Progressive Transmission of Cell Octrees

F. Velasco J.C. Torres A. León
Computer Graphics Research Group
Department of Software Engineering - E.T.S. of Computer Science
The University of Granada, Granada, Spain E-18071
fvelasco@ugr.es jctorres@ugr.es aleon@ugr.es

ABSTRACT

This paper presents a technique to carry out progressive transmission of a volume data set using a hierarchical index of the data set. The intermediate data transmitted can be used as a complete representation, allowing the client to perform any computation on the volume and to generate any isosurface. The experiments performed show that a good approximation can be achieved for medical data sets with less than ten per cent of the data transmitted. The hierarchical index used is a compressed BONO index (cell octree), that is also used to speed up the isosurface generation.

Keywords

Volume modelling, progressive transmission, multiresolution.

1. INTRODUCTION

Volumetric data are usually represented by a set of property values v_i measured in a point set $(x_i,y_i,z_i) : i = 1,2,..,N$; these values are samples from an unknown function $f(x,y,z)$.

One of the most commonly performed operations is visualization, and the well known *Marching Cubes* method [Lor87] is often used. By this method a function $F(x,y,z)$ is defined to estimate the unknown function $f(x,y,z)$; a threshold value γ_k is fixed and the surface $F(x,y,z)= \gamma_k$ is built as the geometry to visualize the volume. The volume must be represented by a structured regular grid (which can be built using the function $F(x,y,z)$). Then, every cubic cell is classified as one of 15 possible cases by comparing the threshold value with the values of the cell's vertices. Fourteen cases (called active cells) have a predefined triangulation that represents the isosurface inside the cell. By joining all the pieces of isosurfaces, the whole isosurface is rendered.

Although this method is simple and works well, several improvements have been suggested, such as [Lop02, Cig00, Che95], in order to solve some ambiguous cases; [Wil92, Cig97, She96], in order to look for active cells, and [Sch92, She96, Baj96, Shu95] to reduce the mesh size.

We have presented a representation schema on the basis of using a multiresolution grid. Bigger cells are used where the data are uniform and higher resolution cells are used where the data require it. This schema is called **Cell Octree** [Vel02a, Vel01].

In this paper we propose a method to carry out progressive transmission of a volume over the net using cell octrees, in such a way that the receiver receives the volume in different stages, increasing the resolution in each of them but with a useful volume from the start.

This paper is organized in the following way: section two summarizes cell octrees, section three describes our proposal for the progressive transmission of cell octrees, and subsequent sections discuss some details of high level implementation, the results obtained with real volumes and draw some conclusions.

2. CELL OCTREE

A cell octree is a data structure that is formed by a structured regular grid and an octal tree that indexes the grid. We use a bono [Wil92] as the start point and by a bottom-up process, several prunes are carried out in the tree; with every prune, eight **brother** leaf nodes are constantly pruned and the former **father** internal node is converted into a new leaf node. With every prune, the eight cells represented by the pruned leaf nodes are no longer accessible and a new bigger

WSCG POSTERS proceedings
WSCG'2003, February 3-7, 2003, Plzen, Czech Republic.

cell that is represented by the new leaf node is now accessible (see figure 1).

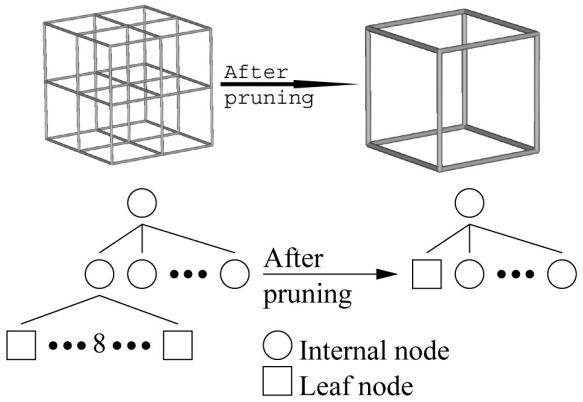

Figure 1. Pruned cells and the new cell. Its representation at the tree

In [Vel02a] several pruning criteria are presented. A summary version of the proposal, in English, is available in [Vel01].

3. VOLUME TRANSMISION ON NET

Today, it is usual for systems to be interconnected by a network in order to enjoy advantages such as resource sharing, information centralization, information sending to/receiving from other systems, etc.

In the volume visualization field, Engel et al. propose a server-client volume visualization system that allows a less powerful computer to visualize a large volume that would be stored in a more powerful computer. This latter computer would perform the computations and give the client the information needed to carry out the visualization [Eng99].

The visualization process is divided into 7 steps and the server-client system classified into 6 cases, determining which steps are performed in the server and which in the client.

The authors define the following steps of the visualization process: **storing,** there is no process, only a stored volume; **filtering,** the threshold value is fixed and the active cells are found; **interpolation,** the vertices of the triangle are computed for each active cell; **triangulation,** the triangle meshes are built using the previously computed vertices; **projection,** the 3D triangles are projected to be rendered; **raster,** the 2D triangles are filled pixel by pixel on the frame buffer; and **rendering,** the frame buffer is dumped to the output device.

These 7 steps allow us to have 6 classes of server-client system (Table 1; **Ci** means *"Class number i"*, **S** means that the step is performed on the server, and **C** means that the step is performed on the client).

Step	C1	C2	C3	C4	C5	C6	C7
Storing	S	S	S	S	S	S	C
Filtering	S	S	S	S	S	C	C
Interpolation	S	S	S	S	C	C	C
Triangulation	S	S	S	C	C	C	C
Projection	S	S	C	C	C	C	C
Raster	S	C	C	C	C	C	C
Rendering	C	C	C	C	C	C	C

Table 1. Classification made by [Eng99] and our proposal (class 7)

The information that is transmitted is an image for class 1, 2D triangles for class 2, 3D triangles for class 3, the vertices of the triangles for class 4, active cells for class 5, and a volumetric model for classes 6 and 7.

The classes proposed by [Eng99] (1 to 6 in Table 1) have two steps in common: visualization is always performed on the client and storing is always on the server. Thus, when the threshold is changed or, in some classes the point of view is changed, a new transmission is required. We propose (class 7 in Table 1) to send the whole volume but in a progressive way, so that the client always has a visualizable volume from the start. Even though the resolution is poor, the client can change the threshold and the point of view, getting quick visualizations without asking the server for new sendings. The resolution and the quality of the visualizations improve progressively, until the maximum resolution requested by the client. This proposal can be combined with a selective transmission, so the user, after the first low resolution sending and after selecting the threshold, can select a point of interest and then only a subvolume will be resent at high resolution.

4. IMPLEMENTATION

In order to achieve progressive transmission of a cell octree, both the grid and the tree need to be transmitted by the server and received by the client. Moreover they need to be synchronized. With our proposal, no extra information is needed to carry out progressive transmission.

The next three subsections show how our algorithm can send/receive a cell octree: the first one shows the transmission of the grid, the second one shows the transmission of the tree and the third one shows the overall transmission algorithm.

Transmission of the grid

The transmission of a volume must begin with the sending of its size: MAXX, MAXY and MAXZ. With these data, both sender and receiver compute a value, called interval which leads and synchronizes the transmission. The initial interval value is computed as:

$$Interval = \min \{x=2^n : n \in N \wedge x \geq \max \{MAXX, MAXY, MAXZ\}\}$$

Interval defines the distance between two consecutive values on the grid for a particular resolution. This is divided by two when the level changes until it is one for the higher resolution.

The transmission pseudocode for the first level is shown at [Vel02b]. The reception pseudocode is similar.

The Interval parameter allows us to send the values needed by a level and avoids sending each value more than once by using several boolean variables.

[Vel02b] gives the transmission pseudocode for the second level and subsequent ones. The reception pseudocode is similar.

Transmission of the tree

To transmit the tree it is necessary to do it level by level, so the tree must be traversed and sent in a first-breadth way. A data structure *queue* is thus needed. Moreover, a mark, called beginLevel, is needed in order to know when a level has finished and another one is beginning.

The queue can store tree nodes and the mark beginLevel as elements. The functions to manage this are: emptyQueue(), which returns true if the queue is empty; lastQueue(element), which puts the element as the last one of the queue; firstQueue(element), which returns the first element of the queue. This element is then eliminated from the queue.

The pseudocode to send the tree is also shown in [Vel02b].

The pseudocode to receive the tree is similar, but the commands to access the children of a node are replaced by commands to create them.

Progressive transmission of a cell octree

The algorithm to transmit the volume is similar to the one used to transmit the tree but it is modified to perform a grid point transmission before transmitting a level of the nodes of the tree.

A cell octree has a particular leaf node type called *eight cells leaf node* [Vel02a] which allows us to access eight cells; thus the last level of the tree needs another level of grid points, although this extra level is not needed if the last level of the tree is the root node and this is not an *eight cells leaf node*. A boolean variable anotherLevel is used to manage the latter case.

The client receives values and nodes; when a node arrives (internal or leaf) it is tested. If the threshold is between the maximum and minimum value stored in the node, the triangles for this node will be built.

The client has a list of triangles for each level in order to store the triangles that have been built from the leaf nodes of that level and the eight cells leaf nodes of the previous level. These triangles are reusable on subsequent levels. Only the triangles built from internal nodes are not reusable on subsequent levels.

A variation with respect to the tree reception algorithm shown in the previous section is that the *eight cells leaf nodes* are pushed in the queue too. Thus, they are found at the next level, and their eight cells are processed.

Finally, note that although the reception pseudocode performs the visualization when every node arrives, the visualization can be independent of the reception.

5. RESULTS

The proposed idea has been implemented and tested with a particular volume, providing data about the size of the information transmitted on every level, about the visualization time for every level and about the number of triangles built on every level. Images of the volume are also shown.

The size of the volume is 220 x 230 x 220. Data and images are shown in Table 2 and in Figure 2.

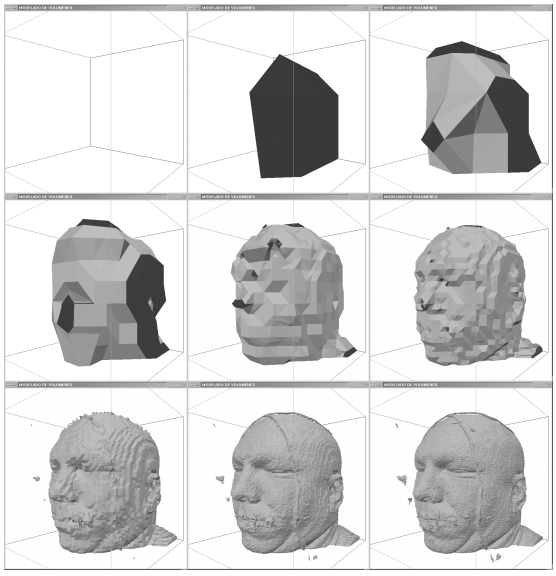

Figure 2. Images at every level

In Table 2, the first size column is the size that is sent on every level, whereas the second size column is the total size been sent so far. The two columns about the time reflect the time spent building the triangles and that spent projecting them.

Level	Size (KB)	Total size (KB)	Time to build (ms)	Time to project (ms)	Triangles
0	0.01	0.01	0.0	0.0	0
1	0.08	0.09	0.0	0.4	14
2	0.61	0.70	0.1	0.8	85
3	4.64	5.33	0.3	1.5	386
4	28	33	1.6	3.5	1,702
5	182	216	7.0	8.8	7,640
6	1,114	1,330	29.7	30.1	34,050
7	7,168	8,498	125.1	115.7	150,576
8	21,383	29,881	694.6	459.0	658,766

Table 2. Transmission data

Note that with little more than 1 Mbyte (up to level 6) an acceptable image is visualized while spending only a short time. A user can work with the volume knowing the volume is available locally, and so a change of the threshold or of the point of view does not need new transmissions. A new transmission will be performed when a higher resolution is needed. The system, thus, allows quick visualization of complex volumes.

6. CONCLUSIONS AND FUTURE WORK

This paper shows how to manage the progressive transmission process of a volume using cell octrees. The method has been proved to be efficient, allowing the client to work with the volume when very little information has been transmitted.

The data structure can also be used to reduce the whole volume level of detail to achieve interactive rendering when the model is being manipulated.

As future work, we plan to investigate the use of the representation to obtain multiresolution representations dynamically, taking into account the viewpoint or interest area of the user.

7. ACKNOWLEDGEMENTS

This paper has been supported by the *Ministerio de Ciencia y Tecnología* and by *FEDER* through grant TIC2001-2099-C03-02.

8. REFERENCES

[Baj96] Bajaj, C.L., Pascucci, V. and Schikore, D.R. Fast isocontouring for improved interactivity. In ACM Symposium on Volume Visualization, pp. 39-46,99, San Francisco, USA, 1996.

[Cig97] Cignoni, P., Marino, P., Montani, C., Puppo, E. and Scopigno, R. Speeding up isosurface extraction using interval trees. IEEE Transactions on Visualization and Computer Graphics, 3(2):158-170, 1997.

[Cign00] Cignoni, P.,Ganovelli, F., Montani, C. and Scopigno, R. Reconstruction of topologically correct and adaptive trilineal surfaces. Computer and Graphics, 24(3):399-418. 2000.

[Che95] Chernyaev, E. Marching cubes 33: construction of topologically correct isosurfaces. Technical Report CN/95-17. Available as http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz, CERN.

[Eng99] Engel, K., Westermann, R. and Ertl, T. Isosurface extraction techniques for web-based volume visualization. In IEEE Visualization, pp. 139-146, San Francisco, USA, 1999.

[Lop02] Lopes, A. and Brodlie, K. Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing. IEEE Transaction on Visualization and Computer Graphics. Accepted but not published yet. 2002.

[Lor87] Lorensen, W.E. and Cline, H.E. Marching cubes: A high resolution 3D surface construction algorithm.ACM Computer Graphics, 21(4):163–169, 1987.

[Sch92] Schroeder, W.J., Zarge, J.A. and Lorensen, W.E. Decimation of triangle meshes. ACM Computer Graphics, 25(2):65-70, July 1992.

[She96] Shekhar, R., Fayyad, E., Yagel, R. and Cornhill, F. Octree-based decimation of marching cubes surfaces. In Visualization'96, pp. 335–342,499, San Francisco, USA, 1996. IEEE.

[Shu95] Shu, R., Zhou, C. and Kankanhalli, M.S. Adaptive marching cubes. The Visual Computer, 11:202-217, 1995.

[Vel01] Velasco, F. and Torres, J.C. Cell octree: A new data structure for volume modeling and visualization. In VI Fall Workshop on Vision, Modeling and Visualization, pp. 151-158, Stuttgart, Germany, 2001.

[Vel02a] Velasco, F. Representation and Visualization of Volumetric Data. Ph.D. thesis, University of Granada, Granada, Spain, 2002.

[Vel02b] Velasco, F. Pseudocodes that have been referred to in this paper. http://giig.ugr.es/~fvelasco/code/wscg03/

[Wil92] Wilhelms, J. and Van Gelder, A. Octrees for faster isosurface generation. Extended abstract. ACM Transactions on Graphics, 11(3):201-227, 1992.