AN ENCODING METHOD FOR COMPRESSING GEOGRAPHICAL COORDINATES IN 3D SPACE

Chengyang Qian [ab]*, Ruqiao Jiang [b]*, Muxian Li [b]

[a] Key Laboratory of Virtual Geographic Environment (Nanjing Normal University), Ministry of Education – cyqian@gmail.com
[b] Suzhou Industrial Park Geone Information Technology Co., Ltd. – jiangrq@dpark.com.cn

Commission VI, WG VI/4

KEY WORDS: Compression Encoding; Geographical Coordinates Compression; 3D Map; Octree; Cube Index Code;

ABSTRACT:

This paper proposed an encoding method for compressing geographical coordinates in 3D space. By the way of reducing the length of geographical coordinates, it helps to lessen the storage size of geometry information. In addition, the encoding algorithm subdivides the whole space according to octree rules, which enables progressive transmission and loading. Three main steps are included in this method: (1) subdividing the whole 3D geographic space based on octree structure, (2) resampling all the vertices in 3D models, (3) encoding the coordinates of vertices with a combination of Cube Index Code (CIC) and Geometry Code. A series of geographical 3D models were applied to evaluate the encoding method. The results showed that this method reduced the storage size of most test data by 90% or even more under the condition of a speed of encoding and decoding. In conclusion, this method achieved a remarkable compression rate in vertex bit size with a steerable precision loss. It shall be of positive meaning to the web 3d map storing and transmission.

1. INTRODUCTION

3D urban data is one of the most important basic data for the study of the virtual geographical environment and the application of Smart City. Generally, the quality of 3D urban models is refined as much as possible to meet the requirements of visualization effects of design models. As a result, if we want to make use of 3D city models over the Internet and to bring it up to the required standard of the network transmission in real-time rendering, proper subdivision and encoding techniques need to be used to compress data while remaining the key features and appearance as best as possible. Past researches could be grouped into the following categories: (1)3D model Compression and simplification. Earlier studies focused on optimizing model storage structure such as generalized triangle meshes(Deering, 1995), floating-point data compression method(Ha and Yoo, 2016; Isenburg et al., 2004; Isenburg et al., 2005), topological surgery scheme(Taubin and Rossignac, 1998), and random accessibility of mesh compression framework(Choe et al., 2009). 3D model simplification was also solved by cartographic methods(Du et al., 2008; Forberg, 2007; Kada, 2008; Mehra et al., 2009; Thiemann and Sester, 2004). (2) Multi-resolution LOD algorithms. Representative works such as multi-resolution terrain mesh visualization(De Fioriani et al., 1997), progressive mesh(Hoppe, 1996), view-dependent progressive mesh(Hoppe, 1998) were proposed to reduce details at different scales. For regular grids, a variety of algorithms were designed based on recursive partitioning and visibility culling. For instance, real-time continuous LOD surface visualization algorithm was designed based on hierarchical quadtree(Lindstrom et al., 1996). Particularly, the pyramid model is one of the most widely used multi-resolution LOD methods when visualizing geographic data(Dang et al., 2014; Du et al., 2006; Luebke, 2003). (3) Optimization for Internet transmission or visualization. Gong (Gong et al., 2011) proposed an extended dynamic 3D R-tree structure concerning LOD which improves the 3D query performance. Ma (Ma and Li, 2016) proposed the KD-tree based on the compression method of a 3D dynamic scene which reduces the limitation of network bandwidth. Benefiting from both regular and recursive rules, the octree is capable of compressing the model size due to restructuring spatial relationship and finally accelerates searching speed (Peng and Kuo, 2005; Yue et al., 2009). This is also usually used for spatial subdivision in 3D geographic scenes (Jiong et al., 2017; Lixin and Jieqing, 2009).

This paper started from the geographical space subdivision, and then transform the geographical coordinates into the relative coordinates in the subdivision space according to the octree structure. Finally, testing results were discussed to illustrate the performance and practicability of the method.

In later sections, a statement of the framework of our method, which containing an octree-based subdivision and an encoding structure, is elaborated. Following by the description of the whole process of geometry encoding and decoding, the evaluation indicator and result presented.

2. METHODOLOGY

2.1 Overview of the method

This method of encoding in 3D geographic space aiming to speed up the transmission of complicated and dense 3D models in web maps, includes three phases: (1) the whole 3D geographic space being iteratively subdivided into cubes based on octree structure, (2) resampling coordinates of vertices into a discrete representation in each cubes at different subdivision depth of the octree, (3) encoding the geometry of models with

* Corresponding author

both subdivision index as well as resampled coordinates and then generating the final LOD models, as is shown in Figure1.

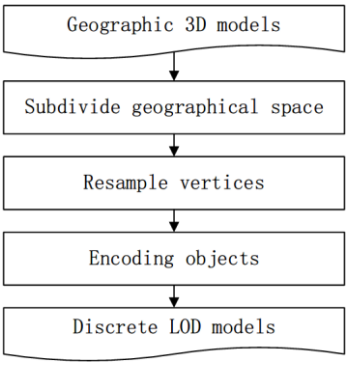

Figure 1. A framework of the encoding method

2.2 Octree-based Subdivision and Encoding

The full extent of the space is equally subdivided into 8 cubic subspaces at the first depth level. And then each cube is iteratively subdivided into 8 equal parts at the next depth level until the depth meets the predetermined value. An octree can be defined in this way, and each node can represent the subdivided space. Every node in the octree is encoded with the combination of particular codes related to halving depth and position on r, c, h coordinate axis, which is called as the Cube Index Code (CIC) in this method. At the first depth, the root cube represents the full space, whose code is defined as (1,1,1).

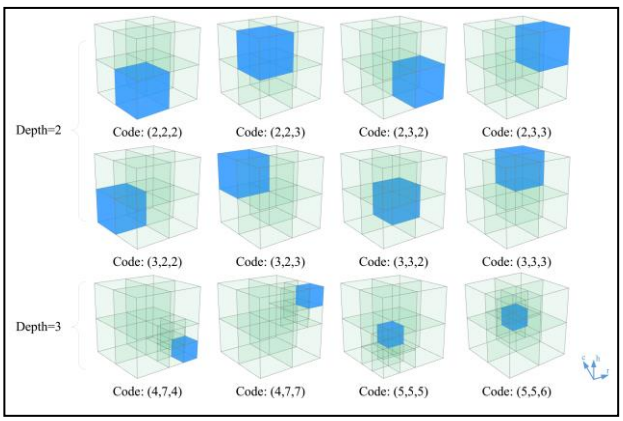

Figure 2. Cube Index Code of octree nodes

The CIC consists of r, c, h coordinates which is recursively defined in a halving way. The relation between halving codes is shown in (1) - (8). Therefore, the node$_0$ on the bottom-left corner is encoded as (2,2,2) while the code of the node$_7$ on the top-right corner is (3,3,3) at the second depth. Figure 2 illustrates all the codes of nodes at depth2.

$$node_0\begin{cases} r_{depth+1}=2r_{depth} \\ c_{depth+1}=2c_{depth} \\ h_{depth+1}=2h_{depth} \end{cases} \quad (1) \qquad node_1\begin{cases} r_{depth+1}=2r_{depth} \\ c_{depth+1}=2c_{depth}+1 \\ h_{depth+1}=2h_{depth} \end{cases} \quad (2)$$

$$node_2\begin{cases} r_{depth+1}=2r_{depth}+1 \\ c_{depth+1}=2c_{depth} \\ h_{depth+1}=2h_{depth} \end{cases} \quad (3) \qquad node_3\begin{cases} r_{depth+1}=2r_{depth}+1 \\ c_{depth+1}=2c_{depth}+1 \\ h_{depth+1}=2h_{depth} \end{cases} \quad (4)$$

$$node_4\begin{cases} r_{depth+1}=2r_{depth} \\ c_{depth+1}=2c_{depth} \\ h_{depth+1}=2h_{depth}+1 \end{cases} \quad (5) \qquad node_5\begin{cases} r_{depth+1}=2r_{depth} \\ c_{depth+1}=2c_{depth}+1 \\ h_{depth+1}=2h_{depth}+1 \end{cases} \quad (6)$$

$$node_6\begin{cases} r_{depth+1}=2r_{depth}+1 \\ c_{depth+1}=2c_{depth} \\ h_{depth+1}=2h_{depth}+1 \end{cases} \quad (7) \qquad node_7\begin{cases} r_{depth+1}=2r_{depth}+1 \\ c_{depth+1}=2c_{depth}+1 \\ h_{depth+1}=2h_{depth}+1 \end{cases} \quad (8)$$

This encoding method ensures two important rules. Firstly, each of the nodes in different depths of octree contains unique identification. Secondly, only one digit of the code is different between any two adjoining nodes at the same depth and the difference is 1. Given that the code of space subdivision is integral, it code value is spatially continuous when being used as spatial index.

2.3 Geometry Encoding

In a large 3D scene, the amount of vertices and triangular faces determines the storage size of a 3D model and undoubtedly have significant influences on network transmission and the rendering performance. Furthermore, Geographic coordinates are usually stored in double precision coding, which occupy the majority of storage for the whole model. Based on the reason above, this method segments each cube into 512 pixels along each axis of the octree subdivision and defines a new pixel coordinate inside each octree nodes as 512×512×512 grids, as is shown in Figure3. As a result, this technique generates a discrete representation to describe positions in the inner space of each node of the octree. Each vertex of the model will be attached to nearest isotonic unit of pixel. On the other hand, the edge length of any node is double that of its sub nodes at next depth. It means the ratio of pixels-to-meters of current depth, the scale, doubled from one to the next. Similar to the tile map, this encoding method lose different precision of position representation and geometric description at different scales. And this loss can be calculated according to a rigorous mathematic relationship between the octree node depth and the original extent of geographical space. These particular features of the method are favourable to progressive real-time transmission allowing users to visualize or apply analytical GIS methods to partially transmitted 3D model data sets. Models are not always stored in the same octree node and those crossing different cubes need to be split into parts. As a result, the geometry information of the original model separately in different sub models, allowing partial transmission of geometry data.

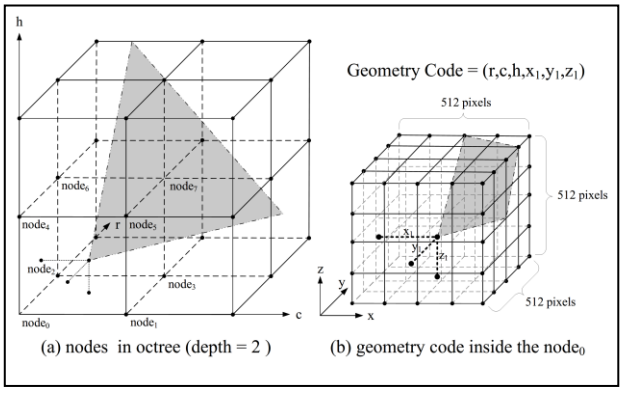

Figure 3. Geometry code of model's vertices inside the octree node

A problem that cannot be ignored is that, the geometry information may not be complete if a model crosses several octree nodes. To overcome this, every triangle face that intersects the boundary surfaces of octree node should be divided into a number of triangles or polygons. Afterwards, these polygons are triangulated by newly-added edges linking to grid boundary, which is supposed to ensure validity of mesh in each octree node. In the end, vertices on the boundary surfaces will be replaced after complete loading of their neighbouring octree nodes. Then shared edges need to be collapsed and duplicate vertices be removed, as is shown in Figure 4. Accordingly, the geometry information can be encoded with the combination of CICs and pixel coordinates as (r, c, h, x, y, z), for instance (2,2,2,100,132,56). The value of x, y and z must be an integral number and varies from 0 to 511. Since all vertices in one octree node have the same CIC value, every vertex in a split model need to store its pixel coordinates, (x, y, z), only and share the same CIC value recorded in the file. This approach helps to reduce a large amount of storage size in comparison with the original geographical coordinates in double precision.

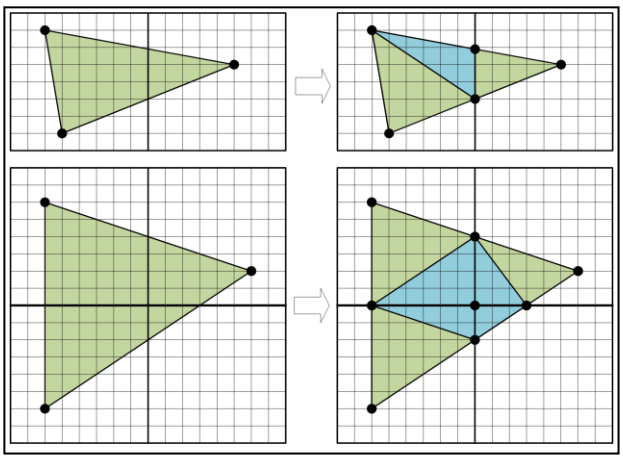

Figure 4. An overhead view of the triangulating process

2.4 Decoding

Decoding processing is a simple reverse of the encoding process. Firstly, duplicate vertices on edge surface of octree node, which were created during geometry encoding, should be removed. They have at least one geometric coordinates being equal to 512. So it is easy to find and fix. Secondly the original geographical coordinates of the octree node are calculated based on its depth, CIC and the predefined extent of the whole geographical space, which can be defined as

$$g_x = O_x + (c + \frac{x}{512}) \times \frac{range}{2^{depth\text{-}1}} \quad (9)$$

$$g_y = O_y + (r + \frac{y}{512}) \times \frac{range}{2^{depth\text{-}1}} \quad (10)$$

$$g_z = O_z + (h + \frac{z}{512}) \times \frac{range}{2^{depth\text{-}1}} \quad (11)$$

Where g_x is geographical x coordinate of the vertex, O_x is geographical coordinate of original point of the whole space, c is the first part of the CIC of current octree node, x represents the x value of the geometry code, range means the maximum edge length of the full extent, depth is the depth of the octree node, and g_y, g_z, O_y, O_z, y, z are defined correspondingly.

Thus, the geographic coordinate of vertices is recovered from CIC and Geometry coding. Meanwhile, the split meshes on edge can be merged on this basis.

3. RESULTS

The test data set contains 5 different kinds of geographic entities and scenes including building, Chinese ancient architecture, animal, city area and terrain. The tests are conducted on a 64-bit Intel Core i7-7700HQ 6M L3 Cache processor, 8 GB of RAM, Nvidia GTX 1050 Graphics. The algorithm was running on Windows10, Google Chrome v8 5.8.283. 3D dataset applied to assess the method follows the Three.js JSON model format3. The results reflected that this encoding method was able to describe geometry of geographical entities correctly and was conductive to geometry compression.

Bits per vertex (*bpv*) and compression rate(*cr*) were used to evaluate the geometry compression efficiency of this method

$$bpv = \frac{size\_vertex}{num\_vertex} \quad (12)$$

bpv is defined as bit size of whole vertexes, *size_vertex*, being divided by number of vertexes, *num_vertex*.

$$cr = \frac{size\_after}{size\_before} \times 100\% \quad (13)$$

Where size_before is bit size of whole vertexes before encoding and *size_ after* is bit size of whole vertexes after encoding.

TABLE I. THE ASSESSMENT OF THE ENCODNG METHOD IN DIFFERENT 3D GEOGRAPHICAL ENTITIES

(The table is attached at the end of the article)

In TABLE I, vc is short for the vertices count, fc for the triangle face count, size for the file storage size. The results revealed that this method was able to reduce vertex bit size by more than 90% in most models, and archived dramatic reduction in bpv. The time cost of the encoding was accredited while the time cost of decoding was fairly low and even browsers cannot record the decode time (replaced by "~"). The speed of encoding and decoding reflected a big advantage in applying this method to web 3D data loading. In addition, a good compress rate was obviously helpful to speed up the network transmission. It is noteworthy that this method performed better in dense 3D models, such as city area but played relatively worse in models with sparse vertices. Figure 5. illustrates that models were gradually simplified with the depth decreasing, and these simplified models were still able to maintain good geometry features after reducing vertices and triangle faces.

(This picture is attached at the end of the article)

Figure 5. details revealed by models at different depth

4. CONCLUSIONS

This paper proposed a compression encoding method of geographical coordinates in 3D space, and simplifications are adaptive to scales. A combination of the Cube Index Code of octree and Geometry Code inside the subspace of octree nodes is applied to represent 3D position. Hence this combination code can not only reduce the length of geographical coordinates but also simplify the geometry of model. The code implies the relationship between scale and geometry precision of geographical entities and is capable of supporting progressive network transmission.

By assessing the method on different models, this paper found this method was conductive to speed up the progressive transmission on the Internet and achieved a considerable compression rate in vertex bit size with a steerable precision loss. It shall be of positive meaning to web 3D map storing and transmission.

ACKNOWLEDGEMENTS

The research is supported by National Natural Science Foundation of China (NO.41571382); The authors express their great thanks to everyone who have helped in the writing of this article.

REFERENCES

Choe, S., Kim, J., Lee, H. and Lee, S., 2009. Random accessible mesh compression using mesh chartification. *IEEE Transactions on Visualization and Computer Graphics*, 15(1), pp. 160-173.

Dang, Q., Huang, F., Peng, W. and Chen, G., 2014. Design and Implementation of 3D Terrain Data Self-adaption Pyramid Technology. *Beijing Surveying and Mapping*(05), pp. 12-15.

De Fioriani, L., Magillo, P. and Puppo, E., 1997. Building and traversing a surface at variable resolution, Visualization'97. Visualization'97., Proceedings. IEEE, pp. 103-110.

Deering, M., 1995. Geometry compression, Proceedings of the 22nd annual conference on Computer graphics and interactive techniques. ACM, pp. 13-20.

Du, Y., Wu, Y., Wang, X. and You, X., 2006. Research on Pyramid Model for Global Multi-resolution Virtual Terrain Environment. *Journal of System Simulation*(04), pp. 955-958+967.

Du, Z., Zhu, Q. and Zhao, J., 2008. Perception-driven simplification methodology of 3D complex building models. *ISPRS2008*, Beijing.

Forberg, A., 2007. Generalization of 3D building data based on a scale-space approach. *ISPRS Journal of Photogrammetry and Remote Sensing*, 62(2), pp. 104-111.

Gong, J., Zhu, Q., Zhang, Y., Li, X. and Zhou, D., 2011. An Efficient 3D R-tree Extension Method Concerned with Levels of Detail. *Acta Geodaetica et Cartographica Sinica*(02), pp. 249-255.

Ha, M. and Yoo, H., 2016. Preprocessing using classification for lossless compression of 3D geometry data. *International Journal of Applied Engineering Research*, 11(13), pp. 7947-7950.

Hoppe, H., 1996. Progressive meshes, 23rd annual conference on Computer graphics and interactive techniques. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. ACM, pp. 99-108.

Hoppe, H., 1998. Smooth view-dependant level-of-detail control and its application to terrain rendering IClfl Proceedings of IEEE Visualization. *ResearchTrianglePark, WorthCarolina*, 3542.

Isenburg, M., Lindstrom, P. and Snoeyink, J., 2004. Lossless compression of floating-point geometry. *Computer-Aided Design and Applications*, 1(1-4), pp. 495-501.

Isenburg, M., Lindstrom, P. and Snoeyink, J., 2005. Lossless compression of predicted floating-point geometry. *Computer-Aided Design*, 37(8), pp. 869-877.

Jiong, Z., Jinxin, W., Shixue, L., Jing, Y. and Guangcheng, Z., 2017. Construction of large area true 3D geography scene based on sphere split bricks. *Science of Surveying and Mapping*(10), pp. 1-9.

Kada, M., 2008. Generalization of 3D building models for map-like presentations. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences: XXXVII.[S. l.]: ISPRS*, pp. 399-404.

Lindstrom, P. et al., 1996. Real-time, continuous level of detail rendering of height fields, 23rd annual conference on Computer graphics and interactive techniques. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. ACM, pp. 109-118.

Lixin, W. and Jieqing, Y., 2009. Global 3D-Grid Based on Sphere Degenerated Octree and Its Distortion Features. *Geography and GeoInformation Science*(01), pp. 1-4.

Luebke, D.P., 2003. Level of detail for 3D graphics. Morgan Kaufmann.

Ma, Z. and Li, H., 2016. Fast and effective compression for 3D dynamic scene based on KD-tree division. *Journal of Computer Applications*(09), pp. 2590-2596.

Mehra, R. et al., 2009. Abstraction of man-made shapes, ACM transactions on graphics (TOG). ACM transactions on graphics (TOG). ACM, pp. 137.

Peng, J. and Kuo, C.J., 2005. Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition. ACM Transactions on Graphics (TOG). ACM, pp. 609-616.

Taubin, G. and Rossignac, J., 1998. Geometric compression through topological surgery. *ACM Transactions on Graphics (TOG)*, 17(2), pp. 84-115.

Thiemann, F. and Sester, M., 2004. Segmentation of buildings for 3D-generalisation, ICA Workshop on generalisation and multiple representation, Leicester, UK. Proceedings of the ICA Workshop on generalisation and multiple representation, Leicester, UK, pp. 20-21.

Yue, Q., Yang, Q. and Cai, S., 2009. Three dimensional object modeling and coding compression method. *Science in China Series F-Information Sciences (in Chinese)*, 39(5), pp. 515-525.

Dataset name	*depth*	*Evaluation index*										
		before				*after*						
		bpv	*vc*	*fc*	*size/Bytes*	*bpv*	*vc*	*fc*	*size/Bytes*	*encode_time/ms*	*decode_time*	*cr*
Mansion	6	333.69	3,582	1,194	288,101	72.01	559	1,087	38,702	16	~	13.43%
	5					67.33	408	823	28,139	7	~	9.77%
	4					62.14	362	718	24,061	6	~	8.35%
Terrain	6	194.67	29,921	58,400	5,222,472	69.21	16,818	33,946	1,568,070	2,264	24	30.03%
	5					61.91	5,797	11,971	518,806	1,409	2	9.93%
	4					59.57	1,613	3,348	130,730	702	~	2.50%
City area	6	204.80	48,058	69,596	4,807,976	65.91	4,529	6,724	227,307	1,534	7	4.73%
	5					62.56	2,939	4,720	147,933	1,238	4	3.08%
	4					59.82	1,236	2,618	82,076	913	4	1.71%
Chinese temple	6	326.54	46,263	15,421	4,007,177	74.35	5,161	10,687	463,910	202	8	11.58%
	5					70.52	3,325	7,124	298,592	187	4	7.45%
	4					65.38	2,168	5,315	206,475	163	4	5.15%
bunny	6	203.88	2,813	5,146	304,166	72.04	2,588	5,146	207,742	41	~	68.30%
	5					68.65	2,572	5,114	205,317	28	~	67.50%
	4					63.83	1,978	4,001	157,275	27	~	51.71%

TABLE 1. THE ASSESSMENT OF THE ENCODNG METHOD IN DIFFERENT 3D GEOGRAPHICAL ENTITIES

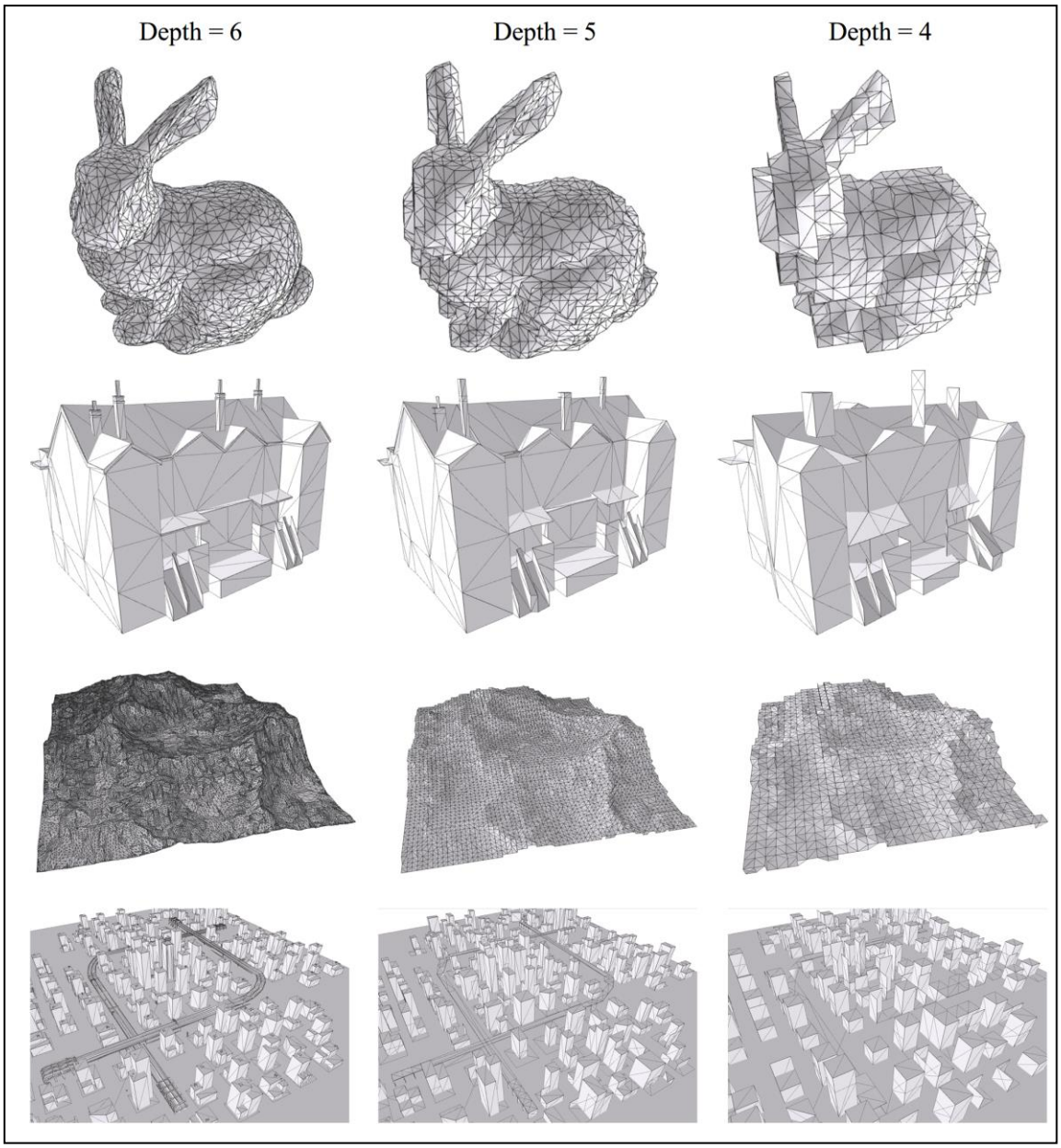

Figure 5. details revealed by models at different depth