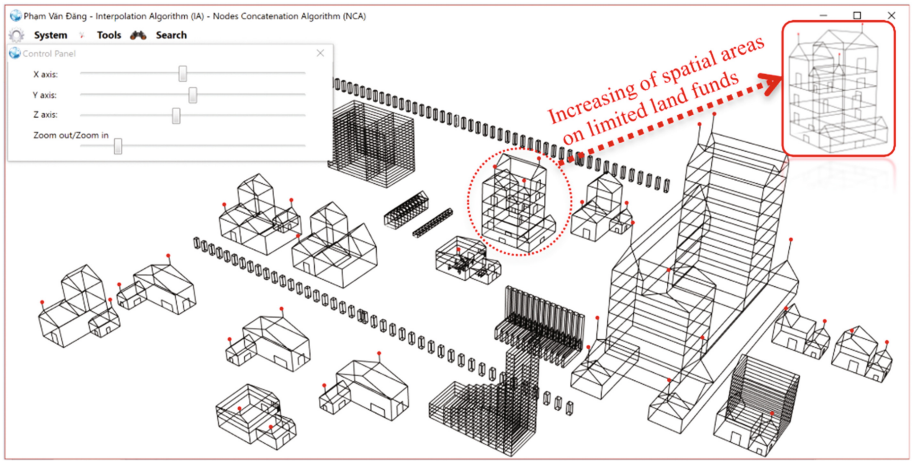


Fig. 10. The form illustrating space components is grouped into a residential building

5 Conclusion

This paper analyzes and synthesizes 3-4D space, time, and semantic data models proposed by authors in the past. These models have contributed greatly to the long history of the development of GIS. The paper classifies models into four main categories, namely, type 1 represents 3D objects by boundaries (B-REP); Type 2 represents 3D objects by voxel elements; Type 3 represents 3D objects by combining basic 3D blocks (CSG); Type 4 represents 3D objects by combining these three types. By classifying and making these observations, we propose the construction of data structures and interpolation algorithms of 3D spatial data, which the authors did not mention are detailed in the paper. Subsequently, the paper has been experimentally set up by combining spatial components to build residential buildings located on narrow ground by means of taking up space overhead to compensate for limited land funds. In addition, the paper proposes the next direction of improvement of the patterns and designs for 3D objects and the improvement of IA and NCA algorithms to eliminate the remaining 3D space data nodes overlap when performing combining 3D space blocks.

References

1. Ding, Y., Jiang, N., Yu, Z., Ma, B., Shi, G., Wu, C.: Extrusion Approach Based on Non-Overlapping Footprints (EABNOF) for the construction of geometric models and topologies in 3D cadasters. ISPRS Int. J. Geo-Inf. **6**(8), 232 (2017). https://doi.org/10.3390/ijgi6080232. (cc by 4.0)
2. Rahman, A.: Developing Three-dimensional topological model for 3D GIS. Project Report, UTM (2005)
3. Rahman, A.: Spatial data modeling for 3D GIS. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-74167-1
4. Tet-Khuan, C., Abdul-Rahman, A., Zlatanova, S.: 3D spatial operations in geo DBMS environment for 3D GIS. In: Gervasi, O., Gavrilova, Marina L. (eds.) ICCSA 2007. LNCS, vol. 4705, pp. 151–163. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74472-6_12
5. NG, T.A., Vinh, P.T, Duy, H.K..: A study on 4D GIS spatio-temporal data model. In: Proceedings of IEEE 4th Conference on Knowledge and Systems Engineering, KSE 2012, Danang, Vietnam, August 2012. IEEE Computer Society (2012). Order Number P4670. ISBN-13: 978-0-7695-4760-2
6. Dang, P.V., et al.: Visual representation of geographic objects in 3D space at levels of different details. In: Proceeding of the 10th National Conference on Fundamental and Applied IT Research – FAIR 2010, Da Nang, 17–18/08/2017, pp. 979–988. Natural Science and Technology Publishing House (2017). https://doi.org/10.15625/vap.2017.000115. ISBN: 978-604-913-614-6
7. Coor: 3D-GIS In: Networking Environments, Computers, Environment and Urban Systems, pp 345–357 (2003)
8. Groger, et al.: City Geography Markup Language (CityGML) Encoding Standard. Open Geospatial Consortium Inc. (2007)
9. Kolbe, T.H.: Representing and exchanging 3D city models with CityGML. In: Lee, J., Zlatanova, S. (eds.) 3D Geo-Information Sciences, pp. 15–31. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-87395-2_2

10. Biljecki, F., Ledoux, H., Stoter, J.: An improved LOD specification for 3D building models. Comput. Environ. Urban Syst. **59**, 25–37 (2016)
11. Dang, P.V., et al.: Levels of Detail for Surface in Urban Data Model, International Conference on Future Information Technology, vol. 13, pp. 460–464. ICFIT, Singapore (2011). ISBN 978-981-08-9916-5
12. Chokri, K., Mathieu, K.: A simplified geometric and topological modeling of 3D building enriched by semantic data: combination of surface-based and solid-based representations. In: ASPRS 2009 Annual Conference Baltimore, Maryland (2009)
13. The Three Dimensional Visualization & Analysis of Geographic Data, by: James Swanson Maps.unomaha.edu/Peterson/gis/Final_Projects/1996/Swanson/GIS_Paper.html. Accessed Apr 2017
14. Lieberwirth, U.: 3D GIS voxel-based model building in archaeology. Archaeopress, Oxford (2008)
15. Gröger, G., et al.: Representation of a 3D city model in spatial object-relational databases. In: XXth ISPRS Congress, Geo-Imagery Bridge- ing Continents, Commission 4, ISPRS (2004)
16. A Tool for visualizing 3D Geometry Models. http://www.codeproject.com/Articles-/42992/A-Tool-for-Visualizing-D-Geometry-Models-Part. Accessed Nov 2017
17. Oracle Spatial User's Guide and Reference, Release 9.0.1, Part Number A88805-01, June 2001. Accessed Nov 2017
18. Elem_Info_Arraying: An alternative to SDO_UTI-L.GetNumRings and querying SDO_E-LEM_INFO_it self. http://www.spatialdbadvisor.com/oracle_spatial_tips_-tricks/89/sdo_utilget_numrings-an-alternative. Accessed May 2017