A New Method for Dynamic-Loading Large Terrain Dataset Visualization in Flight Simulation

Xie Jianbin, Liu Tong, Zhuang Zhaowen
School of Electronics Science and Engineering of NUDT, ChangSha 410073, China
E-mail: jbxie@126.com

Wang Jinyan, He Yizheng
China Aeronautical Radio Electronics Research Institute, ShangHai 200030, China

Abstract

Large terrain dataset visualization is very difficult in flight simulation, because the size of terrain dataset is orders of magnitude larger than the main memory of system in flight simulation. The dataset are usually stored out-of-core. The relatively slow access speed of out-of-core makes data retrieval the bottleneck of the terrain visualization in flight simulation. The previous works on large terrain dataset visualization focus on two aspects: simplifying scene and organizing data. They always organize the terrain dataset on the disks, then load them into main memory part by part and render them efficiently. However, organization of terrain data on disk is quite difficult because the error, the triangulation dependency and the spatial location of each vertex all need to be considered, and the frame deadlock or skip phenomenon will occur probably.

This paper proposes a new method for large dataset loading and rending in flight simulation. This method presents a new concept called sub-dataset, which are the all data to render the terrain in a frame. By loading the sub-dataset into main memory and real-time render them, we can realize the large terrain dataset visualization in flight simulation. Virtually, the terrain dataset refer to the longitude and latitude (L/L) coordinates of the flight position. By means of the index of L/L information, we can find the sub-dataset we need to load and render. The new method is very simple, and it costs much less time, provides visual continuity, and obtains fast visualization and animation of large scale terrain in flight simulation.

1. Introduction

Terrain visualization plays an essential role in flight simulation. In flight simulation, terrain data is obtained from the natural environment. The size of the data is usually very large. For instance, the US Geology Survey provides digital elevation data covering most parts of the United States, sampled at 10 or 30-meter resolution. The volume of the entire data set is measured in terabytes. In this situation, the size of terrain database is too large to be loaded into main memory one time. Therefore, large terrain dataset are usually stored out-of-core, loaded into main memory part by part, then rendered efficiently. Many algorithms have been proposed to visualize terrain data at interactive frame rates. Previous algorithms mainly focus on two aspects: simplifying scene and organizing data. Simplifying scene means improving the performance of large terrain visualization by replacing the original model with a simplified one. The simplified model can construct the same scene as original one. Organizing data means using an effective method to organize terrain data on disk, and then load and render the data directly. The organized data can improve coherence and reduce the number of paging events from external storage to main memory. Organization of terrain data on disk is quite difficult because the error, the triangulation dependency and the spatial location of each vertex all need to be considered. Previous terrain clustering algorithms did not consider the per-vertex approximation error of individual terrain data sets. Therefore, the vertex sequences on disk are exactly the same for any terrain [8].

In flight simulation, the software and hardware resource is relatively limited. In this situation, large terrain dataset visualization is more difficulty. The algorithm for terrain dataset loading, organization, and rendering should be more effective and fast.

This paper proposes a new method for dynamic-loading large terrain dataset visualization in flight simulation. In previous works, they usually organized the large terrain dataset on the disk firstly, and then loaded the needed data into main memory to render. In our approach, we load the needed data into main

0-7695-3065-6/07 $25.00 © 2007 IEEE
DOI 10.1109/DMAMH.2007.32

memory firstly, and then render them directly. We present a new concept called sub-dataset. The sub-dataset can be loaded into main memory rapidly by means of the index of L/L information of the present flight position. After loading sub-dataset, we can render them directly. The new method is very simple, and it costs much less time, provides visual continuity, and obtains fast visualization and animation of large scale terrain in flight simulation. The experiments show that the new method requires less calculation, can rapidly manages the sub-missions and renders smoothly without visual delay. In addition, the method is easy to transplant on different platforms.

2. Previous works

In [1] a view-dependent multi-resolution modeling combined with a dynamic loading technique for large-scaled terrain model visualization was proposed. A large-scaled terrain model was partitioned into blocks and then dynamically loaded for browsing. In [4] a simple method to compute index that yielded substantial improvements in locality and speed over more commonly used data layouts was proposed. In [6] a new algorithm of restricted quadtree triangulation, which included among others exact error approximation, progressive meshing, performance enhancements and spatial access, was proposed to describe an all-in-one visualization system which integrates adaptive triangulation, dynamic scene management and spatial data handling. In [7] regular-grid tiles were proposed as the payload data per diamond for both geometry and texture. The use of 4-8 grid refinement and coarsening schemes allowed level-of-detail transitions that were twice as gradual as traditional quadtree-based hierarchies, as well as very high-quality low-pass filtering compared to sub-sampling-based hierarchies. In [8] a novel clustering algorithm was proposed which introduced the level-of-detail (LOD) information to terrain data organization to map multi-resolution terrain data to external memory. In [9] a new method based on the bintree to organize the terrain data was proposed. In [10] a framework for the stereoscopic view-dependent visualization of large scale terrain models was presented. In [11] an efficient method to automatically compute a smooth approximation of large functional scattered dataset given over arbitrarily shaped planar domains was presented. In [12] a novel multi-resolution triangular mesh called direct mesh that is designed specifically for secondary storage was proposed, which supported available indexing methods natively and required no modification to multi-resolution triangular mesh structure.

3. Sub-dataset

The general digital elevation model (DEM) dataset of the terrain is a regular grid stored in a file format. Every single data file is a real description of a terrain and is called a dataset. So the real terrain is made up of many terrain blocks, which are corresponding to dataset and put together to form a terrain. But the display window only shows a small part of the terrain block every frame. The DEM data which corresponds to the small part of the terrain block to be showed on the display window is called a sub-dataset. According to this definition, the dataset include the all data in the DEM data file, the sub-dataset only contains the needed terrain data o be showed on the display window.

The terrain data file stores the DEM data, and also stores the reference position L/L of the terrain block corresponding to the dataset. Therefore, the DEM data and the terrain L/L information have a one-to-one relationship.

Once the present position L/L information is obtained from the Global Position System (GPS), the DEM dataset used at present can be decided. Further, the DEM sub-dataset needed at the presser frame for rendering can also be decided. Through rendering the corresponding sub-dataset, the rendering of the large scale DEM data will be realized simply and rapidly.

4. The corresponding relationship of the dataset to the present L/L information

The mathematic relationship between the three-dimension rectangular coordinates (x, y, z) and the L/L coordinates (B, L, H) of a point on the earth is expressed as:

$$\begin{cases} x = (R+H)\bullet\cos B\bullet\cos L \\ y = (R+H)\bullet\cos B\bullet\sin L \\ z = (R+H)\bullet\sin B \end{cases}$$

Where R is the radius of the earth and H is the height above sea level.

There are two forms for indicating the corresponding relationship of the geometric data of the terrain to L/L information in the geometric model of the terrain. One is that the description part of the DEM data file is inserted the L/L information. Another is of using the L/L information to directly name the DEM dataset file. For instance, the American's DEM data STRM (Shuttle Radar Topography) is stored in a file by the name of X1X2X3X4.hgt.zip. This DEM data provides a file for every L/L grid. In the file, X1 is N or S denoting North or South, X2 is the grid bottom latitude of the grid. X3 is E or W denoting East or West and X4 is the left side longitude of the grid. This paper uses the similar way to name the DEM data within the range of the east longitude and north latitude. The name of DEM data file used in authors' experiments is X1X2.dtm. The bottom latitude is X1 of north latitude and the left side longitude is X2 of east longitude.

5. A new dynamic-loading method

5.1 Deciding the present dataset

It is necessary to obtain the position relationship between sub-dataset and dataset when we want to load required sub-dataset into main memory. Here the dataset is the regular grid model, the presumed grid size is WIDTH×HEIGHT and the fixed size of the sub-dataset is width×height. The loaded sub-dataset is called as Sub. The left-bottom vertex point of the Sub is the start point P and the right-top point is the end point Q. As judged by the point P and Q, the DEM dataset including point P is the present DEM dataset called Curr.

For establishing coordinate system of the Curr, the definition of the right angle coordinate system is that the left-bottom vertex point of the Curr is the origin of the coordinate system, x axis is in the horizontal direction and y axis is in the vertical direction. During the roaming process, the Curr is alterable, so the coordinate system is alterable relative to the Curr. However, the coordinate system of the Curr is steady relative to the Sub because the coordinate definition of the Curr depends on the sub.

5.2 Judgment of the present dataset's position

It is known that the adopted DEM is a regular grid model, so there are only eight vicinal dataset surrounding each dataset, as indicated in the Figure 1. The gray Square ① in Figure 1 is the present dataset Curr. According to the coordinate of the point P(Px, Py), there are nine possible instance position of the movement of the present dataset. When the dataset moves to different places, the data file of the corresponding place will be opened. In this case, the judgment of the dataset movement to the following place can be made and the data entry of the following sub-dataset will carry out easily.

Using pseudo-code, the judgment of the present dataset's position can be described as follows:

```
If 0 ≤ Px < WIDTH and 0 ≤ Py < HEIGHT, then
Curr=①, right coordinate system is not change;
If Px < 0 and 0 ≤ Py < HEIGHT, then
Curr=⑥, reset up coordinate system;
If Px ≥ WIDTH and 0 ≤ Py < HEDHT, then
Curr=②, reset up coordinate system
```

Similarly, when Px and Py have different value, the DEM dataset Curr will locate in the corresponding positions, such as ③,④,⑤,⑦,⑨.

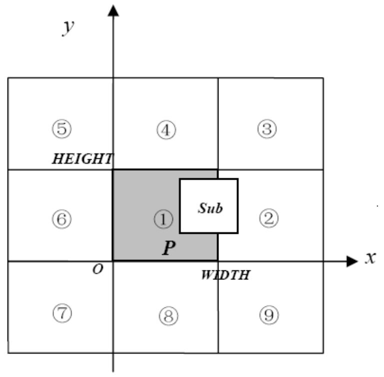

Figure 1. The dataset Curr's position

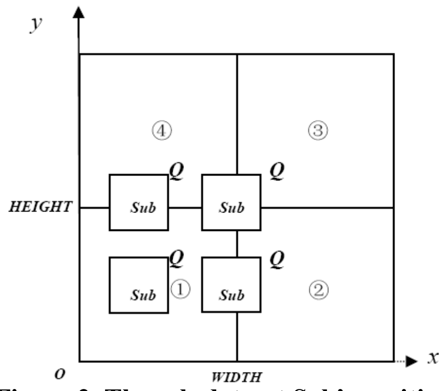

Figure 2. The sub-dataset Sub's position

5.3 Judgment of the sub-dataset's position

It is known that the coordinate system of the dataset Curr depends on the point P and the sub-dataset is smaller than the dataset, so there are four possible positions of the sub, as shown in Figure 2. After judging the sub-dataset position and opening the corresponding dataset file, the required sub-dataset will be read in.

When Q is on the dataset Curr, the present dataset file will be opened to get the data of the sub-dataset. When Q is on the right, it needs to open the present dataset file and the dataset on the right side to get all the data of the sub-dataset. If Q is on different positions, the vicinal dataset files will be opened to get all the needed data of the sub-dataset.

We can describe the above procedure for judging the sub-dataset's position using pseudo code as follows:

```
If Qx ≤ WIDTH and Qy ≤ HEIGHT(Q is in Square ①), then
    Open the ①dataset file, and read in the data of the sub-dataset in the corresponding position;
If Qx > WIDTH and Qy ≤ HEIGHT(Q is in Square ②), then
    Open the ①and ②dataset files, and read in the data of the sub-dataset in the corresponding position;
If Qx ≤ WIDTH and Qy > HEIGHT(Q is in Square ④), then
    Open the ① and ④dataset files, and read in the data of the sub-dataset in the corresponding position;
If Qx and Qy are where else(Q is in Square ③), then
    Open the ①, ②, ③and ④dataset files, and read in the data of the sub-dataset in the corresponding position.
```

5.4 Process of loading sub-dataset

Through the analysis above, the process of loading sub-dataset can briefly illustrate as follows:

Step1: obtain the present L/L information from GPS;

Step2: decide the present dataset according the present L/L position;

Step3: open the present dataset file, read in the data's width and height, and read in the origin coordinates and L/L span of the dataset;

Step4: calculate the start point P's coordinates, and calculate the end point Q's coordinates of the sub-dataset;

Step5: decide the present dataset's position according the point P, open the new present dataset file, and read in the dataset's width and height;

Step6: decide the sub-data's position according to the point Q, open all the dataset files relative to the sub-dataset, and load the data of sub-dataset into main memory.

The calculation of the point P and Q is as follows:

$$\begin{cases} Px = w \bullet (CB - OB) / DB \\ Py = h \bullet (CL - OL) / DL \end{cases}$$

$$\begin{cases} Qx = Px + width \\ Qy = Py + height \end{cases}$$

In the above equations, w and h are the width and height of the present dataset respectively, the origin coordinates of the dataset is (OB,OL), the longitude span is DB, latitude span is DL, and the present latitude and Longitude is (CB, CL).

When having loaded the data of sub-dataset into main memory, we render the data by means of fixed quadtree or bintree model directly. In this way, we can realize the large terrain dataset visualization rapidly and smoothly.

6 Results

In our experiments, the adopted DEM data is the regular grid model, the size of the grid is 2048×2048, and the total number of the DEM data is 100. The size of sub-dataset is 512×512. The whole loading method only needs 512×512 bytes main memory and costs less 1 millisecond to load data of a sub-dataset(CPU: 1.7G, RAM: 256M). In our experiments, the whole roaming process seems very smooth without stagnancy at the frame rate of about 37fps. Figure 3 shows a frame of the experiments, and the edge shows the boundary of the two different DEM dataset.

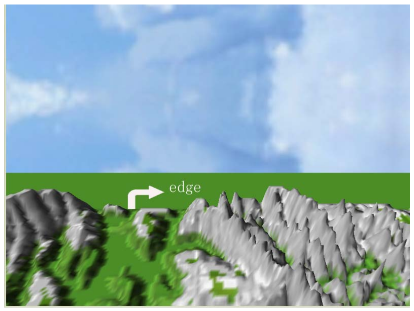

Figure 3. A frame of the experiments

The method of this paper is very simply and fast, it dose not need to organize data out-of-core. In this way, file fragmentation can be avoided and DEM files can be added or reduced at any moment. In addition, this algorithm realizes data loading by means of the index of longitude and latitude information, so it needn't forecast the movement direction of view-point and load non-current frame data beforehand. By comparison of literature [8], the occupancy of CPU is reduced to 33% and the occupancy of memory is reduced to 62%. In this way, our algorithm adapts embedded environments well.

In our method, we just load terrain data. In future, we will consider loading texture data in the same way. The size of sub-dataset is fixed now, which makes our method do not very well in some situation. We will do more research on this problem.

References

[1] B. Rabinovich and C. Gotsman, “Visualization of large terrains in resource-limited computing environments”, Visualization '97, Proceedings, Oct. 1997, pp. 95_102.

[2] R. Pajarola, “Large scale terrain visualization using the restricted quadtree triangulation”, Visualization '98, Proceedings, Oct. 1998, pp. 19_26, 515.

[3] J. Haber, F. Zeilfelder, O. Davydov and H.-P. Seidel, “Smooth approximation and rendering of large scattered data sets”, Visualization, 2001, VIS '01, Proceedings, Oct. 2001, pp. 341_571.

[4] P. Lindstrom and V. Pascucci, “Visualization of large terrains made easy”, Visualization, 2001, VIS '01, Proceedings, Oct. 2001, pp. 363_574.

[5] Din-Chang Tseng, Chung-Chieh Huang and Cheng-Ta Chen, “Dynamic-loading view-dependent multiresolution terrain visualization”, Geoscience and Remote Sensing Symposium, 2001, IGARSS '01, IEEE 2001 International, July 2001, Vol.3, pp. 982_984.

[6] P. Lindstrom and V. Pascucci, “Terrain simplification simplified: a general framework for view-dependent out-of-core visualization”, Visualization and Computer Graphics, IEEE Transactions on, July-Sept. 2002, pp. 239_254.

[7] U. Gudukbay and T. Yilmaz, “Stereoscopic view-dependent visualization of terrain height fields”, Visualization and Computer Graphics, IEEE Transactions on, Oct.-Dec. 2002, pp. 330_345.

[8] Xiaohong Bao and Renato Pajarola, “LOD-based clustering techniques for efficient large-scale terrain storage and visualization”, Visualization and Data Analysis, 2003, pp. 5009-023.

[9] Yanqing Lu, “Study of the Real-Time Rendering for Large-Scale Terrain Dataset”, A Dissertation Presented to the Graduate School of Zhejiang University in Partial Fulfillment of the Requirement for the Degree of Doctor of Philosophy, 2003.

[10] L.M. Hwa, M.A. Duchaineau and K.I. Joy, “Real-time optimal adaptation for planetary geometry and texture: 4-8 tile hierarchies”, Visualization and Computer Graphics, IEEE Transactions on, July-Aug. 2005, pp. 355_368.

[11] Jianjun Li, Junshan Li, Shuangyan Hu, Xia Ye and Zhisheng Li, “Research of Terrain Simplification Based on Regular Grid”, Intelligent Control and Automation, 2006. WCICA 2006, The Sixth World Congress on, June 2006, pp. 9893-9896.

[12] Kai Xu, Xiaofang Zhou, Xuemin Lin, Heng Tao Shen and Ke Deng, “A Multiresolution Terrain Model for Efficient Visualization Query Processing”, Knowledge and Data Engineering, IEEE Transactions on, Oct. 2006, pp. 1382-1396.