ATLANTIS PRESS

Advances in Intelligent Systems Research, volume 159

International Conference on Mathematics, Modelling, Simulation and Algorithms (MMSA 2018)

Real-Time Rendering of Massive Geological Data

Zhijie Jia[1,2], Xianhai Meng[1,2,*], Zhongxiang Duan[1,2] and Qin Yang[1,2]

[1]School of Computer Science and Engineering, Beihang University (BUAA), Beijing 100191, PR China

[2]State Key Laboratory of Software Development Environment, Beijing 100191, PR China

[*]Corresponding author

***Abstract*—The visualization technique is an effective tool to extract useful information from the geological data. In this paper, we proposed an efficient visualization framework that can render large-scale geological truncated mesh in real-time. The geological models of different resolutions were generated by cell merging, and the rendering resolution was dynamically selected according to the viewpoint position. The experiments confirm the ability to handle massive geological data of our method, and the coarse resolution rendering only causes little detail loss for the visual image.**

Keywords—geological data; LOD; real-time rendering

I. INTRODUCTION

The analysis of the geological data is essential for the research and development of the crustal movement, the origin of life, the change of continents, the earthquake prediction and the mineral mining. Although the visualization technique is an important tool to extract the necessary information in geological models and many studies have been conducted on the visualization of the large-scale terrain data, it remains a difficult issue for the real-time rendering of massive geological data.

In order to efficiently exhibit the information of the large-scale data, the Level of detail (LOD) techniques are widely used in real-time rendering [1]. Lindstorm et al. proposed a continuous view-based level of detail model (CLOD), and their algorithm first divides the terrain file into several block files. These data blocks are organized and managed through quadtree and the model is simplified based on the screen error [2]. For the 3D spatial data, Li et al. proposed the multi-resolution data structure based on octree to organize the time series geospatial data [3].

In the field of massive data rendering, Lindstorm and Pascucci [4] [5] proposed out-of-core 3D terrain visualization framework while the method on GPU accelerating for large-scale terrain rendering was presented by Zhu et al. [6]. The entire terrain data is evenly divided into small pages, and real-time dispatching of large-scale terrain data is achieved through viewpoint-based preloading method. A page buffer pool management technique is devised to divide each page into sub-blocks for batch rendering. The cracking process between sub-blocks at different levels is carried out in the preprocessing stage. Asirvatham and Hoppe [7] apply the main idea of geometry Clipmap algorithm to the simplification strategy. The simplification algorithms are aimed to simplify the triangular meshes. Typically, the real-time optimally adapting meshes algorithm (ROAM) [8] are used to achieve real-time simplification with binary tree. It is a top-down strategy that simplifies mesh by merging triangular queues and refines by splitting triangular queues. Hoppe [9] proposed a View-dependent refinement of progressive meshes algorithm (VDPM) which simplifies the original mesh to a simple mesh by using the edge-folding operation, and generates several continuous LOD models.

Geological data has the characteristics of huge capacity and spatial diversity. These features make it different from terrain data. Unlike terrain data, most of which are surface grids, the geological data is represented by volume grid. Therefore, the multi-resolution simplified model of the volume grid is demanded.

This paper presents an efficient framework for the real-time rendering of massive geological data. According to the characteristics of geological data, we adopt the geological model of truncated grid and propose an algorithm to simplify the polyhedron grid. In the preprocessing stage, the data is divided into blocks, and the quadtree multi-resolution simplified model files are generated. At the same time, the sight distances of the simplified models at all levels are automatically calculated. In the rendering stage, we use the memory page loading of OpenSceneGraph (OSG) to render in real-time according to the viewpoint in order to display the panoramic view with the coarsening models and render in details with the fine models.

II. REAL-TIME RENDERING OF MASSIVE GEOLOGICAL DATA

A. Geological Data Model

Because of the spatial features of geological data, the DEM (Digital Elevation Model), which is commonly used to represent topographic data, is not suitable to geological data. In this paper, we intend to use the truncated grid to organize geological data based on its features.

As shown in Figure 1, the truncated grid is composed of grid cells that are generated by the vertically cutting of the subdivision layers. The stored information of truncated grid can divide into the following parts: (1) The geometry part is used to store the geometric data files, including the information such as vertices, faces and cells. Each face in the truncated grid contains the indexes of its vertices and each cell stores the index of its faces; (2) The cell part is used to store attribute files related to the grid cell; (3) The face section is used to store attribute files related to the grid face; (4) The framework part is used to store the information of model framework, such as

various geological interfaces. Every cell of the truncated grid is a polyhedron and has its index of the **x, y, z** axes. The cell also contains the information of its related strata and faults. According to the fault and strata, the whole geological model can uniquely identify some part of the area, which is regarded as the connected region. The aim of this paper is to rendering these truncated grids in real-time, and the main issues are to build the multi-resolution coarsening models for them. Besides, the memory management should be also considered as the multi-resolution data is too huge and difficult to completely load into memory in one time.

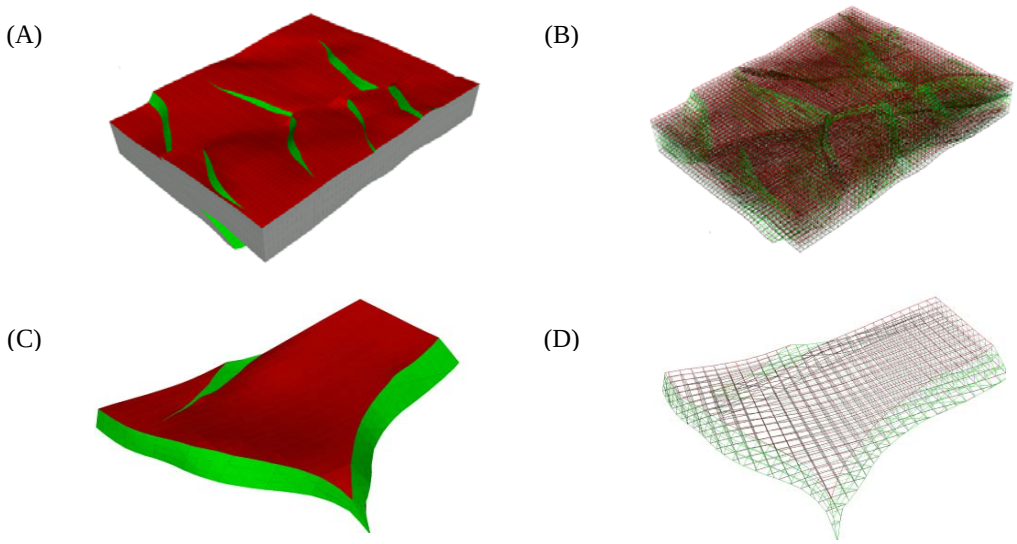


FIGURE I. TRUNCATED GRID OF GEOLOGICAL MODEL. (A) THE WHOLE GEOLOGICAL MODEL. (B) THE TRUNCATED GRID OF THE WHOLE GEOLOGICAL MODEL. (C) ONE GEOLOGICAL BLOCK. (D) THE TRUNCATED GRID FOR ONE GEOLOGICAL BLOCK.

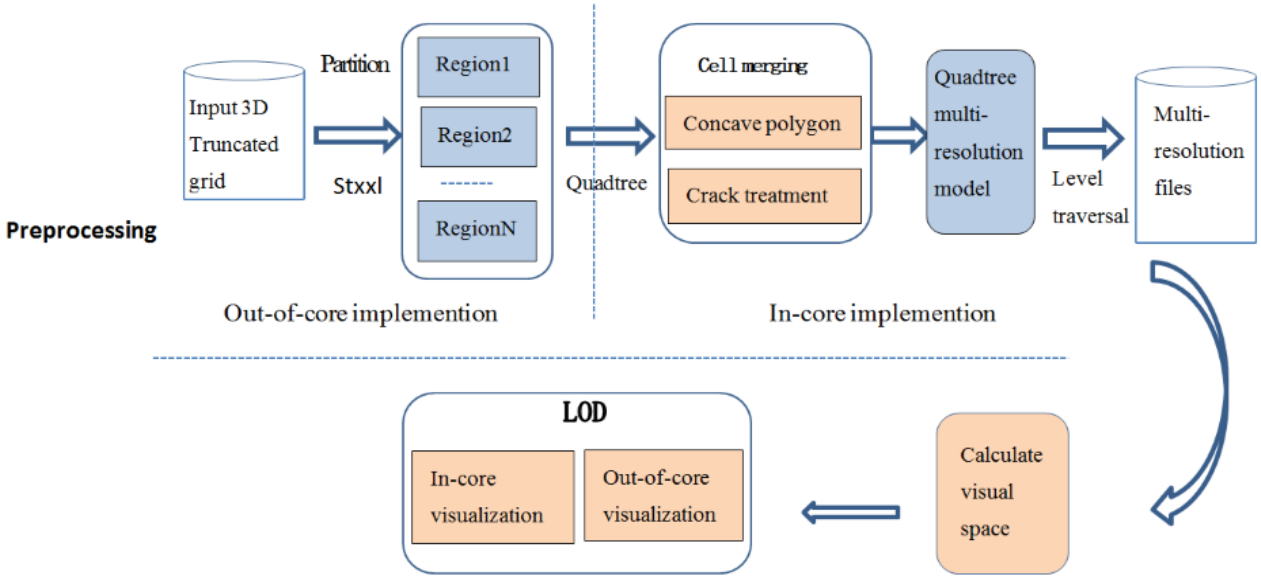


FIGURE II. THE RENDERING FRAMEWORK FOR MASSIVE GEOLOGICAL DATA.

B. Design of Rendering Framework

The overview of our real-time rendering framework for massive geological data is shown in Figure. 2. At first, we construct the multi-resolution simplified models according to the characteristics of truncated grid, and then generate the multi-resolution model files. In the rendering stage, we automatically control the number of the rendering elements according to the position of the viewpoint and the view frustum and dynamically load the model files into the memory. The following content of this paper explains the key issues of our method in details.

1) The Reading and Storage of Massive Geological Data

As to The construction of the multi-resolution model, several issues have to be considered. At first, the quantity of the original data is too huge to be loaded into memory all at once. Secondly, different geological regions can not be merged. Besides, in order to reduce the size of the generated model files, the original data and its index should also be partitioned according to the geological region. However, the truncated grid does not store grid cells in the order of spatial distribution, which makes it impossible to directly block the original files according to the file size.

To solve the above problems, we first employ an implementation of the C++ standard template library called STXXL to read and store all the geological data on the external memory [10]. The template library STXXL includes the algorithms for the external memory (out-of-core) computations and is very suitable to process huge volume data stored only on disks. Subsequently, the original geological data can be partitioned through the division of vertex set. In the meantime, the indexes of the faces and cells are also set according to the segmentation result of the vertices. In this way, we can store and process the geological data of large-scale and ensure that the multi-resolution model files will not be too large at the later stage, which can reduce memory usage and IO time in the mid of rendering.

2) The Multi-resolution Mesh Generation by Cell Merging

Most of the truncated cells are regular hexahedra and the ones near the boundary are irregular polyhedrons. For each cell, there are at most four faces with certain normals. In most cases, the geological models we discuss exhibit considerably greater horizontal scale than vertical. In order to evenly partition the data, we propose to use the quadtree, instead of octree, to organize the geological model and generate the multi-resolution mesh in this section. The quadtree is constructed according to the cell indexes along the **x** and **y** axis. The original cells are distributed in the leaf nodes of the quadtree, and we design an algorithm to merge the original cells to generate the cells of coarse resolution stored in the parent nodes. In our algorithm, we merge two cells by eliminating their coincident faces. The rest of the faces of the two cells then form into a new cell by deleting the coincident points and edges. However, in the process of cell merging, two issues need to be seriously considered:

a) Crack Treatment: Merging two faces of adjacent cells may affect other faces in the grids because the shared vertices and edges. The deletion of the shared vertices and edges may lead to cracks in the mesh. These cracks may also appear between adjacent nodes of the same level (Figure 3). In order to avoid the cracks, we attempt to store the related faces of each edge and efficiently check the vertices and edges. If the vertices and edges affected by face merging are still shared by other cells, they will not be deleted.

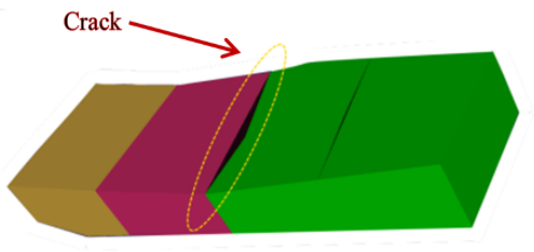


FIGURE III. THE CRACK BETWEEN TWO CELLS.

b) Concave Polygons: The merging of two cells may be accompanied with faces of concave polygons generated that can not be correctly rendered. Actually, every concave polygon should consist of at least one concave vertex. In order to avoid rendering the concave primitives, the faces containing concave vertices, which is recognized as the concave polygon faces, will be represented by the polygons of the previous faces. The concave polygon faces may be vanished after the further face merging to generate the cells of coarser resolution.

3) Rendering the Cells in Differents Resolutions

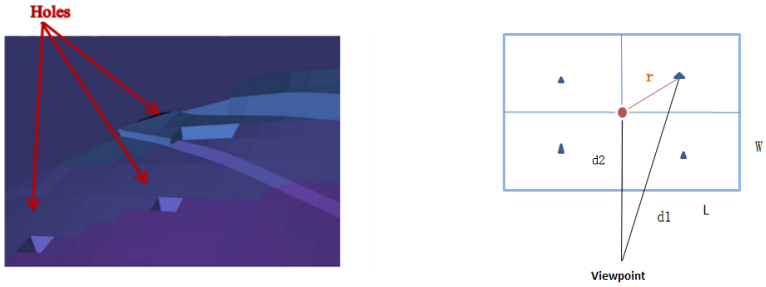
The rendering of coarse resolution will result in the image of poor quality while the detailed rendering may have to handle too many cells and slow down the visualization performance. The key of LOD visualization is to choose the rendering cells of appropriate resolution according to the viewpoint position. In this section, we intend to select the rendering resolution of different levels according to the distance between viewpoint and the center of the bounding box.

As shown in Figure 4, when rendering relatively fine data, some nodes are not displayed as the centers of the bounding box for the parent nodes and the child nodes are at different positions, and the adjacent nodes are not in the same resolution level. To fix this problem, we intend to make the original and generated cells belong to different continuous resolution levels. For a single cell, assume that the rendering resolution level which it is belong to range from R_{min} to R_{max}. Let L and W be the length and width of the bounding box, for one node of the quadtree, consider $S(h)$ and $r(h)$ that satisfy

$$S(h) = \alpha \times 2^h Max(L,W), \alpha \in [8,15]. \quad (1)$$

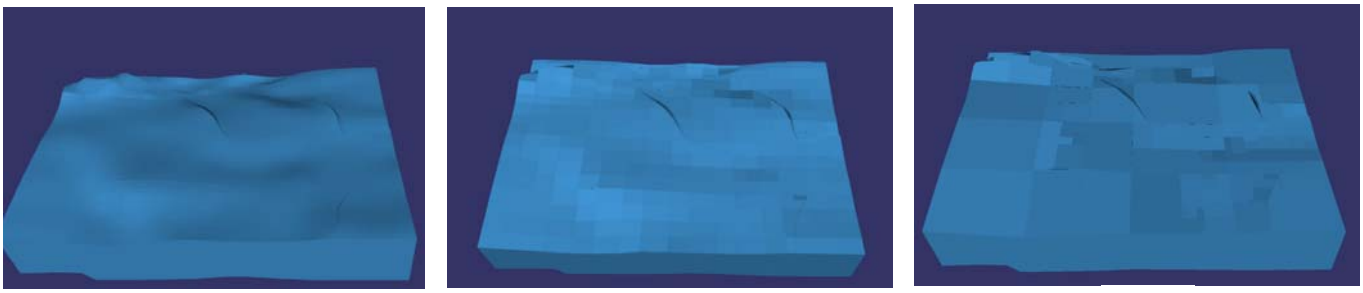
$$r(h) = \lambda \times 2^h \sqrt{L^2 + W^2}/2, \lambda \in [2,3]. \quad (2)$$

In the formulas above, h is the height of node. As to the cells of leaf nodes, R_{min} =0 to R_{max} =S (0), and R_{min} = S (h) - r (h) to R_{max} =S (h) for the intermediate nodes. We set the $\alpha \in$ [8,15] and $\lambda \in$ [2,3] to avoid the cells that are failed to display and prevent superabundant cells in the view frustum. There will be no rendering holes as the adjacent nodes share a few of belonging rendering resolution levels.



(A) (B)

FIGURE IV. FIX THE HOLES IN MULTI-RESOLUTION RENDERING. (A) THE HOLES WHEN RENDERING THE FINE DATA. (B) THE VIEW SHOWS THE CENTERS FOR ADJACENT NODES AT DIFFERENT POSITIONS.



(A) (B) (C)

FIGURE V. RENDERING THE TRUNCATED GRID IN DIFFERENT RESOLUTIONS. (A) RENDERING THE ORIGINAL TRUNCATED MESH. (B) RENDERING THE MESH AFTER COARSENING TWO TIMES. (C) RENDERING THE MESH AFTER COARSENING FOUR TIMES.

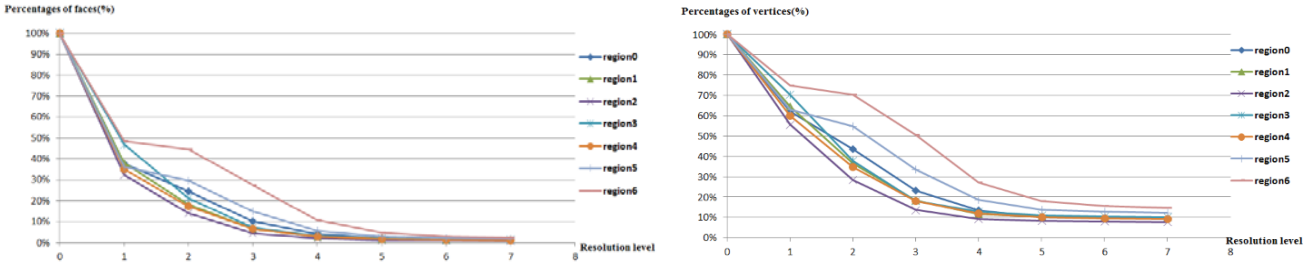


FIGURE VI. THE PERCENTAGES OF FACES AND VERTICES OF DIFFERENT RESOLUTION LEVELS COMPARED TO ORIGINAL MESH.

III. IMPLEMENTATION AND RESULTS

We implement our algorithm to do visualization for massive geological data by employing the 3D graphics toolkit called OpenSceneGraph (OSG), and the OSG nodes known as PagedLOD are used to do dynamic scheduling for the rendering nodes stored in the external memory [11]. The experimental environment is the desktop PC with the Intel(R) Core(TM) i5-3470 CPU, 4GB memory and the NVIDIA GeForce GT 630 graphics card, and the test data contains multiple geological regions. The cell number of truncated grid for rendering is 72,360, and the number of faces is 222,980.

A. The validity of Cell Merging

The overall view of different resolution levels for truncated grid are shown in Figure 5, and the percentages of the vertices and faces of different resolutions, compared to the original mesh in each geologic region, are shown in Figure 6. It is shown that our method can significantly reduce the vertices and primitives after the mesh coarsening by cell merging for different rendering levels, which means the coarsening will notably improve the rendering performance. In addition, the cell merging causes little shape changing for the panorama view.

In the mid of real-time rendering, the rendering resolution is dynamically adjusted according to the viewpoint position. In Figure 7, we render the primitives of different levels in different colors. As we can see in the figure, there will be more detailed rendering as the viewpoint approach the geological model. Figure 8 shows the frame rates when rendering in different resolutions. The frame rates are always above 25 fps, which meet the requirements for real-time rendering.

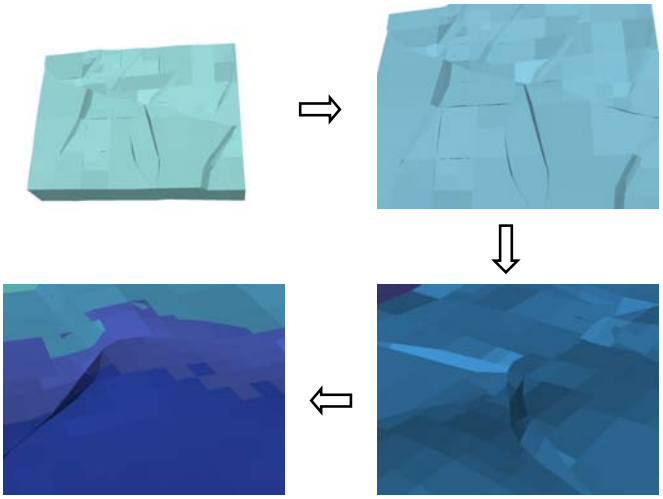


FIGURE VII. RENDERING THE TRUNCATED GRID IN DIFFERENT DETAILS . THE CELL COLORS DEEPEN WITH MORE DETAILED RENDERING.

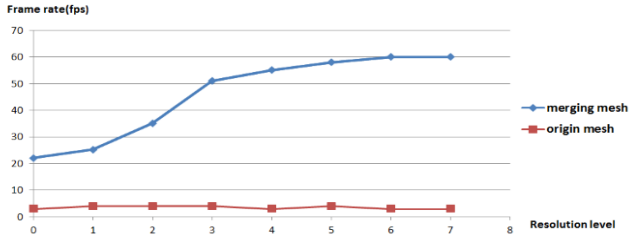


FIGURE VIII. FRAME RATE STATISTICS OF GEOLOGICAL DATA RENDERING.

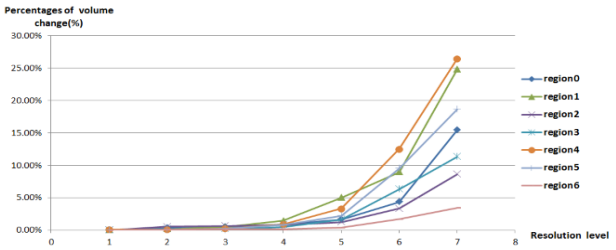


FIGURE IX. ERROR MEASUREMENTS ON MESH COARSENING OF GEOLOGICAL DATA BY CALCULATING THE VOLUME CHANGE.

B. *The Error Measurements on Mesh Coarsening*

There are two ways of error measurements to estimate the mesh coarsening result. One is to measure the appearance similarities according to the differences between the rendering images, and another is to estimate the geometry similarities for mesh of different resolutions. In our rendering framework, we evaluate the volume changes of the geological model to estimate the shape differences for the rendering primitives of different resolutions. Compared to the original truncated grid, the volume differences for the coarsening mesh of different resolutions are shown in Figure 9. As we can see in the figure, the volume changes at most 3% compared to the original shape of the geological model when the coarsening operations are conducted less than 4 times. The also indicates our method make little change for the shape of the geological model by mesh coarsening.

IV. CONCLUSION AND FUTURE WORK

In this paper, we introduce an efficient real-time rendering framework for the massive geological data. The algorithm to simplify and merge cells is designed in the framework to generate the multi-resolution files for the geological models represented by truncated mesh, and the geological models of multi-resolution generated by our algorithm witness only a slight shape changing. The technique of paging scheduling for external memory is employed to process the geological data of large-scale, and the rendering performance is improved by automatically calculating the visual space. The experimental result shows that our method generates valid cells of multi-resolution and performs well in the handling of massive geological data.

ACKNOWLEDGMENT

National Natural Science Foundation of China (Grant 61003110) and Fund of the State Key Laboratory of Software Development Environment (Grant SKLSDE-2010ZX-10) jointly supported this work.

REFERENCES

[1] T. T. Brunyé, H. A. Taylor, and M. Worboys, "Levels of Detail in Descriptions and Depictions of Geographic Space," Spatial Cognition & Computation, vol. 7, pp. 227-266, 2007/09/13 2007.

[2] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, and G. A. Turner, "Real-time, continuous level of detail rendering of height fields," presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996.

[3] J. Li, H. Wu, C. Yang, D. W. Wong, and J. Xie, "Visualizing dynamic geosciences phenomena using an octree-based view-dependent LOD strategy within virtual globes," Computers & Geosciences, vol. 37, pp. 1295-1302, 2011/09/01/ 2011.

[4] P. Lindstrom and V. Pascucci, "Visualization of large terrains made easy," presented at the Proceedings of the conference on Visualization '01, San Diego, California, 2001.

[5] P. Lindstrom and V. Pascucci, "Terrain simplification simplified: a general framework for view-dependent out-of-core visualization," IEEE Transactions on Visualization and Computer Graphics, vol. 8, pp. 239-254, 2002.

[6] Y. Zhu, X. Shi, and X. Li, "A Method for Large-Scale Terrain Rendering Based-on GPU," in 2010 Second International Conference on Multimedia and Information Technology, 2010, pp. 211-214.

[7] A. Asirvatham, "Terrain rendering using GPU-based geometry clipmaps," GPU Gems 2, pp. 27-45, 2005.

[8] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. C. Miller, C. Aldrich, and M. B. Mineev-Weinstein, "ROAMing terrain: real-time optimally adapting meshes," presented at the Proceedings of the 8th conference on Visualization '97, Phoenix, Arizona, USA, 1997.

[9] H. Hoppe, "View-dependent refinement of progressive meshes," presented at the Proceedings of the 24th annual conference on Computer graphics and interactive techniques, 1997.

[10] R. Dementiev, L. Kettner, and P. Sanders, "Stxxl: Standard Template Library for XXL Data Sets," in Algorithms – ESA 2005, Berlin, Heidelberg, 2005, pp. 640-651.

[11] L. I. Xiao-Long and J. F. Cao, "Research on the Technology of Multi-dimensional Dynamic Visualization of Sub-marine Oil Spill Based on OSG," Coastal Engineering, 2015.