A New Quadtree-based Terrain LOD Algorithm

Jian Wu[1,2]
1.Provincial Key Laboratory for Computer Information Processing Technology; 2.The Institute of Intelligent Information Processing and Application, Soochow University, Suzhou 215006, China
Email: szjianwu@163.com

Yuan-feng Yang[1,2], Sheng-rong Gong[1,2], Zhi-ming Cui[1,2]
1.Provincial Key Laboratory for Computer Information Processing Technology; 2.The Institute of Intelligent Information Processing and Application, Soochow University, Suzhou 215006, China

***Abstract*—Terrain LOD algorithm is a dynamic and local dough sheet subduction algorithm. On the basis of the research on traditional quadtree algorithm, this paper proposed a new terrain LOD algorithm using quadtree. On deviation computing standard, besides static deviation and view distance, motion vector and observation vector are introduced, which made the result of deviation computing close to the accurate deviation value. On crack elimination, a new crack elimination algorithm is proposed, which increases the list of crack polygon specially and eliminates all the cracks by one time rendering. So, the judge of crack vertex is not required in the course of terrain block segmentation and rendering. The experimental result shows that our algorithm is practical and effective, which can meet the need of real-time rendering for massive terrain data.**

***Index Terms*—quadtree; terrain LOD; deviation factor; crack elimination**

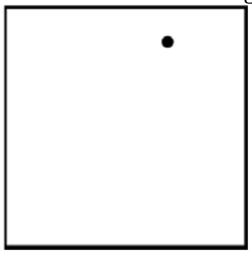
I. INTRODUCTION

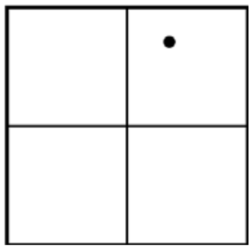
With the development of 3D GIS and VR, 3D visualization technology of massive terrain gets more and more concern. Because of the limitation of the own characteristics of terrain data and computer processing ability and storage capacity, traditional technology cann't meet the need of the real-time rendering of massive terrain data. So, many data simplification algorithms are proposed and have made some progress, among which LOD algorithm and its some varieties is the most representative[1]. Because of simple data structure and high-efficient quality, Quadtree LOD algorithm using RSG gets widely used. Terrain data is stored in the memory in the traditional quadtree LOD algorithm and cann't meet the need of massive data storage. Some scholars put forward the Out-of-Core simplification method, which store the terrain data in the external storage firstly and read into the memory using some scheduling algorithm. But this method have the shortcoming of occupying much CPU resource. In the realistic outdoor scene simulation, the need for CPU resource in the physical simulation and particle system is very large, and much CPU resource occupied by LOD algorithm cann't be endured.

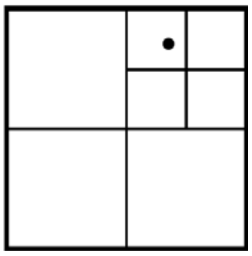
For these reasons, this paper proposed a new quadtree-based terrain LOD algorithm. First of all, change the traditional status of the sub-block matrix into the form of hash table to store the terrain information block, which not only increases the amount of information storage, but also reduces the memory consumption under the premise of hardly no speed loss. On the deviation calculating, in addition to the static deviation and view distance, two deviation factors: motion vector and observation vector are introduced, so that the result of deviation calculation is more close to accurate calculation deviation and the exact degree of deviation evaluation is improved. On the crack elimination, this paper presents a new algorithm to eliminate the cracks, which increases the specialized list of crack polygon, through a rendering to eliminate all the cracks, therefore it has no need to determine the crack vertex in the process of the terrain blocks segmentation and rendering.

II. TRADITIONAL QUADTREE LOD ALGORITHM

Quadtree-based LOD algorithm was first proposed by LindStrom and it suggested that it works in the terrain data of RSG. The algorithm is based on the idea that the terrain is divided into inconsistent small pieces under the premise that the user can accept the loss of the quality of images, to fit the similar original DEM data and to keep the number of triangles need for rendering in a reasonable magnitude[2,3]. Quadtree LOD algorithm requested terrain on behalf of the DEM elevation data are the square, and the length is the N-th power of 2. In the node division process, the quadtree structure is used to store the results of segmentation, and each quadtree node represents a terrain block. Top-level node represents the entire data block, the leafy nodes represent the terrain blocks that can be directly rendering rather than partition, and the middle non-leaf nodes represent the terrain blocks that need to continue segmenting because it does not meet the requirements of the precision. The principle of the Quadtree partition terrain is shown in Figure 1.


(a) The 1th Segmentation


(b) The 2th Segmentation


(c) The 3th Segmentation

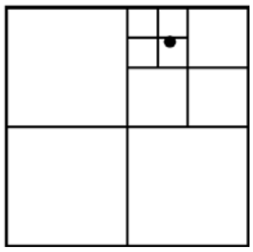

(d) The 4th Segmentation

Figure 1. Quadtree segmentation procedure of terrain block

As shown in Figure 1, quadtree partition is a recursive process, the parent node contains child nodes completely. The segmentation process is a progressive refinement process. In the final step, the terrain block where the viewpoint lies is segmented most precisely, and it shows the best quality. However, the parts far away from the viewpoint have the rough segmentation result. In this way, the quantity of rendering triangles has reduced significantly without the loss of image quality. The use of quadtree structure can be easily describe the split process mentioned above, as shown in the hierarchical structure of Figure 2, in which gray terrain block can be rendered directly.

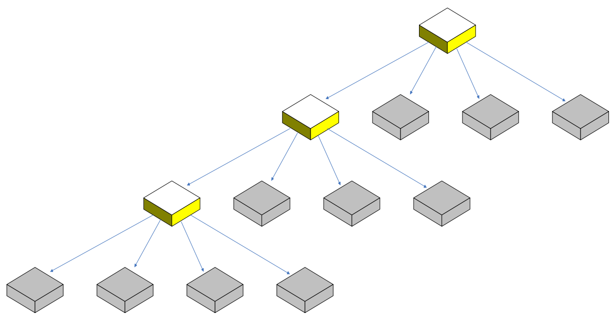


Figure 2. Quadtree hierarchical representation of segmentation procedure

In the process of quadtree partition, a standard must be used to determine whether the current quadtree node should continue to split. Through the deviation standard, we can calculate the node deviation for each node. If node deviation meets a certain threshold, we do not have to split, otherwise divide it into four sub-nodes. As a method of space division, quadtree segmentation method is not unique for terrain LOD algorithm. But the basic principle is similar, and different deviation standards can be used in different areas, which has the application in many field of computer graphics.

III. Deviation Calculation

The standard of the deviation evaluation is very important in the process of quadtree generated. Under the premise of achieving the desired image, a good standard deviation evaluation has the least number of quadtree nodes partition. The deviation value of terrain block reflects the real gap between the real terrain elevation data with the data when the current terrain block does not continue segmenting. In general, there are two types of calculation deviation, one way is accuracy calculation which can reflect the real gap between the current terrain block and the real data. Another method is approximate calculation that is only similar to the gap between the current terrain block and the real data to a certain extent. The process of the accurate calculation method is so complexity that can not calculate the real-time deviation of terrain block when the viewpoint moves. Although similar deviation evaluation standard can only calculate the approximate deviation, its compute speed is quick. Thus the approximate deviation calculation is used as deviation evaluation criteria to the quadtree segmentation quadtree in general[4]. Because few factors are taken into account relatively, the classical approximation deviation calculation has only considered with the static deviation that related to the roughness of surface and the dynamic deviation that related to distance between the viewpoint with the center of the current terrain block, so it can not meet the requirement of real-time roaming for massive terrain data.

A. Static Deviation

The calculation of static deviation is just related with the roughness of terrain surface. Generally speaking, the rougher the terrain is, the preciser the segmentation of terrain block should be. As shown in Figure 3, there are nine important points in one terrain block, which includes central point 0, edge point 2,4,6,8 and corner point 1,3,5,7. Through the elevation value of corner point 1,3,5,7, the appropriate elevation value of point 0,2,4,6,8 can be calculated via interpolation.

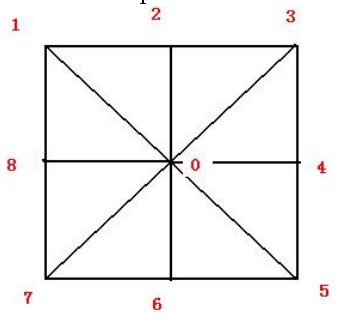


Figure 3. Schematic diagram of terrain block

From Figure 3, the value of static deviation is measured by the deviation between the interpolated elevation value of central point and edge point with the actual elevation value. Then the maximum value of them is regard as the final static deviation.

B. Viewpoint Distance

The second evaluation factor is the distance from the viewpoint of current camera to the central point of terrain block. According to the projection principal of camera in computer graphics, to the object with same size, the farer the distance to viewpoint is, the smaller the projection area is. Inversely, the closer the distance to viewpoint is, the bigger the projection area is. So to terrain block far from viewpoint, less triangular dough sheets are needed for rendering, that is, the terrain blocks may be a bit bigger. And to terrain block close to viewpoint, more triangular dough sheets are needed.

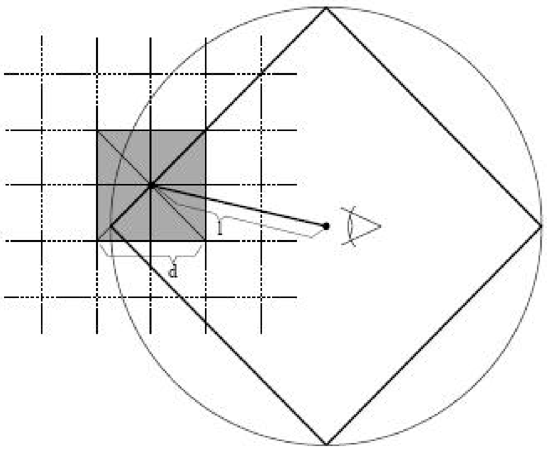


Figure 4. Schematic diagram of viewpoint distance

As shown in Figure 4, the value of viewpoint distance can be gotten from calculating the distance among the two points directly. One point is the position of viewpoint gotten from the information of camera. The other one is the central position of current terrain block gotten from the member variable of the quadtree structure.

C. Motion Vector

The traditional deviation calculation only connected with the rough degree of objects and the distance from the viewpoint to the center of the current terrain block. In fact, when the viewpoint is fast-moving, the rendering image will produce a optical phenomenon named motion-blurred, by this time rendering images would has visual fuzzy feeling even if the precision is very high, as shown in Figure 5. At this time, the level of detail of the rendering terrain block can be reduced appropriately, and we regard the mobile speed of cameras as one of the deviation evaluation factors.

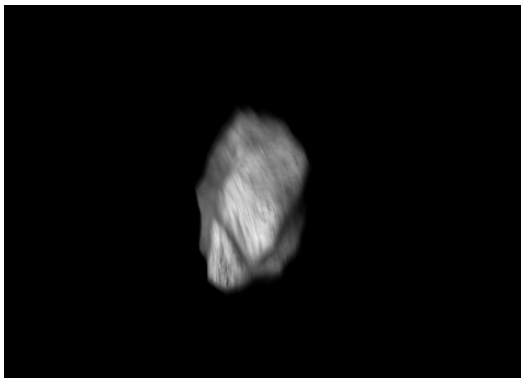


Figure 5. Vague effect of moving object

The deviation calculation of terrain block with considering the moving speed of camera is suitable for the VR system which the camera has regular movement. The faster the movement of the camera, the lower levels of detail of LOD. The cameras' moving speed is called motion vector, the method of solving motion vector is to record the current location of the camera at a fixed frame count, and then derive the current moving speed and the direction in contrast with the previous location. The quantity of midfeather frame takes 5 generally, and it can also be fine adjustment according to specific circumstances.

D. Observation Vector

Using the observation vector as the deviation evaluation factor of LOD originated from the precise deviation calculation. Accurate deviation calculation calculates deviation in the image space, because its computation is too complex, so it can not be applied in practice, but its thought could be used for reference. Under the premise that other conditions are same (the rough degree of upper, the distance to camera), if the line of sight is perpendicular to the terrain film, the terrain film occupy a larger area of the screen. The greater angle between the line of sight and the normal vector of the terrain film, the smaller size of the terrain film on the screen. When angle greater than 90 degrees, this shows that the terrain film with his back to the observer can be omitted without rendering. And the principle of deviation calculation related to the observation vector is shown in Figure 6, the precision segmentation of terrain block a higher than b's.

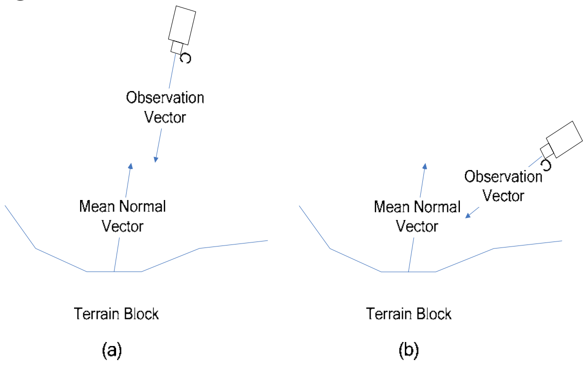


Figure 6. Schematic diagram of deviation calculation related with observation vector

The method to calculate the related deviation calculation factors of the observation vector is: derive the mean normal vector of the terrain block firstly, and then calculate the dot product between the observation vector with the mean normal vector of terrain block. Because in the normal conditions the direction of the observation vector is contrary to the terrain block's mean normal vector, the dot product is taken opposite direction of the observation vector. The value of the dot product is the two vector's mode and then multiplied by the cosine values between their angle, and cosine functions is a decreasing function between[0^o, 90^o], that is, the greater the value of dot product, the smaller the angle. Therefore, using the value of the dot product can determine the size of angle between two vectors. Suppose the observation vector is v_e, the mean normal vector is v_a, then the formula for calculating the deviation v_p related to the observation vector as follows: $v_p = \max(-v_e, v_a, 0)$. The accuracy mean normal vector can not be obtained before the terrain film deciding whether to segmentation. Even if the terrain block has been divided, get the precise mean normal vector also spend a long time. Therefore, we use the mean normal vector as the approximation in the deviation determination, that is, use the mean normal vector of triangle sectors connected by central point with the four corners to replace the actual mean normal vector.

E. Deviation Calculation Formula

Synthesizes factors discussed above, on the base of other scholars' work, in this paper the deviation calculation formula is determined in the following form.

$$f = \frac{l.v}{d.c_1.\max(c_2.dh,1).c_3.c_4.v_p}$$

Where d is the side length of the current terrain block, dh is the rough degree of the terrain, l is the distance from viewpoint to the center of terrain film, v for camera's traveling speed, v_p is the dot product that the observation vector to the mean normal vector. c_1、c_2、c_3、c_4 are regulatory factors which are used to adjust the impact of various factors to the deviation. c_1 adjusts the distance from the viewpoint to the center of terrain film, c_2 adjusts terrain rough degree, c_3 adjusts the traveling speed of camera. c_4 adjusts the angle that the observation vector with mean normal vector. In the process of terrain partition, through the value of f to determine whether to continue devise the current terrain. If f is less than 1 divide the node, otherwise directly rendering the nodes. Apart from the value of f, the value of d is to determine whether to continue divide the node. If d=2 that the current node has been assigned to the smallest precision degree can not be divided again, by now, even if f>1 must stop the division.

IV. Crack Elimination Algorithm

Quadtree LOD algorithm will reduce the total number of triangles through dividing terrain into different size square terrain films, each terrain film has a different levels of detail, but in the middle of different levels of details will have cracks. As a result, during the rendering process through the Quadtree LOD algorithm, we must consider the elimination of cracks[5].

A. The Cause of Crack

Quadtree LOD algorithm is a part simplification method, each terrain piece has different resolution after it being dealt with. If we render directly without doing this, cracks will appear. In Figure 7, points A, B are ignored in drawing the right terrain, which will lead to cracks in the two terrain piece.

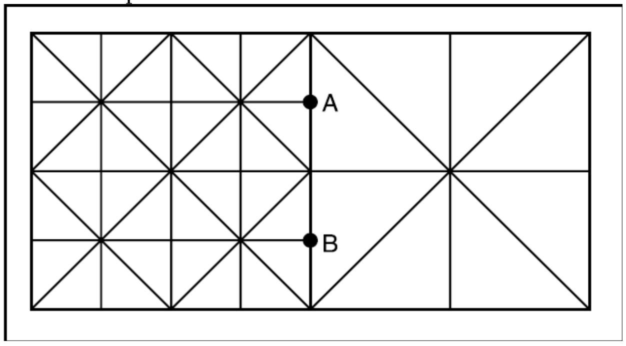


Figure 7. The cause of crack (crack generated at A and B)

Figure 8 is the result of rendering directly, the cause of the cracks is: the precision of the left terrain piece is higher than the right part.

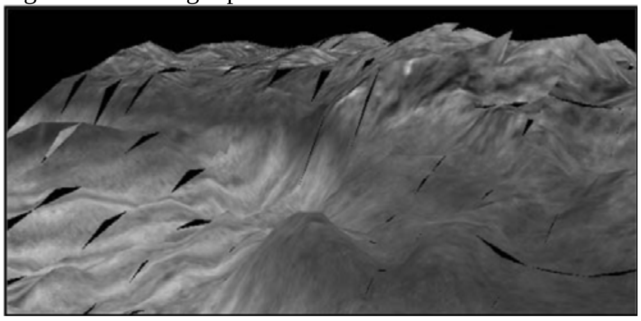


Figure 8. Crack generation at the time of direct rendering without crack elimination

B. Traditional Algorithm of Crack Elimination

There are two methods to eliminate the above-mentioned A, B cracks: one is remove A, B in the left high-precision terrain piece. That is, when rendering the left terrain film, if the detail level of the right part adjacent to point A is less than the left, then we will abandon point A when generating a rendering list, so that the emergence of cracks can be avoided. Another method is to increase detail level of the right terrain block[6]. That is, when rendering the right terrain block, take point A as a vertex of the terrain piece too, re-partition the right terrain piece. The method of increase vertex of the low level detail terrain piece will need to re-add triangle as a result of increase vertex, its efficiency is not as good as the method skip the vertex of the high-level details of the terrain blocks. So in practical applications we use the first method to avoid cracks, the principle of the first method eliminate cracks is shown in figure 9.

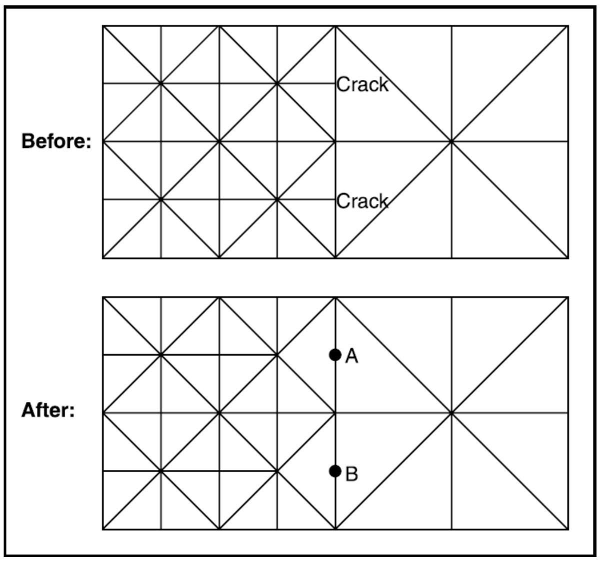


Figure 9. Algorithm principal of crack elimination

The eliminating cracks algorithm of the first method generally be nested in the rendering code, then determine whether a vertex should be added to the rendering list to eliminating cracks in the rendering process.

The pseudo code is as follows.

```
//If return true, it represents that current Vertex doesn't generate the crack, and can be added.
bool AddVertex(Tile, Vertex)
{
  //If current point is edge point, it is necessary to judge. Otherwise, add current point directly.
  if(Vertex is EdgeVertex)
  {
    //Get the neighboring terrain block
    neighborTile=GetNeighbor(Vertex);
    //If the neighboring terrain block doesn't exist,
     add current point direcly.
    if(neighborTile not exist)
      return true;
    //If the Lod of current terrain block is larger than
    the neighboring terrain block, omit it.
    if(Tile.Lod > neighborTile.Lod)
      return false;
  }
  return true;
}
```

In this eliminating cracks algorithm, the detail level of adjacent terrain piece can not be larger than one, or the algorithm will be ineffective. There are two ways to enable the detail level of adjacent terrain does not larger than one: one is to re-traverse all the terrain piece after finished the node partition algorithm, re-partition those adjacent terrain piece whose level larger than one, ensure that their levels is less than or equal to 1. Another way to start from the deviation-judge equation, through mathematical proven, as long as the static deviation of two adjacent terrains meets a fixed conditions, we can ensure that the detail levels of adjacent terrain blocks not larger than one.

C. *Batch Crack Elimination Algorithm*

The traditional crack elimination algorithms all start from change the geometrical structure of the terrain itself. Whether increase or delete vertex, it needs to ensure the detail levels of adjacent terrain not larger than 1 in the terrain partition process or after the process then in some ways. In the rendering process, each edge point needs to judge whether to be added, so to eliminating cracks. This algorithm needs to traverse the terrain twice, as well as increase a great amount of codes in the rendering function.

This paper puts forward a more efficient crack dealing algorithm. This algorithm only need to re-traverse the quadtree terrain film after the quadtree nodes partition is finished, record all the vertex producing cracks and the number of the cracks, generate a cracks list. In this process, no additional code is needed to change the detail levels of the terrain film, we can make levels of adjacent terrains larger than one. In the rendering process, it does not need to determine whether to add edge point, just need to one-time generate some triangles according to the cracks list to cover the cracks after the end of the normal rendering, in this way, only one-time rendering can make all the cracks eliminated. The principle of this cracks elimination is shown in Figure 10.

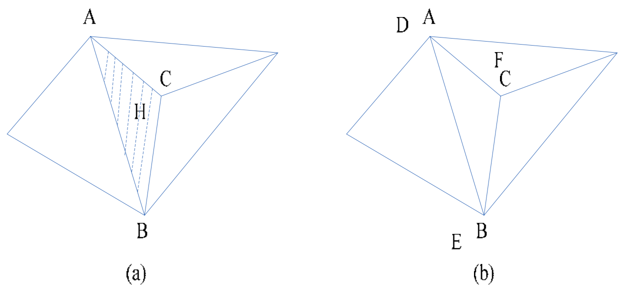


Figure 10. Schematic diagram of batch crack amendment

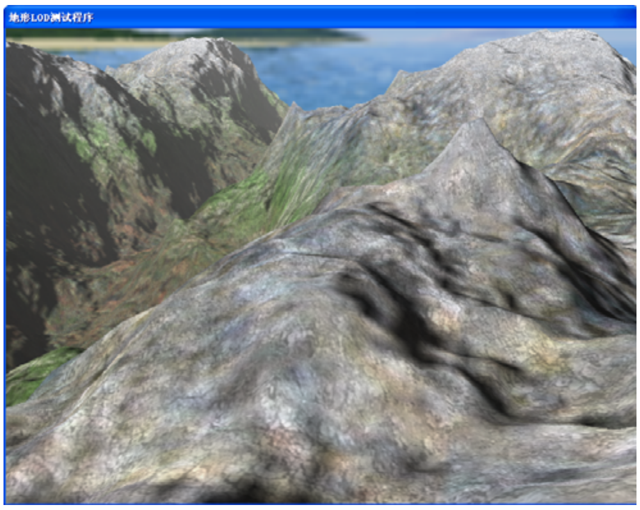
The pseudo code that generates the CrackList is shown as follows.

```
void GenerateCrackList()
{
//dealing with each terrain film portioned currently
for each Tile in TileSet
  // if there are cracks, judge the LOD difference with
  its adjacent terrain, if larger than 1, then cracks must
  be existed.
  if(HasCrack(Tile))
  {
    // obtain the vertex list make up of the cracks.
    Crack = GetCrack(Tile);
    //if this crack not existed in the crack list.
    if(Crack not exist in CrackList)
    {
      // add this crack to the crack list.
      CrackList.AddCrack(Crack);
    }
  }
loop
}
```

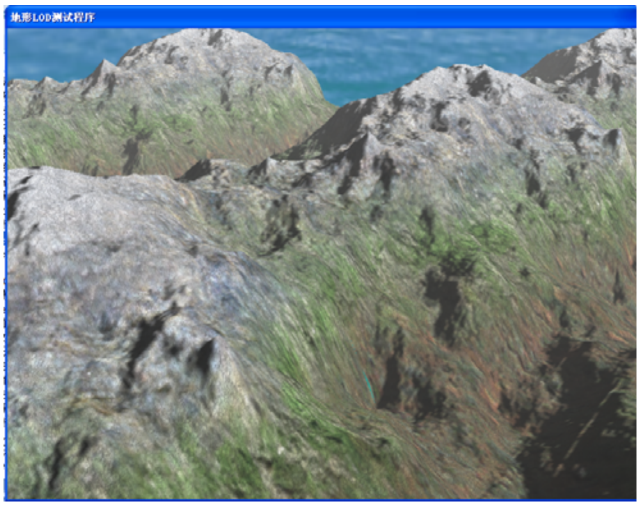
The Crack structure contains a series of vertex number which make up of the cracks. According to the information of Crack in the CrackList based on above algorithm, generate a series of triangles, which can cover all the cracks, and it only need one time rendering in order to eliminate all the cracks.

V. Experimental Result and Analysis

A. *Experimental Effect*

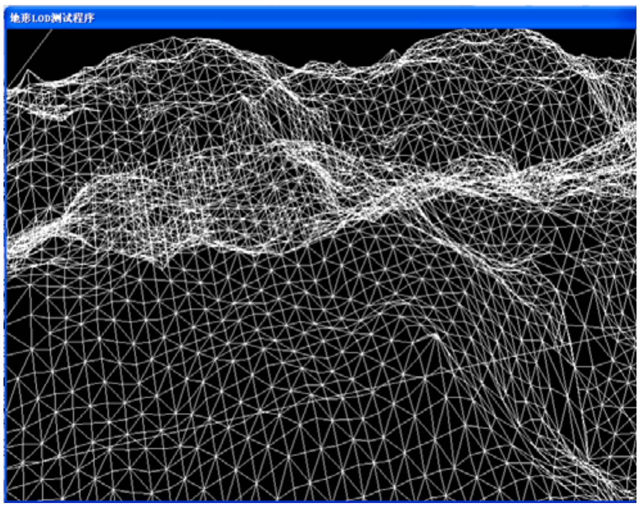


(a)

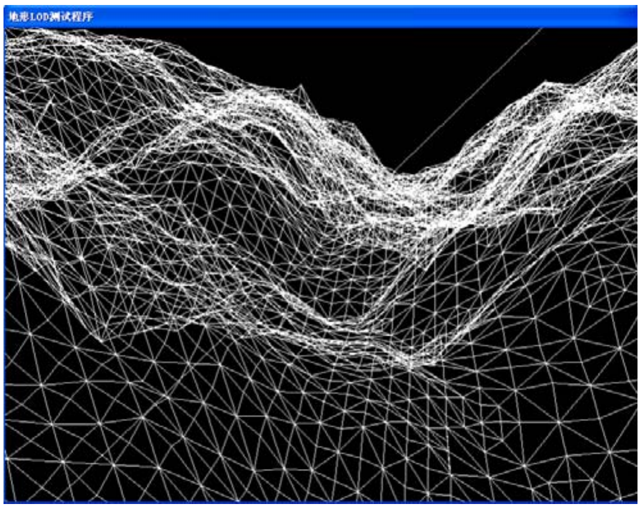


(b)

Figure 11. The screen-capture image under normal browsing state



(a)



(b)

Figure 12. The screen-capture image under wire-frame model

B. *Algorithm Steps*

The proposed algorithm flowchart is as follows.

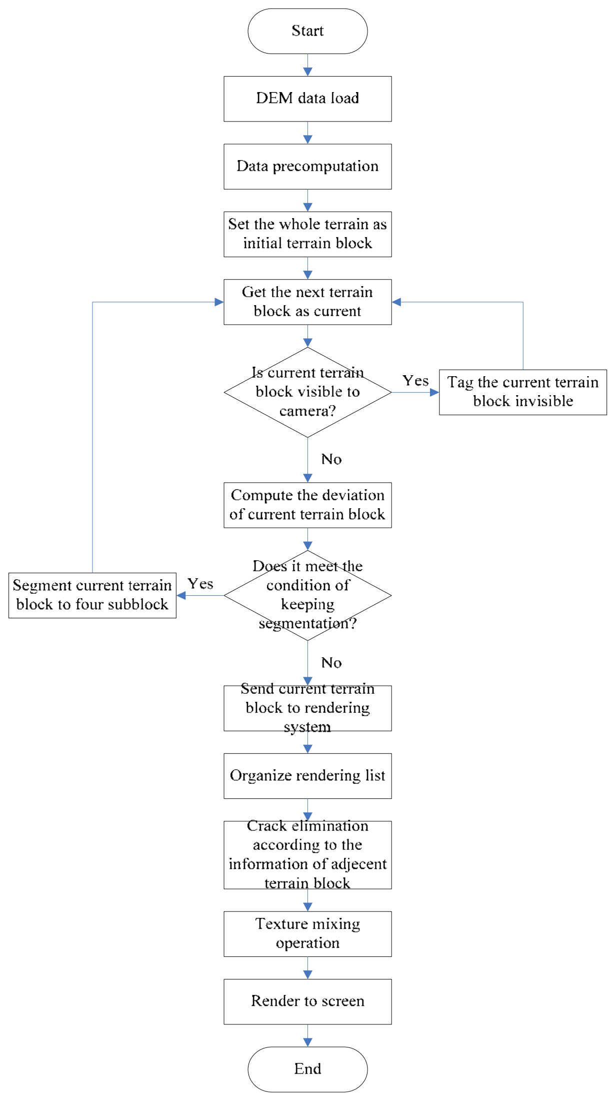

Start
DEM data load
Data precomputation
Set the whole terrain as initial terrain block
Get the next terrain block as current
Is current terrain block visible to camera?
Yes
Tag the current terrain block invisible
No
Compute the deviation of current terrain block
Does it meet the condition of keeping segmentation?
Yes
Segment current terrain block to four subblock
No
Send current terrain block to rendering system
Organize rendering list
Crack elimination according to the information of adjecent terrain block
Texture mixing operation
Render to screen
End

Figure 13. The algorithm flowchart

C. Result Analysis

LOD algorithm using quadtree has been realized completely in the experimental program. Hardware and software environment are shown in Table 1 and Table 2.

Table 1 Hardware environment

CPU	Intel Core(TM)2 Duo CPU T7250 @ 2.0GHz
Memory	1.00GB
Display Card	NVIDIA Quadro NVS 135M

Table 2 Software environment

Programming Language	C++
Compiling Environment	Visual studio6.0
Graphical API	DirectX9.0
Operating System	Windows XP
Shader Support	No

The used terrain data is elevation data with 256×256 size generated randomly from fractal algorithm. By graphic stitching, infinite terrain scene is simulated. Through camera cutting and the simplification of LOD algorithm using quadtree, the quantity of triangular dough sheet, which need to render at the time of normal browsing, is approximately 10,000. Meanwhile, the frame rate can reach the speed of 130~170 frames per second.

The following table is about the comparison of FPS under different initial terrain parameters setting.

Table 3 The comparison of FPS under different initial terrain parameters setting

FPS	Crack Elimination Algorithm		Deviation Calculation		Overall Optimization	
Roughness	Trad. Algo.	After opti.	Trad. Form.	After Opti.	Before Opti.	After Opti.
0.1	107	127	115	106	82	151
0.3	104	121	114	113	81	148
0.6	99	112	109	121	78	136

The following table is about the quantity of triangular dough sheet under different initial terrain parameters setting.

Table 4 The comparison of the quantity of triangular dough sheet under different initial terrain parameters setting

Quatity of dough sheet	Crack Elimination Algorithm		Deviation Calculation		Overall Optimization	
Roughness	Trad. Algo.	After Opti.	Trad. Form.	After Opti.	Before Opti.	After Opti.
0.1	9282	9431	9634	9621	9502	9703
0.3	10042	10258	10134	9952	10552	10524
0.6	10156	10553	11463	10456	11543	10663

From the analysis of table 3 and table 4, under the same terrain condition, the quantity of triangular dough sheet that batch crack elimination algorithm needs to render is more than the traditional crack elimination algorithm because of the importation of crack nodes. And because of the reduction of CPU computation, FPS will rise inversely. If the terrain is relatively flat, it cannot reduce the quantity of triangular dough sheet by introducing new deviation factors, and will slow down the rendering speed. But when the terrain becomes rough, it can reduce the quantity of triangular dough sheet obviously. Without any optimization, FPS can only reach around 80 because each frame all needs to re-generate the quadtree nodes and each node all need to send the rendering action to display card. Comparatively, after the optimization of memory and rendering, FPS has the apparent improved.

VI. Conclusion

With the deep research on traditional quadtree algorithm, this paper made the improvement on the quadtree LOD algorithm. This paper extends the traditional assessment standard of deviation, which not only consider static deviation and view distance, and introduce additional two assessment factor of deviation: motion vector and observation vector. It made the deviation computation result more close to accurate value. In the aspect of crack processing, batch crack filling algorithm based on discrete vertex is proposed according to the characteristics of block terrain, which improved the speed of crack elimination greatly. The experiment shows that the algorithm proposed in this paper save the CPU resource remarkably, and can meet the real-time rendering need of massive terrain data.

ACKNOWLEDGMENT

This research was partially supported by the Natural Science Foundation of China under grant No. 60970015, the 2008 Jiangsu Key Project of science support and self-innovation under grant No. BE2008044, the Project of Jiangsu Key Laboratory of Computer Information Processing Technology and the Applied Basic Research Project (Industry) of Suzhou City under grant No. SYJG0927.

REFERENCES

[1] Zhao Youbing, Shi Jiaoying, Zhou Ji, et al. A Fast Algor ithm for Large Scale Terra inWalkthrough. Journal of Computer-aided Design & Computer Graphics, 2002, 14(7):624-628.
[2] Rottger Stefan, Heidrich Wolfgang, Slusallek Philipp, et al. Real-time generation of continuous levels of detail for height fields[C]. In Proceedings of the 6th International Conference on Central Europe Computer Graphics and Visualization, Plzen, Czech Republic, 1998:315-322.
[3] Eric Lengyel. Mathematical method in 3D game and computer graphics[M]. Beijing: tsinghua university press, 2004.
[4] PAJAROLA R. Overview of quadtree-based terrain triangulation and visualization[R]. Technical Report UCI-ICS-02-01, Information & Computer Science, University of California Irvine, 2002.
[5] PAJAROLA R, ANTONIJUAN M, LARIO R. QuadTIN:Quadtree based triangulated irregular networks[A]. Proceedings IEEE Visualization, 2002, pp.395-402.
[6] Yin Yuan, Chen Guojun, Wu Wei. An Algorithm of Avoiding Crack for Rendering Parting Terrain. Journal of Computer-aided Design & Computer Graphics, 2006, 18(10):1557-1562.

Jian Wu was born in Nantong on the 29th April, 1979, and got master degree in the field of computer application technology from Soochow university, Suzhou city, China in 2004. The main research direction is computer vision, image processing and pattern recognition.

He works as a teacher in the same college after his master graduation. Now he is pursuing the doctoral degree. In the year 2008, the 8th IEEE International Conference on Computer and Information Technology was hosted by University of Technology, Sydney, Australia, and he was invited to serve as the session chair of “Image Processing, Computer Vision and Video surveillance”.

Mr. Wu, a member of China Computer Federation. He was awarded the Third Prize of 2007 Suzhou City Science and Technology Progress and the 2008-2009 Soochow University Graduate Scholarship Model.

Yuan-feng Yang was born in Yancheng on the 3th November, 1973, and got master degree in the field of computer application technology from Soochow university, Suzhou city, China in 2006. The main research direction is image processing and pattern recognition.

Sheng-rong Gong was born in Tianmen on the 12th April, 1966, and got doctoral degree in the field of computer application technology from Beihang University, Beijing city, China. The main research direction is image and video processing.

Zhi-ming Cui was born in Shanghai on the 4th July, 1961. Professor, PhD Candidate Supervisor. The main research direction is deep web and video mining.