Progressive Streaming and Rendering of 3D Terrain for Cyber City Visualization

Fuan Tsai[1], Hau-Shiung Liu[1], Jim-Kim Liu[2], Kuo-Hsing Hsiao[2]
[1]Center for Space and Remote Sensing Research,
National Central University, Jhongli, Taiwan
[2]Energy and Resources Laboratory, ITRI, Hsin-Chu, Taiwan
ftsai@csrsr.ncu.edu.tw , 943202070@cc.ncu.edu.tw

Keywords: progressive streaming, 3D terrain model, visualization, cyber city

Abstract:

Cyber city is a system that allows users to inquire about real world information quickly and conveniently on the internet. Terrain is one of the most important factors in the rendering of virtual scenes regarding cyber city visualization. For large-area and detailed representations of digital terrain models, rendering in real-time will be difficult because of the large data volume. It is also difficult to smoothly and efficiently broadcast virtual environments in large-scale web applications.

To address these issues, this paper presents an efficient technique to progressively transmit and smoothly rebuild cyber city terrain models for internet-based geoinformatic applications. The full terrain was divided into regular tiles to stream adaptively. Different levels of detail of tiles were created and saved in the serve end and transmitted to clients according to different visualization conditions. The first task for progressive transmission was to transmit the base terrain data. Secondly, the LOD of each visible tile was determined based on viewing parameters in real time. Detail vectors describing changes of the LOD for each tile were then constructed and sent to the client to modify the developed mesh representation of the terrain. Algorithms for the decomposition and reconstruction of subdivision surfaces of tiles were also developed to update each tile as view-dependent parameters changed, so the clients could have smooth scene rendering. With this technique, users can reconstruct smooth approximations of the original scenes adaptively from a rather small amount of data received on the internet to increase the efficiency of terrain rendering in cyber city visualization.

1. Introduction

In response to the development of 3D spatial information for urban planning and management, cyber city modeling is getting important. If users could obtain 3D information they want, the quality and efficiency of management will increase. It is convenient for users if they can search the information in a realistic cyber city model on internet. To implement a cyber city, terrain model is one of the most important factors. For a large area with complex terrains, a significant number of meshes will be needed in order to correctly describe the scene. For internet applications, the large amount of data will be difficult to transmit efficiently.

Level of Detail (LoD) (James, 1976) is a method to reduce the volume of data transmitted. Different "levels" will be constructed from the same digital terrain model (DTM). The LoD of different areas will be decided based on their roughness or other criteria. It could reduce the volume of data effectively because the suitable number of meshes will be determined. View Frustum culling (Coorg and Teller, 1997) is another method to reduce data. In this approach, DTM data will be separated into regular blocks. Only visible blocks need to be transmitted.

For internet applications, another issue is streaming transmission. Streaming is a method that let users gain information even though the transmission is not completed (Hoppe, 1998). It first broadcasts the rough LoD (called base block) for the blocks. After the base blocks transmission is completed, the system checks each block to determine if the LoD is suitable. If not, detail vectors (Labsik et al., 2000) that describe changes between two LoDs will be transmitted to upgrade the LoD of these blocks. Repeat this step until the LoD of all blocks are adaptive. This study integrated these algorithms to develop a progressive streaming system for upgrading LoD of terrain blocks for smooth terrain visualization in a server-client environment.

2. Overview of the methods

The developed system can be separated into two parts. The first was the streaming of terrain data as well as its management. This part focused on pre-processing terrain data on the server side. The second part was dedicated to the adaptive rendering of terrain blocks on the client side.

In order to perform the terrain transmission and management, a tiling system (Pouderoux and Marvie, 2005) was employed. The database was generated once by subdividing the full DTM. Then six levels of detail of each block were constructed. For practical reasons and testing purpose, terrain tiles were encoded into separated files instead of committed into a database system. The workflow of data pre-process is shown in Fig. 1.

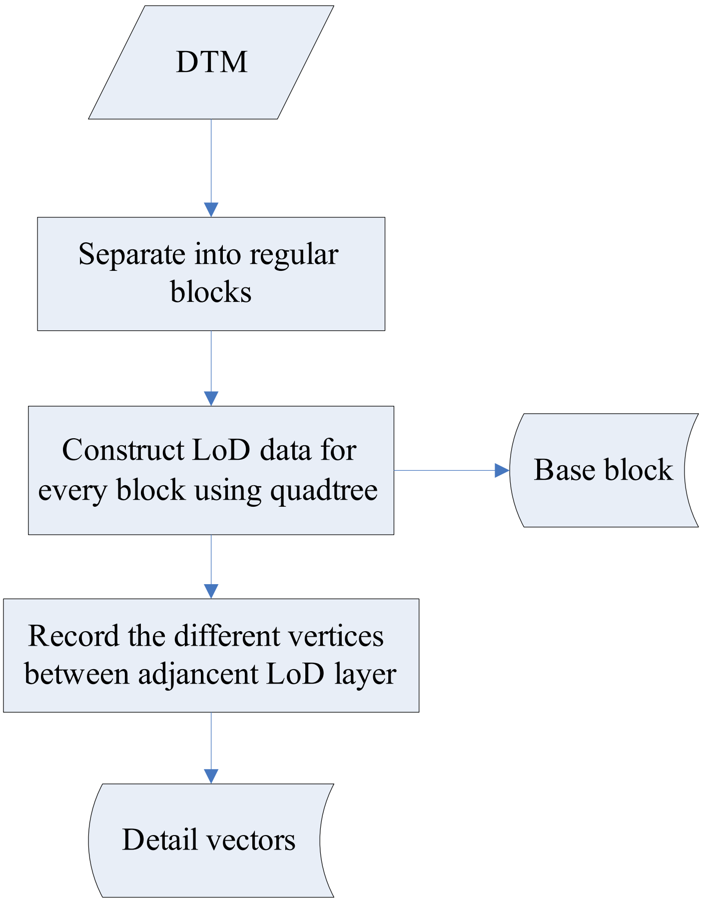

Figure 1. Workflow of data pre-processing

The adaptive rendering of visible area was performed through the use of multi-resolution tiles. Before rendering of each frame, for each block, its suitable LoD was determined. The base block data were broadcasted first, and then detail vectors were transmitted if needed. A block can be rendered through coarse to fine. In this research, OpenDX (http://www.opendx.org) was used as the render engine. Terrain models were rendered using constructed triangulations. The workflow of rendering is shown in Fig. 2.

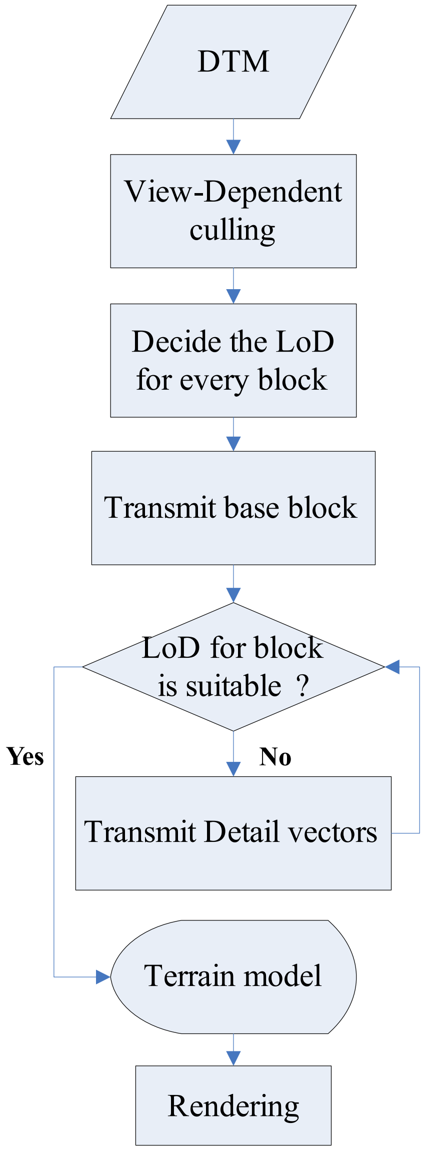

Figure 2. Workflow of rendering

3. LoD generation

First, the DTM data were separated into regular blocks. Then six levels of detail layers were constructed using quad-tree algorithm (Samet, 1984) based on the roughness of each block. Based on a pre-defined threshold, the DTM was subdivided into four blocks. The subdivision continued until each cell was within the threshold.

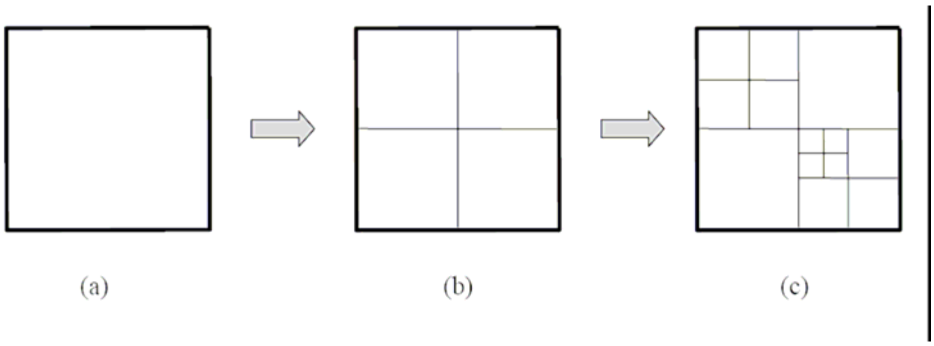

Figure 3. Quadtree process

4. Rendering

On the client side, visible blocks were determined according to view frustum. Fig. 4

illustrates the concept of view frustum. Base-block data of visible blocks were transmitted first. Then, LoD of each block was checked. If the level was not suited, detail vectors were transmitted till the LoD of each block was adaptive.

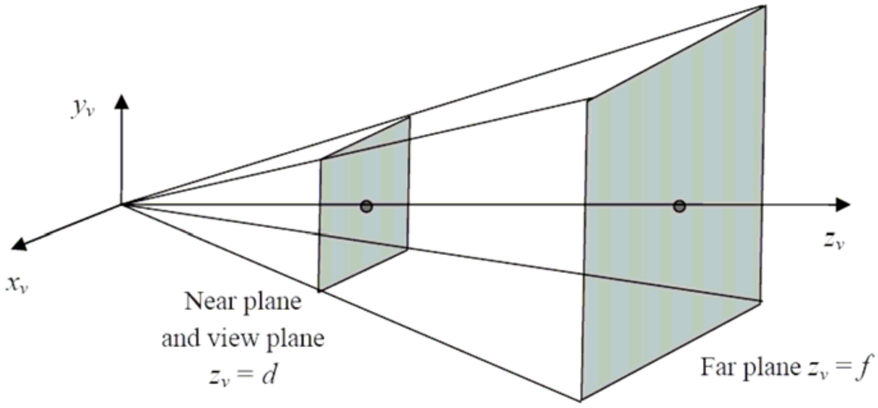

Figure 4. View-frustum

Detail vectors record the information about changes between adjacent LoD layers that rendering needs. They were used to reconstruct the terrain model in the client end. The structure of detail vectors is listed in Figure 5. In the rendering mechanism, triangulations were used to prevent the cracks and make the change of scenes smooth (Fig. 6).

```
DetailVectors{
    blockIndex []
    lodIndex []
    addPts []
    adjIndex []
    rmMesh []
}
```

Figure 5. Data structure of detail vectors

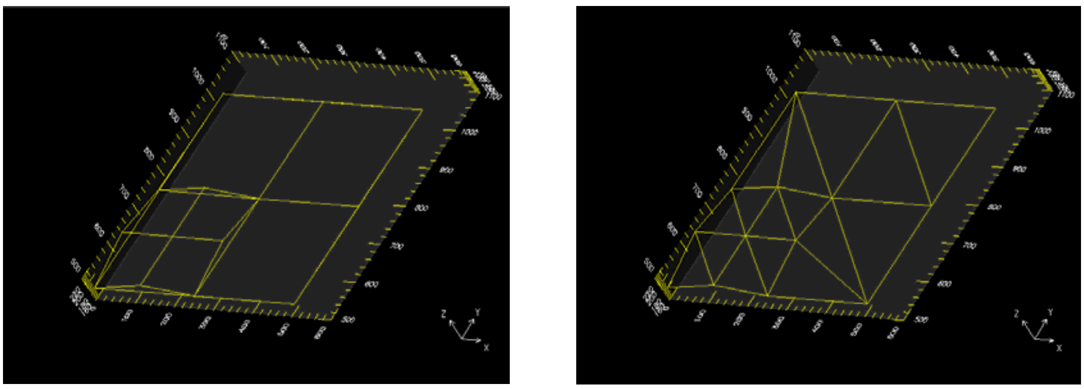

Figure 6. Quadrate mesh (left) and triangulate mesh (right) for the same area

5. Experiment and Results

A few experimental results are presented in this section to demonstrate the developed progressive transmission and rendering system.

A DTM of Tai-Chung city in central Taiwan was used as the study material. The original DTM was grid data of 0.5 meter resolution. The size was 4097×8193 pixels. In this research, the

DTM was divided into 8×16 regular blocks. Each block was 513×513 pixels.

Table 1 shows the threshold for each LoD. Table 2 shows the number of points in each LoD and its storage volume.

Table 1. threshold for quadtree

Level	0	1	2	3	4	5
Threshold (m)	8.0	6.5	5.0	3.5	2.0	0.5

Table 2. amount points of every LoD

Level	0	1	2	3	4	5
Number of pts	92752	154632	290812	637462	1571902	5357502
Volume of storage (MB)	4.2	7.2	13.9	31.4	79.6	278.1

If the origin data were used to render terrain, a huge amount of data would need to be transmitted to users. It would be time consuming, and users would not be able to gain information in real time. On the other hand, if detail vectors were used, it could reduce the volume of transmitted data. Table 3 shows the volume of storage for detail vectors.

Table 3. detail vectors of every LoD

Level	0	1	2	3	4	5
Number of pts	nan	61880	136180	346650	934440	3785600
Volume of storage (MB)	4.2	3	6.7	17.5	48.2	198.5

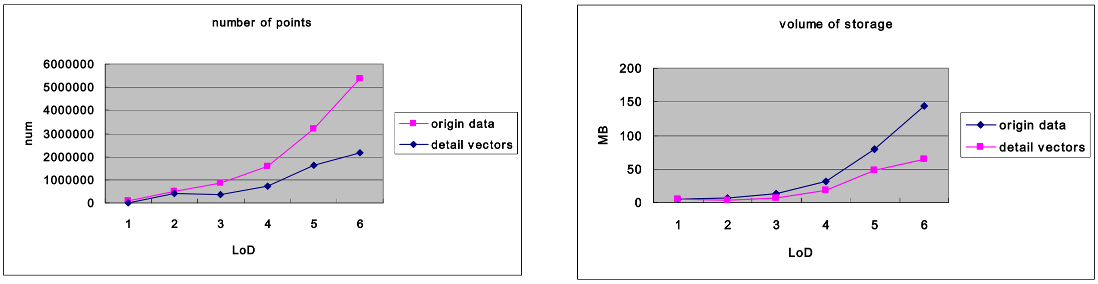

Figure 7. The number of points of each LoD layers (left), and the volume of storage of each LoD layers. Red line shows the origin data and blue line shows detail vectors data.

6. Conclusions

This paper presents a solution that allows users to obtain information for cyber city terrain visualization in almost real-time. Experiments demonstrated that the developed progressive transmission and rendering was more efficient than traditional methods. The developed method produced rendering smoothly. Future work will focus on the progressive texture mapping for terrain visualization.

ACKNOWLEDGMENTS

This study was supported in part by the National Science Council of Taiwan under project No. NSC 95-2221-E-008-104-MY2, and NCU-ITRI Union Research and Developing Center under project No. NCU-ITRI-960304.

Reference:

Coorg S., S. Teller, 1997, "Real-time occlusion culling for models with large occluders", *Proceedings of the 1997 symposium on Interactive 3D graphics*.

Hoppe, H., 1998, "Smooth View-Dependent Level-of-Detail Control and its Application to Terrain Rendering.", *IEEE Visualization*, pp.35-42.

James, H. C., 1976, "Hierarchical geometric models for visible surface algorithms.", *Communications of the ACM*, Vol. 19, pp. 547-554.

Ji J., S. Li, X. Liu, E. Wu, 2004, “P-Quadtrees: A Point and Polygon Hybrid Multi-resolution Rendering Approach”, *Computer Graphics International.*

Labsik U., L Kobbelt, R Schneider, HP Seidel, 2000, “Progressive transmission of subdivision surfaces”, *Computational Geometry: Theory and Applications, 2000 – Elsevier.*

Pouderoux J., JE Marvie, 2005, “Adaptive Streaming and Rendering of Large Terrains using Strip Masks.”, *In Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques.*