Journal of Computer Science & Systems Biology - Open Access

Research Article
OPEN ACCESS Freely available online
doi:10.4172/jcsb.1000067

JCSB/Vol.3 Issue 5

Rendering for 3D Animation Based on Octree

Guofeng Qin* and Nan Jiang

CAD Research Center, Tongji University, Shanghai 200092, China

Abstract

In order to improve capability of some traditional algorithms for 3D virtual reality animation, this article comes up with a new improved algorithm, which can improve traditional ones and is more suitable for 3-D graphic environment. There are two steps. The first step is to break down the entire 3-D scene and then setup the Octree structure. The second step is to transform the Octree onto 2-D plane, and write down nodes which are modified and then render the modified ones. Many experiments demonstrate the superiority of the algorithm. Especially, in the process of blanking, the quadrants are shown from far to near and achieve 3-D display more effectively, according to the given object and the location of view.

Keywords: Octree; Blanking algorithm; 3-D; Neighborhood nodes

Introduction

Now days, computer animation technology is widely utilized in virtual reality for industry, joy games, GIS, education and many other fields. It gives "life" feeling to virtual human in computer animation. A key technology is rendering in 3D environment. 3D graphic rendering are relative to the motion controlling, motion path planning and 3D graphic displaying (Geng and Zhou, 1996; Ma et al., 2006). 3D graphic rendering algorithms are affecting time cost and memory resource, which are essential factors of computer animation.

A fast search Octree algorithm is used for finding the 4 pairs of feature points to estimate the viewing direction uses on effective two level database, which is based on matching the object contour to the reference viewing directions. The initially best matched viewing direction is calibrated by searching for the 4 pairs of feature points between the input image (Lee et al., 2010).

In industry design and 3D film, 3-D animation design is necessary for last practicing. An Octree-based (numerical control) NC simulation (Oct-OAC) system is developed for end milling. Oct-OAC has a geometric modeling module to simulate the geometry of material removal process. Every object in the machining environment such as cutter, instantaneous work-piece, swept volume, etc. is stored as Octree, an inexact representation of solid. Using this module, one can predict the geometry of the material removed at any instant of time and update the geometry of the blank subsequently. Optimization of cutting parameters using Oct-OAC is achieved through optimization module using a mechanistic model for computation and prediction of the cutting forces at any instant. The basic input for this module is the geometry of the contact surface between the cutter and work-piece which comes from the geometric modeling module using an Octree-based solid modeler. The mechanistic modeling module can predict the instantaneous cutting forces from the instantaneous contact geometry and other process parameters like material combination of cutter-work-piece, parameters defining cutter geometry, and current cutting parameters such as N and f. Using this prediction, it will modify the cutting parameters for maximizing the material removal rate. This way, the mechanistic modeling module does what an adaptive controller will do with the help of force sensing. Therefore, the NC program optimization done using the Oct-OAC system is actually off-line adaptive control (Karunakaran et al., 2010). For the computer animation processes, modeling, tracking and rendering are essential. Rendering plays a crucial role in the entire animation production process and blanking is the key to rendering (Zhou and Yang, 1993). A parallel geometric multi-grid algorithm for solving variable-coefficient elliptic partial differential equations on the unit box is studied with utilizing highly non-uniform, Octree-based, conforming finite element discretizations. This Octrees are 2:1 balanced, that is, we allow no more than one Octree-level difference between octants that share a face, edge, or vertex. This parallel algorithm input is an arbitrary 2:1 balanced fine-grid Octree and whose output is a set of coarser 2:1 balanced Octrees that are used in the multi-grid scheme. The overall scheme is second-order accurate for sufficiently smooth right-hand sides and material properties (Sampath and George, 2010).

There are many traditional algorithms which are easy to operate, such as Z-buffer algorithm but it often takes more time to calculate, they are not suitable for animation rendering. Depending on the different classification of several traditional blanked algorithms are introduced in brief. The rendering important function in computer animation can be seen Figure 1. Firstly, the 3D model will be constructed, then the models will be integrated for animation, lastly, the 3D rendering will be processed in an assembly animation scene.

By the method of object in space

(1) Graphic formula: According to analytic theory, we can determine a given point in the front or back of the plane, through the standard equation of the plane. The Graphic formula uses this theory to determine observable points on the surface. If the point locates

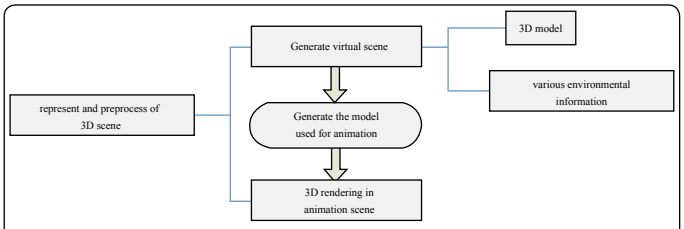

Figure 1: Octree of 3-D object and the number.

***Corresponding author:** Guofeng Qin, CAD Research Center, Tongji University, Shanghai 200092, China, E-mail: gfqing@tongji.edu.cn

Received October 19, 2010; **Accepted** December 16, 2010; **Published** December 18, 2010

Citation: Qin G, Jiang N (2010) Rendering for 3D Animation Based on Octree. J Comput Sci Syst Biol 3: 117-122. doi:10.4172/jcsb.1000067

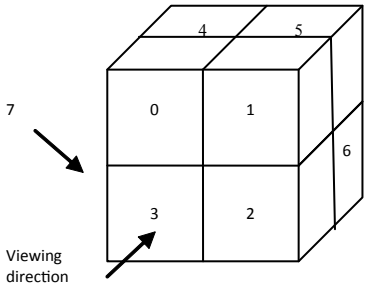

Figure 2: Octree of 3-D object and the number.

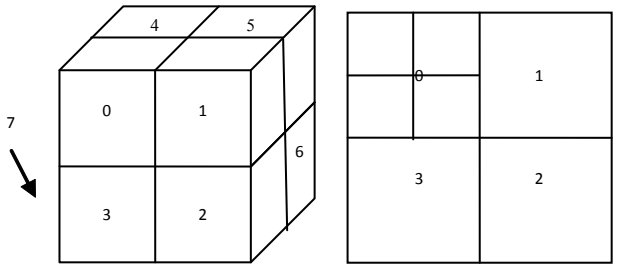

Figure 3: Releationship between Octree and quadtree.

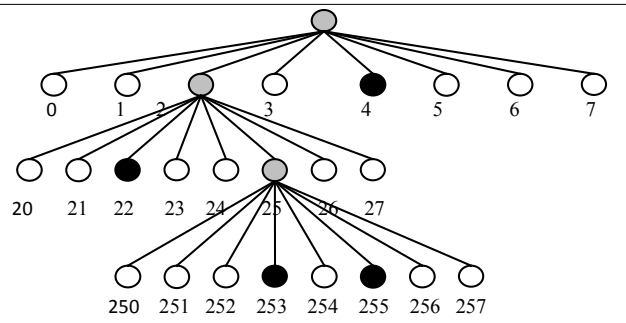

Figure 4: The structure of Octree.

on the back side, the surface is not visible and should be hidden; the other hand is visible.

(2) Ray tracing: This method requires analysis of every point on the display screen, each point must be given to reflect the brightness of the object surface reflectance properties and characteristics of the color parameters.

By the method of image space

(1) Z-buffer: First, we should build a large buffer to store all the values of color on 2-D plane along the Z-axis, as a result, it is called Z-buffer. The number of cells in Z-buffer is equal to the number of pixels on the screen and they are correspondence.

(2) Scan Line Algorithm: Set the screen with the color of the background, sort the object from far to near from their point of view, so it creates a priority list of depth. And then draw objects from far to near, as a result that the near object covers the far one.

Because of the traditional algorithms' insufficiency in animation rendering, referring to some applications of Octree in the field of blanking, an improved Octree algorithm was studied to be suitable for computer animation rendering (Li and Lu, 2001; Xiao et al., 1998).

Octree

Construction of octree

The composite Octree algorithm is suitable for organizing the 3-D scene and blanking. Objects Octree is a hierarchical data structure that this data structure greatly simplifies the hidden surface elimination (Yamaguchi et al., 1984; Sampath and George 2010). In the Octree structure, objects have been arranged in a certain order according to space, it is showing as Figure 2.

First, build an external cube that contains all 3-D objects and then divide the cube into eight equal sub-cubes (Hengshan et al., 2005). If the child has the same unit cube, which means that the cube is full or empty, we can stop dividing the cube. For the inconsistent sub-cube unit, it requires further decomposition and then split it into eight sub-cubes, until all the sub-cube is the same or it has reached the required decomposition of precision. According to this method, we can divide a scene to an Octree structure.

After the Octree structure has been build, we should map it to the plane, we should test its visibility. For a given observation of a certain direction, some sub-cubes are visible and others are invisible. When the sub-cube with one side is visible, we call the space the front node; on the contrary, if the sub-cube with one side is invisible, call it the back node. The child or children in the back of the tree behind the front ones will be covered by the front ones.

The relationship between 3-D structure and 2-D structure is shown as Figure 3. Looking through the perspective directions, the sub-cube in front of the plane, such as 0 、1、2、3 will be mapped firstly. When the plane has not been completely occupied, it should be divided into four surfaces. When the front ones are empty, the back ones will be mapped in instead of them.

According to the Figure 3, for the observer, node 0 1 2 3 are front nodes, node 4 5 6 7 are back one. When traverse the Octree by the order from front to back, the back node will be eliminated. In the way, node 0 1 2 3 will be first traversal in front of node 4 5 6 7. According to this order to travel all the Octree, children of node 0 1 2 3 will must be traversal in front of children of node 4 5 6 7.

Data structure

Construct the following structure according to the 3-D scene showing as Figure 4.

Comment :
● Entire occupied
◍ Part occupied
○ Empty
The data structure of Octree can be seen as follow.

```
Struct tOctree{
        Int id;
        Status status;
Int color;
Struct tOctree * children[2][2][2];
Boolean flag;
};
Struct tQuadtree{
Int id;
Status status;
Int color;
Stuct tQuadtree * children[2][2];
Boolean flag;
};
```

Journal of Computer Science & Systems Biology - Open Access

Research Article
OPEN ACCESS Freely available online
doi:10.4172/jcsb.1000067
JCSB/Vol.3 Issue 5

When construct the Octree, the key is to transform the coordinate in the space to the path from root to nodes in the Octree, including two facts, one is structural path when construct the Octree and the other is query path when travel the Octree. We should meet the following two needs: (1) With the storing type of the Octree, we need to find the corresponding code stored in the node according to the starting point. (2) In the display of the 3-D object, under the condition of given viewpoint position, determine the different display order, so as to achieve the blanking effect. First of all, a problem to be solved is that given coordinates of any point of space objects (x,y,z), find the linear Octree encoding of the location.

According to Figure 4, encode the Octree, linear Octree encoding is to overcome the general lack of Octree encoded to form a highly efficient coding method. Linear Octree encoding only stores the location information of the leaf nodes. Leaf node is encoded as address code, address code is commonly used in the implicit leaf node location and size information. For dividing an $2^n \times 2^n \times 2^n$ Octree, use the characteristics of octal code. In the Octree model, the location of a node can be uniquely determined by an octal number. Given any point $V_q(x_q z_q)$ in space, its path in the Octree is $P_q = P_{n-1} P_{n-2} \ldots P_1 P_0$.

$$Q_8 = \sum_{i=0}^{n-1} P_i 8^i$$

P_i is the octal number, $P_i \in [0, 7]$, $i \in [0, n-1]$. P_i indicate the number between the siblings of the node. By this way, we can express the full path in the Octree from P_0 to P_{n-1}.

We will quote a shift calculation, calculating the encoding of any point in the Octree through shift operations.

z_{n-1} y_{n-1} x_{n-1} $P_{n-1} = z_{n-1}2^2 + y_{n-1}2 + x_{n-1,}$

y_i x_i $P_i = z_i 2^2 + y_i 2^2 + x_i$

y_0 x_0 $P_0 = z_0 2^2 + y_0 2 + x_0$

z_0 y_0 x_0 $P_0 = z_0 2^2 + y_0 2 + x_0$

In order to calculate z_i, y_i, x_i, what we need do is to do some shift calculate on the z_q, y_q, x_q. In the implementation, the logical bit operation can be completed. When we know a point (z_q, y_q, x_q), we can find the back point (z_m, y_m, x_m) through the viewing direction and then find its corresponding path in Octree easily.

By calculating the number of Octree levels, we can not only find this leaf node in Octree, but also calculate the size of leaf nodes.

$L \times W \times H = (a/2^n) \times (b/2^n) \times (c/2^n)$

'n' means decomposition frequency, 'a' mean Length, 'b' means Width, 'c' means Height.

Blanking algorithm for animation rendering

When the viewing direction has been changed, one frame will have large amount similar with the front one. Shown as Figure 5, the red squares occupy 2B and 3B in picture a. After it has changed, they occupied 2C and 3C in picture b and the other part stay unchanged. In this condition, we can only change 2B, 2C, 3B, 3C.

In order to achieve this goal, now, we define the value of flag in the data structure of the Octree. If one node in Octree have been changed, set the value of the flag of this node with 1, otherwise set it 0. If one node changed, set the flag of all it parents node with 1, until the root node. Till here, we have got the Octree structure of next frame, according to this structure and when the frame and the next frame is the same, we can skip the romance and keep the color of the picture unchangeable; if the next frame has change, map it onto plane.

The blanking Algorithm for Animation Rendering can described as follow.

Input: position value (z_i, y_i, x_i) of points, $i \in [0, n-1]$; the viewing direction.

Output: set color of the planes

Procedures of the blanking Algorithm:

(1) Initialize the 3D scene, getting the orientation value of the view point;

(2) Judge the orientation value of the new view point by comparing with the old view point. If the view point has been changed, re-rendering of the scene will be started and go to step (3), else go to step (12);

(3) Construct Loops for each plane of the quad-tree, define the front and back node corresponding to the plane;

(4) Judge whether the front node is an entity. If the front node is not an entity, go to step (5), else go to (6);

(5) Sub-divide the front node;

(6) Judge whether the front color is empty. If the front color is not empty, go to step (7), else go to (8);

(7) Assign the color of the front node to the plane;

(8) Judge whether the back node is an entity. If the back node is not empty, go to step (9), else go to (10);

(9) Assign the color of the back node to the plane;

(10) Set empty to the color of the plane, go to (3);

(11) Sub-divide the back node, go to (8);

(12) Keep up with the same data of color, end the rendering program.

The process flow of the improved Octree algorithm can be seen in Figure 6.

Experiment and result analysis

The experiment is operated on the platform that is AMD Turion(tm) 64 Mobile, Technology MK-36 1.60GHz, 1.00GB Memory, Windows XP SP3 and the develop tool is VS2008. The max depth of

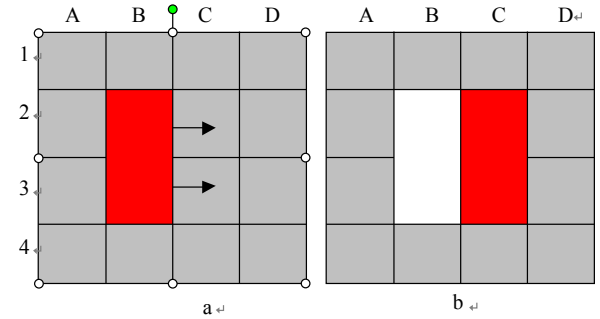

Figure 5: Before and after the 2D graph change.

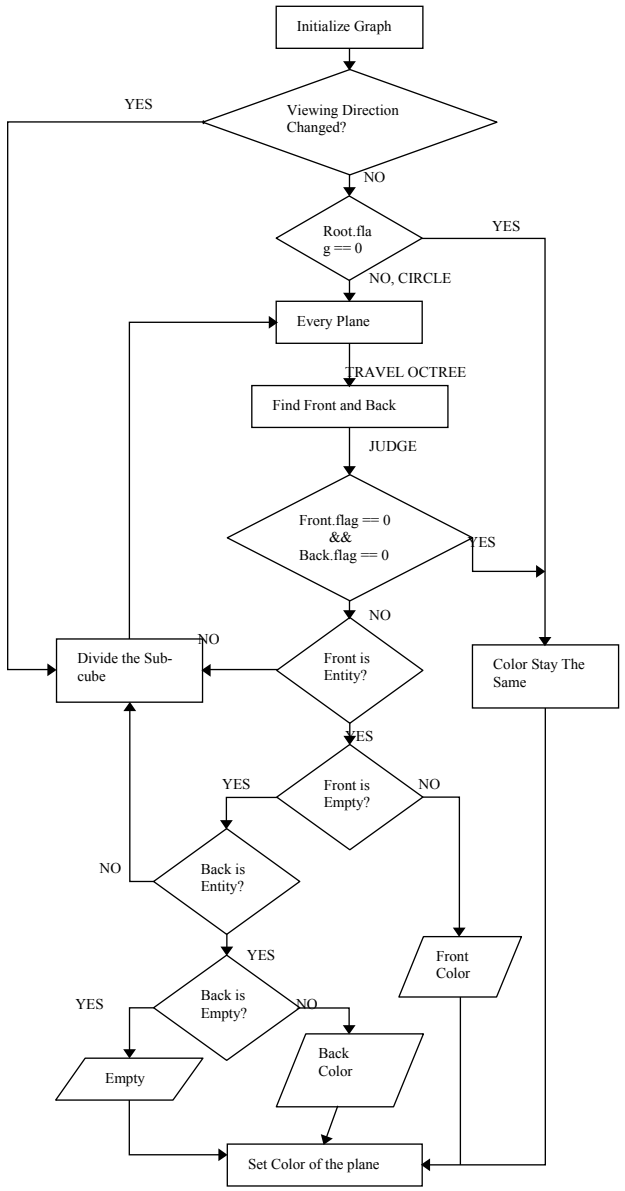

Figure 6: The flow chart of Octree algorithm.

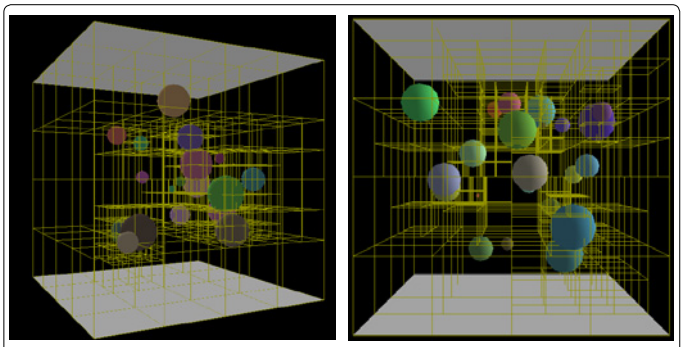

Figure 7: Octree Experiment.

the Octree is 6. The number of the balls in the scene can be controlled by the blank keyboard. Stat the time used to render the whole scene. Time of the process is calculated in Micro-second (us). The details of the Octree experiment can be seen in Figure 7.

The results of the improved Octree algorithm compared with the common Octree algorithm also can be seen in Table 1 and in Figure 8.

In the above data, the max depth of the Octree is 6, so the scene is simple to render. The number of nodes in Octree is not so large to maintain, the different between the common algorithm and the optimized one is not very clear. When there are 80 balls in the scene, the common algorithm is 7.8us slower than the optimized one; in the experience, there are 200 balls in the scene, the difference is more than 41us. It is clear above. But the time to render both grows quickly with the number of balls increasing, because it takes a long time to maintain the structure of the Octree.

Because the Octree make good use of physical correlation in space, this method has more advantages compared with others.

(1) Reduce the space of storage: In general, this method provides

Method (us)	Number of Balls							
	20	***40***	***60***	***80***	***100***	***120***	***160***	***180***
Common	31.3	57.8	96.8	140	181	243	376	395
Optimized	31.3	51.6	78.1	132	170	218	329	359

Table 1: Time consuming.

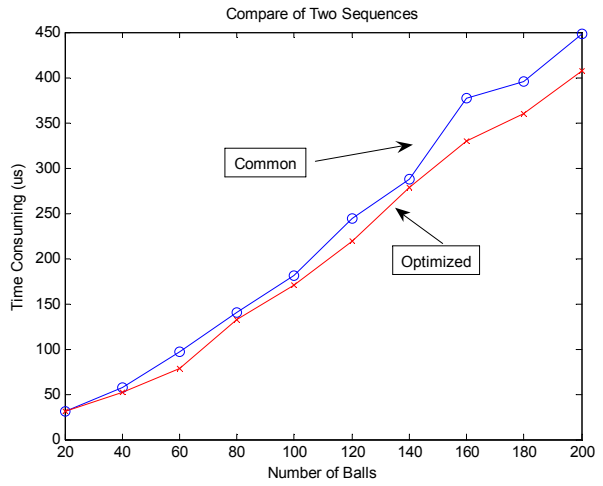

Figure 8: Compare the two sequences.

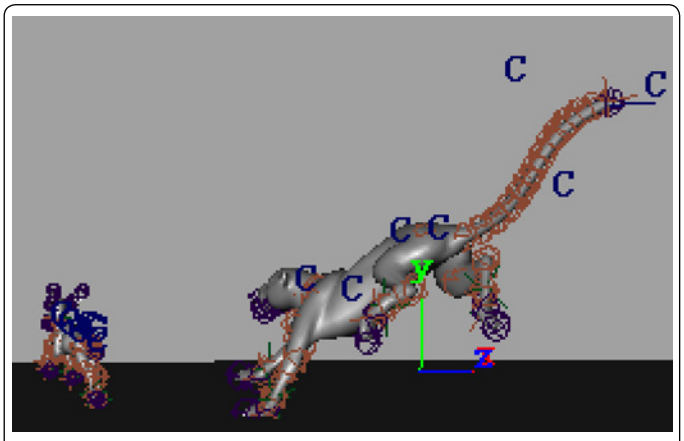

Figure 9: Animals' motion animation rendering.

Journal of Computer Science & Systems Biology - Open Access

www.omicsonline.com

Research Article
OPEN ACCESS Freely available online
doi:10.4172/jcsb.1000067
JCSB/Vol.3 Issue 5

Figure 10: Human motion animation rendering.

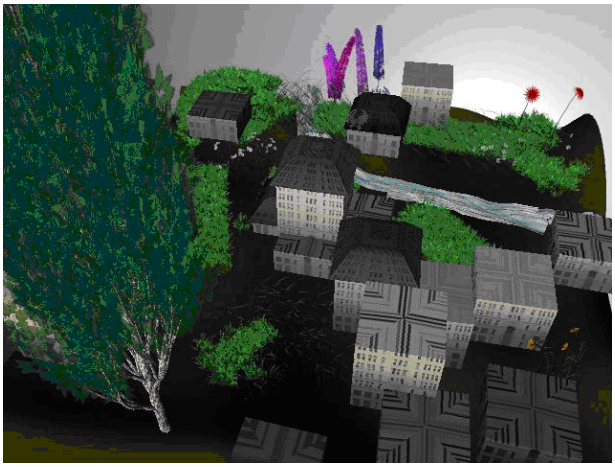

Figure 11: City planning animation rendering.

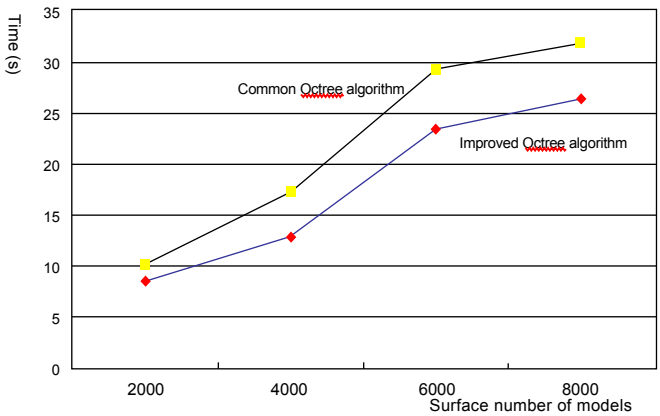

Figure 12:

a storage mechanism for the compression of space objects. Under the condition that consistency in the pixel is great, it can effectively reduce the amount of physical storage space required.

(2) Facilitate to achieve set operations: In other methods, we need to spend a lot of computing resources for set operations. However, by using the Octree, translate the set operations, such as and, delivery, into simple operations of the element.

(3) Conducive to physical operation of display: the Octree has the characteristics of a hierarchical structure and has order. As a result, it is suitable to remove hidden surface and so on.

However, there are also disadvantages about this method. In small scene, it will take a long or more time to construct an Octree. And when the objects in scene are very small, we need a deeper Octree to organize the scene, so it will take up a great memory and more time to maintain the memory. Here is to be improved.

Appliction of improved octree algorithm

The improved Octree algorithm is tested with many projects. Their results verified its good performance, including saving up memory and decreasing time cost. There are three samples. The first one is animals motion animation rendering, its time can decrease about 11% time cost and save up about 8% memory with the improve Octree algorithm. The rendering details of animals' motion animation can be seen in Figure 9. The second one is Human motion animation rendering, its time can decrease about 12.5% time cost and save up about 9.5% memory with the improve Octree algorithm. The rendering details of Human motion animation can be seen in Figure 10. The third one is city planning animation rendering, its time can decrease about 13.5% time cost and save up about 10.5% memory with the improve Octree algorithm. The rendering details of city planning animation can be seen in Figure 11. There exists difference among three samples because there are different number objects in the rendered scene, and the different objects have their different geometric graphic model, including surface grid and vertex. All in all, the memory will be saved up more, and the time cost will be decreased more if the objects are more and their models are more complicated.

With experience of animals' motion animation rendering, human motion animation rendering and city planning animation rendering, their results indicate if there are 2000 surfaces in scene, it will take 10 seconds to render the models by the common Octree algorithm, but it will take 8.7 seconds by the improved Octree algorithm, which decrease about 13% time; if there are 4000 surfaces in scene, it will take 15.8 seconds to render the models by the common Octree algorithm, but it will take 13.5 seconds by the improved Octree algorithm, which decrease about 14.5% time ; if there are 6000 surfaces in scene, it will take 28.9 seconds to render the models by the common Octree algorithm, but it will take 24.2 seconds by the improved Octree algorithm, which decrease about 15.5% time; if there are 8000 surfaces in scene, it will take 31.2 seconds to render the models by the common Octree algorithm, but it will take 26.3 seconds by the improved Octree algorithm, which decrease about 15.7% time. With increasing number of surface in scene, the linear tendency will become gradual, which describes ability and effect of the improved Octree algorithm is non-linear. The details of these results can be seen in Figure 12.

Conclusion

For the traditional algorithms' insufficiency in animation rendering, an improved Octree algorithm was studied to be suitable for computer animation rendering. The entire 3-D scene was be divided into break down the entire 3-D scene and then setup the Octree structure. Second, transform the Octree onto a 2-D plane structure by the improved Octree structure. The experience

verified the result of the improved Octree algorithm, the time cost is decreased about 41us for 200 balls rendering from the common algorithm. By experience of animals' motion animation rendering, human motion animation rendering and city planning animation rendering, their result indicate the improved Octree algorithm can decrease the rendering time, but its rendering ability and effect ability and effect is non-linear. In the future, the stabilization of the new Octree algorithm still should be optimized.

Reference

1. Geng G, Zhou M (1996) A Simple Algorithm for Transformation from an 3D Object to an Octree. Journal of Northwest University 26: 4.
2. Hengshan Wu, Duan X, Chenyang LI (2005) Algorithm for neighbor searching of leaf-coding quadtree. Computer Application in chinese 25: 2624-2626.
3. Karunakaran KP, Shringi R, Ramamurthi D, Hariharan C (2010) Octree-based NC simulation system for optimization of feed rate in milling using instantaneous force mode. International Journal Of Advanced Manufacturing Technology 46: 465-490.
4. Lee YL, Abraham A, Kim DH (2010) 3D OBJECT RECOGNITION USING OCTREE MODEL AND FAST SEARCH ALGORITHM. Neural Network World 20: 359-369.
5. Li J, Lu Y (2001) Study of Hiding Technology about 3D Drawing. Journal of LiaoNing Provincial College of Communications in chinese 3: 4.
6. Ma Z, Ma J, Dai L (2006) An algorithm for hidden surface based on linear Octrees. Ningxia Engineering Technology in chinese 5: 3.
7. Sampath RS, George B (2010) A parallel geometric multigrid method for finite elements on Octree meshes. SIAM Journal on Scientific Computing 32: 1361-1392.
8. Xiao L, Gong J, Xie C (1998) A new algorithm for searching neighbors in the linear quadtree and Octree. Journal of acta geodaetica cartographica sinica in chinese 3.
9. Yamaguchi K, Kunii TL, David RF, Francisco RA (1984) Computer-integrated manufacturing of surfaces using Octree encoding. IEEE Computer Graphics and Applications 4: 60-65.
10. Zhou D, Yang R (1993) An optimal construction algorithm for linear Octrees. J Computers in chinese 16: 4.