Progressive Transmission of Large Archaeological Models

F.J. Melero, P. Cano and J.C. Torres

Depto. Lenguajes y Sistemas Informáticos,
Universidad de Granada, Spain
{fjmelero, pcano, jctorres}@ugr.es

Abstract. One of the most important problems nowadays in virtual archaeology is the fact that scanning real models produces huge amount of polygons, and these polygons must be rendered and displayed in a interactively way, trying to minimize the hardware requirements.
In a global world, internet has become the most important way of knowledge, and archaeology is one of the most exciting knowledge sources. But due to the high resolution of the reconstructed model, it is necessary a huge bandwidth to send the full mesh across the net, and, usually, it's necessary to have the full mesh to reproduce the model.
In this paper, we present a multiresolution scheme to represent the model, based on a modified octree structure, called SP-Octree (Space Partitioning Octree). Using this structure, it is possible to obtain an approximate representation of the model with a few bytes, and by descending through the SP-Octree, the model is refined step by step, having at each level a more accurate representation of the model.
This approach is really useful for net transmission, because if the object is not close enough to the observer, it is possible from the first bytes to simulate the real model by applying impostors.
Keywords: Computer Graphics, Solid Modeling, Multiresolution, Adaptive Visualization, Visualization or Archaeological Sites

1. Introduction

The interactive visualization of three dimensional models is traditionally one of the most important applications of Computer Graphics in Archeology.
The interactive visualization of complex models requires a great effort of the graphics hardware, due to the large number of polygons needed to represent the geometry at the expected level of detail and realism. This resources requirement may leads to a lack of interactivity.
The conflict between the rendered level of detail and the visualization speed has motivated the development of several techniques to reach both goals.

2. Level-of-Detail

Under the generic name of level-of-detail (LOD) techniques we gather techniques that represent a complex geometry in a simplified way when the observer is not close enough to the object (Fig.1).

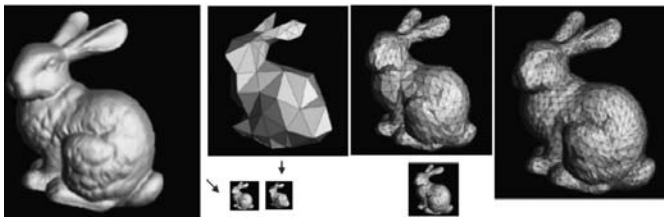

Fig. 1. Bunny at highest level of detail (left). Rest of images show several LODs of the same object. Note as the bunny at lowest resolution is not very different from that at highest resolution when both images are small enough.

2.1 Geometry-Based LOD Techniques

Geometry-based LOD techniques can be classified as: discrete (Clark 1976), i.e. the object is represented in several instances, each one at different LOD; progressive (Hoppe 1996), i.e. the detail is extracted from an unique data structure during runtime; and view-dependent (Hoppe 1997),(Xia and Varshney 1996), which are an extension of progressive techniques, in a way that the LOD is not uniform along the object, but it is anisotropic depending on the point of view.
Although the LOD concept is not restricted to a concrete area (images, solids, volumes...), in three dimensional computer graphics it is usually applied to the simplification of polygonal meshes, mainly triangular ones.
Another approach to represent volumes and solids with variable LOD is to represent them with schemes based on spatial decomposition, using hierarchical structures. Among these methods we can find the Octree model and several extensions to this, like the Extended-Octrees (Brunet, P. and Navazo, 1990).

2.2 Image-Based LOD Techniques

Another approach to solve the conflict between realism and interactivity, are the image-based techniques. The idea under this approach is to replace a complex three dimensional geometry with a 2D image that shows what should be represented with the removed geometry. This image is an impostor.
Many image-based techniques have been presented to represent solids (Maciel and Can Shirley 95), (Aliaga 96), (Sillion, Drettakis and Bodelet 1997), and all of them can be considered as view-dependent. The complexity of the

geometric model where impostors are applied moves from a single plane to triangular meshes.
In Fig 2 we can observe as the complex geometry of the building is substituted by a very few triangles, and when applying the impostor, there is not difference between high computational cost model (upper left) and low cost simplified model (bottom rigth).

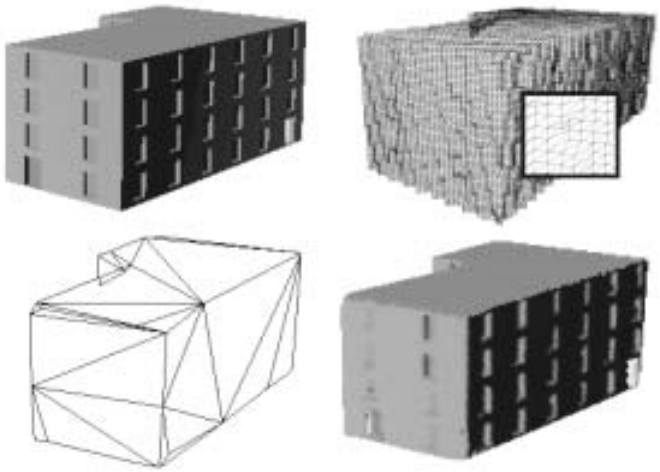

Fig. 2. Model of a building at highest resolution and its textured rendering (upper model), and simplified geometric model (left-bottom), combined with an impostor (right-bottom).

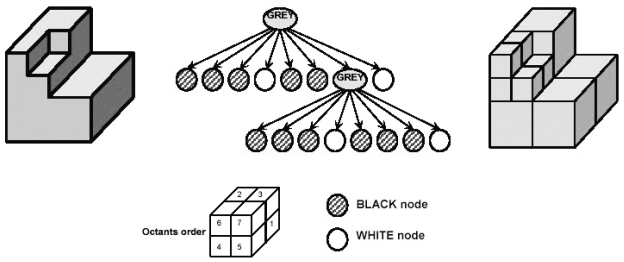

Fig. 3. Original solid (left), octree structure (centre) and octree representation (right).

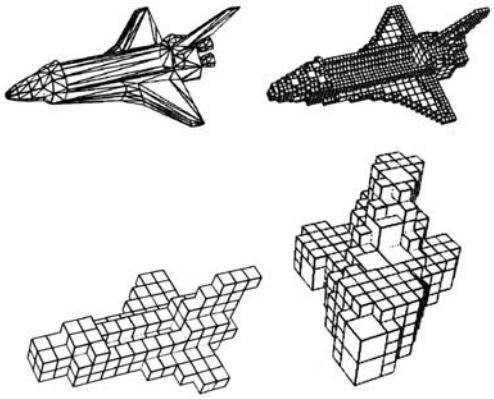

Fig. 4 Original solid (upper left), and different resolution octree-based representations.

3. Progressive Transmission

In recent years there has been an extraordinary development of networking technologies, being nowadays very frequent to work in distributed system, where we can find one or more servers and a lot of clients retrieving data from the servers.
If we want to work in such distributed way, storing a three dimensional model of any object in a very powerful server, and visualizing and editing it in our PC, which acts as client, we have two options:

- Wait until the whole model is received in our client to start our work, which could take us several minutes (depending on the network load)
- Receive in the first network packages a very simplified model of the object we want to work with, visualize it, and acquire more detail step by step, depending on our requirements, in a transparent way.

It seems clear that the best option is the second one, but it is necessary that the three dimensional model is represented in such scheme that the progressive and adaptive transmission of the geometry do not requires more time than the transmission of full model, i.e., it is necessary to be as non-redundant and efficient as possible.
Polygonal meshes are one of the most broadly used techniques to store complex 3D models. Polygonal meshes allow approximating any surface, reducing error by using smaller polygons (usually triangles). The first approach to this problem was the progressive mesh. This approach is not designed for managing with solid models, so could appear non valid models when simplifying the object.
Another approach to represent a solid model is to use a hierarchical structure, usually a tree, in such way that each level-of-detail is represented by the levels of the tree.
Among these structures, we can find:
Octrees. The idea is to subdivide the bounding box of the solid into eight equal octants (voxels), and classify each voxel as black, if it is completely inside the solid, white, if it is completely outside the solid, or grey, if the solid is partially included in the voxel, as shown in figure 3. Each grey node is recursively subdivided and its children nodes are classified again as before. The accuracy of the representation depends directly on the size of the deepest voxels, but it will never represent exactly the solid (Fig. 4).
BSP-Trees. The Binary Space Partitioning Tree (Thibault and Naylor 1987) divides the space by arbitrary oriented and positioned planes, in such way that these planes belong to the solid boundary, being able to classify each obtained half-space internal or external to the solid. (Fig. 5).

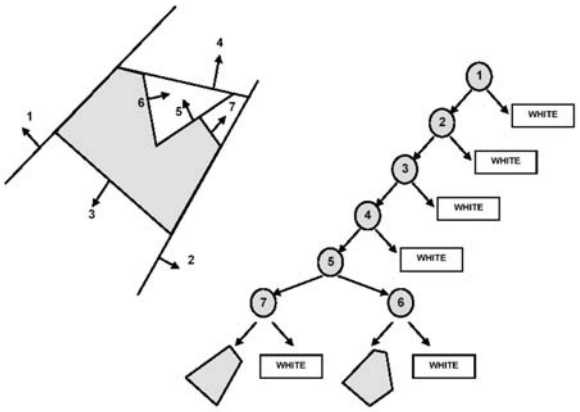

Fig. 5. BSP tree (right) representing a 2D polygon (left).

By combining these two approaches we have developed another data structure that allows us to represent in a hierarchical and exact manner any valid polyhedral solid: the SP-Octrees structure.

4. SP-Octrees

The SP-Octrees – Space Partition Octrees – (Cano and Torres 2002), include in the internal nodes information about the boundary that defines partially the represented object at that level. When descending the tree, whenever a plane is found at an upper level node, it is not necessary to repeat the data of these planes.
Basically, the idea is to classify each node once depending on the characteristics (convexity or concavity) of the planes that appear in it. Also, the internal nodes store those planes that do not change in the children nodes, so it is not necessary to repeat the information. This information allows delimiting the space where the solid is defined inside the node.
SP-Octrees use six different node types (Fig. 5):

- white node, when it is completely outside the solid.
- black node, when the node is completely inside the solid.
- convex node, if the intersection between the solid and the node is convex. The node contains the set of planes from the boundary that defines the convexity, except those that have been used at previous levels of the tree.
- concave node, if the intersection between the solid and the node is concave. In this case, the node contains the set of planes that defines the concavity.
- grey nodes contain concavities and convexities, and must be divided into eight equal octants. This node is represented by planes from the convex part of the boundary, filling the holes with the node bounding box.
- vertex nodes are used when a node contains an unique vertex where convex and concave edges converge. Vertex nodes are needed because it is not enough with categories below to cover all cases (Cano and Torres 2002).

5. SP-Octrees Visualization

In order to visualize an object represented by means of the proposed scheme, we traverse the tree level by level, representing at each node the intersection of the planes that appear in it with its surrounding box and with the planes that appear in their ancestors. In this way, as we have information of the boundary of the object in the upper nodes, the higher levels of the tree allow us to obtain quickly the convex part of the boundary of the object.
In order to draw the object faces it is necessary to trim the planes in one node against those in its descendants.

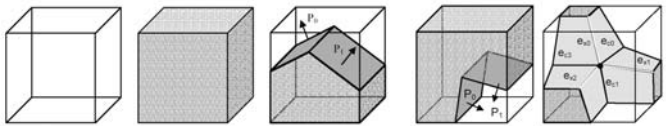

Fig. 6. White, black, convex, concave and vertex nodes.

5.1 Using Impostors in SP-Octrees Visualization

Previous mechanism allows us to make an adaptive visualization, but at intermediate levels, it offers an approximated representation of the solid, which is some times so coarse than it is difficult to perceive the shape we are trying to represent.
To solve this problem we propose to add impostors at intermediate levels to those planes that do not belong to the original shape of the object.
The set of images used as impostors is generated once in a preprocessing time. The images are taken from the vertexes of a sphere that contains the solid bounding box (Fig. 6). These images are taken by rendering the real solid at the maximum LOD. By varying the LOD of the sphere, it is possible to control the amount of images or impostors to be used.
We position the camera at each vertex of the sphere, oriented towards the centre. By using the orthographic projection, we only have to control that the object fits entirely in the projection plane. All the images are stored in a binary file, as well as information about all the visualization and transformation matrices used when the image was taken. The object was rendered with controlled background, so that transparent pixels can be easily recognized.
To render the model, the tree is traversed in a descendent way, generating the corresponding polygons for each level. If the generated polygons do not belong to the boundary, we apply the selected impostor (Fig. 7) by using Projective Texture Mapping (Everitt 2000) and OpenGL (Segal and Akeley), otherwise, if it is a boundary polygon, we apply its original texture.

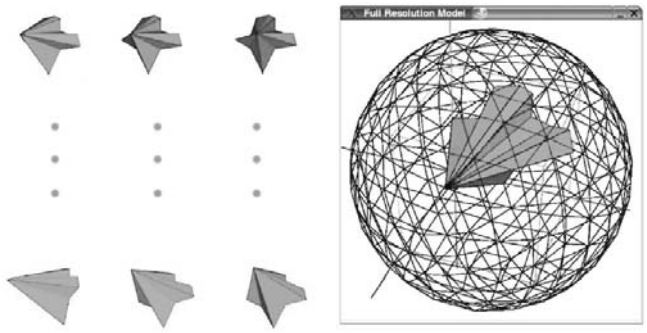

Fig. 7. Sphere used to take impostors, and examples of them (left).

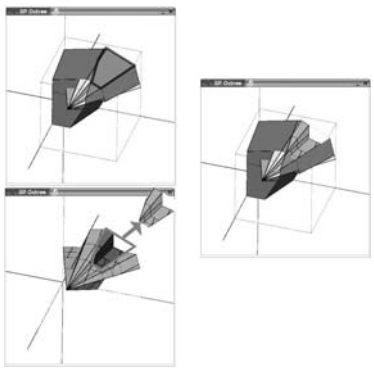

Fig. 8. Over a selected non-boundary plane (upper left image), we project the impostor (lower left image) and it is then seen as the proper shape of the model, by applying just the portion of impostor that fits in that area.

In a progressive transmission, not all the impostors are transmitted at one time. They are sent on demand, i.e. the client viewer asks for those impostors that probably will be needed in a few frames, taking into account the position and direction vector of the observer. Obviously, if the user makes a sudden change in his viewpoint, there will be a few seconds of uncertainty, until the proper impostor is received.

6. Results and Conclusions

SP-Octree is a very useful approach to make a progressive transmission (Cano, Torres and Velasco 2003) and visualization of the solid, saving space with respect to other hierarchical structures.

This paper adds to this approach an improvement of visualization capabilities, being able to recognize the represented solid from the first steps of visualization. As the observer gets closer to the object, the tree refines the nearest nodes to the camera, leaving at low resolution those that are not being seen.

In figure 8 we can see as the proper impostor is selected at each moment, and as from the first level of the SP-Octree, the user perceives the original solid.

In figure 9, it is shown the progressive visualization of a sculpture without impostors, showing as in 4th level the shape is easily recognizable, so the impostors are not more needed.

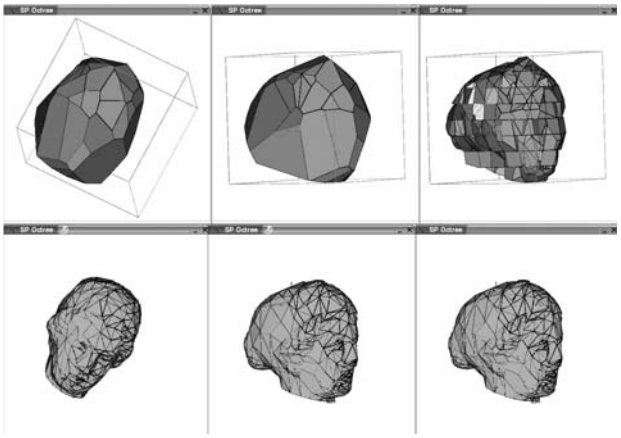

Fig. 9. First two columns: different views of an SP-Octree at level 0 (upper) and at level 0 using impostors (lower). At right, the same model at level 3, without (upper) and with impostors (lower).

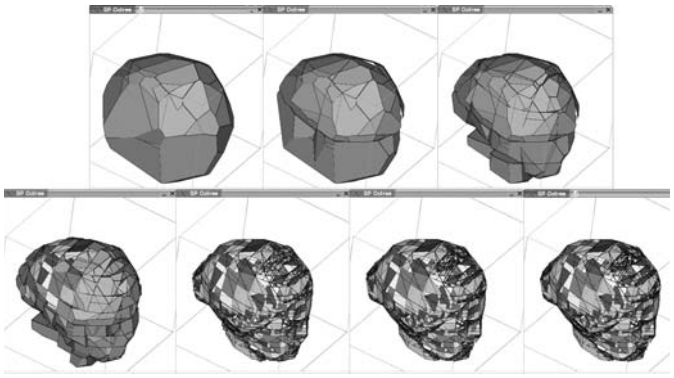

Fig. 10. Progressive transmission of a Venus' head in seven levels (first one is at left top).

These results will lead to be able to represent huge models and scenarios of archaeological sites in a progressive and adaptive way, spending less space and time than with the complete full-resolution model.

Acknowledgements

This work was partly supported by the Spanish Ministry of Science and Technology (MCYT) and FEDER under project TIC2001-2099-C03-02.

References

Aliaga, D., 1996. Visualization of complex models using dynamic texture-based simplification. In *IEEE Visualization'96*, IEEE.

Brunet, P. and Navazo, 1990. I. Solid representation and operation Using Extended Octrees. *ACM Transactions on Graphics* 9, 2. 170–197.

Clark, J., 1976. Hierarchical geometric models for visible surface algorithms, *Communications of the ACM*, vol. 19.

Cano, P. and Torres, J. C., 2002. Representation of polyhedral objects using SP-Octrees. *Journal of WSCG*, 10. 95–101.

Cano, P., Torres, J. C. and Velasco, F., 2003. Progressive transmission of polyhedral solids using a hierarchical representation scheme. *Journal of WSCG*, 11. 95–101.

Everitt, 2000. Projective Texture Mapping. http://developer.nvidia.com.

Hoppe, H, 1996. Progressive meshes. *Proceedings of SIGGRAPH 96*, ACM Computing Series.

Hoppe, H., 1997. View-dependent refinement of progressive meshes. *Proceedings of SIGGRAPH 97*. 99–108.

Maciel, P. W. C. and Shirley, P., 1995. Visual navigation of large environments using textured clusters. In Hanrahan, P., Winget, J., (eds), *Symposium on Interactive 3D Graphics*, ACM SIGGRAPH.

Segal, M. and Akeley, K. The OpenGL Graphics System: A Specification. http://www.opengl.org.

Sillion, F., Drettakis, G. and Bodelet, B., 1997. Efficient impostor manipulation for real-time visualization of urban scenery. *Computer Graphics Forum* 16. 3.

Thibault, W. C. and Naylor, B., 1987. Set Operations on Polyhedra Using Binary Space Partitioning Trees. *ACM Computer Graphics*, 21(4).

Xia, J. and Varshney, A, 1996. Dynamic view-dependent simplification for polygonal models. Proceedings of *IEEE Visualization'96*, 327–334.