Progressive transmission of polyhedral solids using a hierarchical representation scheme

Pedro Cano Juan Carlos Torres Francisco Velasco

Dpto. Lenguajes y Sistemas Informáticos
E.T.S. Ingeniería Informática - Universidad de Granada
C/ Periodista Daniel Saucedo Aranda s/n
18071 - GRANADA (Spain)
{pcano, jctorres, fvelasco}@ugr.es

ABSTRACT

In the last years several 3D model compression methods for multiresolution applications have been presented, most of them using 3D meshes. Octrees are a natural multiresolution representation scheme, although it is approximate. Extensions of classical Octrees that represent polyhedral object exactly have been proposed. One of them are the SP-Octrees, that incorporates boundary information of the represented object in the internal nodes of the octal tree, and include new terminal node types that contain boundary information of the solid. This new scheme can represent polyhedral objects exactly with a smaller storage requirement, and can accelerate basic operations with the model. In this work we present the use of this new representation scheme for progressive transmission of polyhedral solids.

Keywords

Solid modelling, Hierarchical modelling, Octree, Multiresolution, Visualization, Progressive transmission.

1. INTRODUCTION

One of the consequences of the increase on processing and visualization capacity of the present systems is the increase of the complexity of the geometric models that we use. In addition, the development of the distributed systems allows us to transfer those models through networks, making it necessary to increase the speed of transmission and to reduce the storage cost.

Multiresolution models based on triangle meshes [Gar99][Tau99] can solve the model transmission problem, building different levels of detail and transferring in each case the desired level.

Journal of WSCG, Vol.11, No.1, ISSN 1213-6972
WSCG'2003, February 3-7, 2003, Plzen, Czech Republic.

Some of the schemes used to represent solids and volumes are based on the decomposition of the space, and use hierarchical structures to store the model.

An Octree is the representation of a model by means of an octal tree structure obtained by recursive divisions of the bounding box of the solid to codify [Mea82] [Fuj84] [Gar82]. The Octrees representation allows us to perform boolean operations and properties calculation in a simple way, but it is an approximated representation of the solid.

The Binary Space Partition trees (BSP) divide recursively the space using a plane in two separated half-spaces. Initially created to improve the hidden parts removal process [Fuc80], it has also been used to represent polyhedral objects exactly [Thi87]. This scheme offers an unambiguous, but not unique representation.

In previous works an extension of the classical Octrees was proposed by means of the inclusion of information of the boundary of the solid not only in the terminal nodes, but also in the internal nodes of the tree [Can02]. In this way, we avoid traversing the tree to the lowest level to accede to that information and we are able to accelerate basic operations on the model.

In this work we present an improvement of this scheme that allows us to represent any polyhedral object exactly. We also present its use for progressive transmission of the represented solid. In the following section the general statement of the scheme is presented. In section 3 we describe the visualization process of a model in the proposed scheme. In section 4 we describe the progressive transmission of solids using the proposed scheme.

2. SP-OCTREE

In classical Octrees the internal nodes are those that are not homogeneous with respect to the classification criteria. So, in these nodes the only information appearing is the references to their children.

To improve the classical Octrees, hierarchic schemes have been proposed that allow us to obtain an exact representation of polyhedral objects by means of the inclusion of new types of terminal nodes that contain part of the surface of the object, obtaining thus a more compact representation [Bru85] [Bru90] [Car85].

These extensions include information of the solid boundary in terminal nodes. So, the same boundary plane can appear in several neighbouring terminal nodes that share the boundary faces.

The idea of the proposed scheme, SP-Octrees [Can02] (Space Partition Octrees), is based on the inclusion of boundary information in internal nodes that partially defines the object represented in each node of that level. Thus, the information of the boundary faces appears in the upper levels of the tree and it is not necessary to repeat the information in neighbouring nodes that share a face.

When a node is completely out or in of the represented solid we classify it as WHITE or BLACK (figure 1, left and centre).

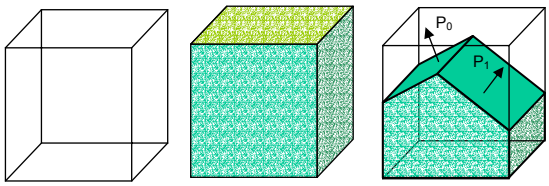

Figure 1. WHITE, BLACK and CONVEX nodes

When the intersection of the solid and the voxel is convex, we use a CONVEX node. Formally, a CONVEX node is the intersection of the half-spaces defined by the planes P_i included in it with its bounding box (figure 1, right).

When the intersection of the voxel and the solid has one concavity we use a CONCAVE node. Formally, a CONCAVE node is the difference of the bounding box of the node with the intersection of the complement of the half-spaces included in it (fig. 2).

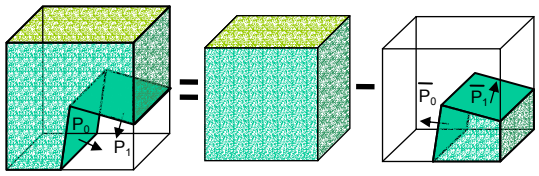

Figure 2. CONCAVE node

When concavities and convexities exist at the same voxel, we classify the node as GREY, dividing it in eight equals octants, but maintaining in the node the information of the planes that are in the convex hull of the part of the solid in the node. Thus, in the children we only need to represent the boundary planes that are not in that convex hull and which form the existing concavities.

Thus, the solid represented by a GREY node will be the union of the solid represented by each child, but restricted to the convex hull represented in the node.

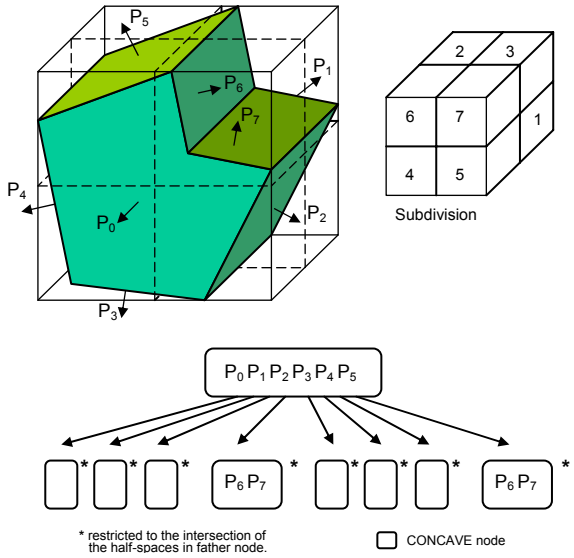

Figure 3. GREY node and its tree.

But with this criteria, if only one vertex of the represented solid exists in the voxel and concave e_c^i and convex e_x^i edges converge in it (figure 4), we always will have a GREY node although we descend in the tree. Therefore, in order to represent this polyhedral exactly, we needed to include a new type of terminal node.

In these cases we classified the node as VERTEX, storing information on what concave and convex edges appear between the existing planes in the node. So, we know the boolean operations to be made with the corresponding half spaces.

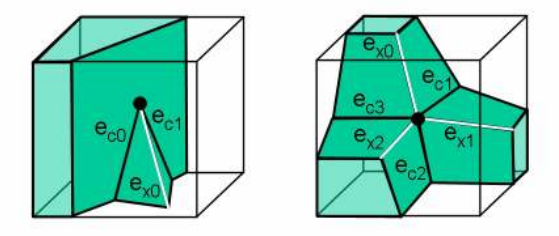

Figure 4. VERTEX nodes

Any node is bounded by the planes that are included in any of its ascendant nodes. The BLACK nodes can be clipped by the planes in their ancestors. The WHITE nodes have an empty set of planes.

For VERTEX nodes, we treat separately the planes that share concave edges from those that share convex ones. The process itself is similar to the one followed for CONVEX and CONCAVES nodes.

An important aspect of the proposed scheme is that the geometry of the represented object is not stored explicitly, which enables the representation obtained to be compact and reduces storage requirements.

3. ADAPTATIVE VISUALIZATION

One of the advantages of the classical Octrees is the inherent arrangement in the scheme, which facilitates the visualization process defining the order of visualization of the nodes. In our case we continue maintaining that arrangement.

In order to visualise an object represented by means of the proposed scheme, we traverse the tree level by level, drawing for each node the intersection of the planes that appear in it with its surrounding box and with the planes that appear in its 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.

To draw the object faces it is necessary to trim the planes in one node against those in its descendants. This can be done while drawing is carried out or we can modify the data structure to store also the geometry on the solid faces in each node of the tree. This can be easily obtained using a secondary B-Rep scheme to accelerate the process.

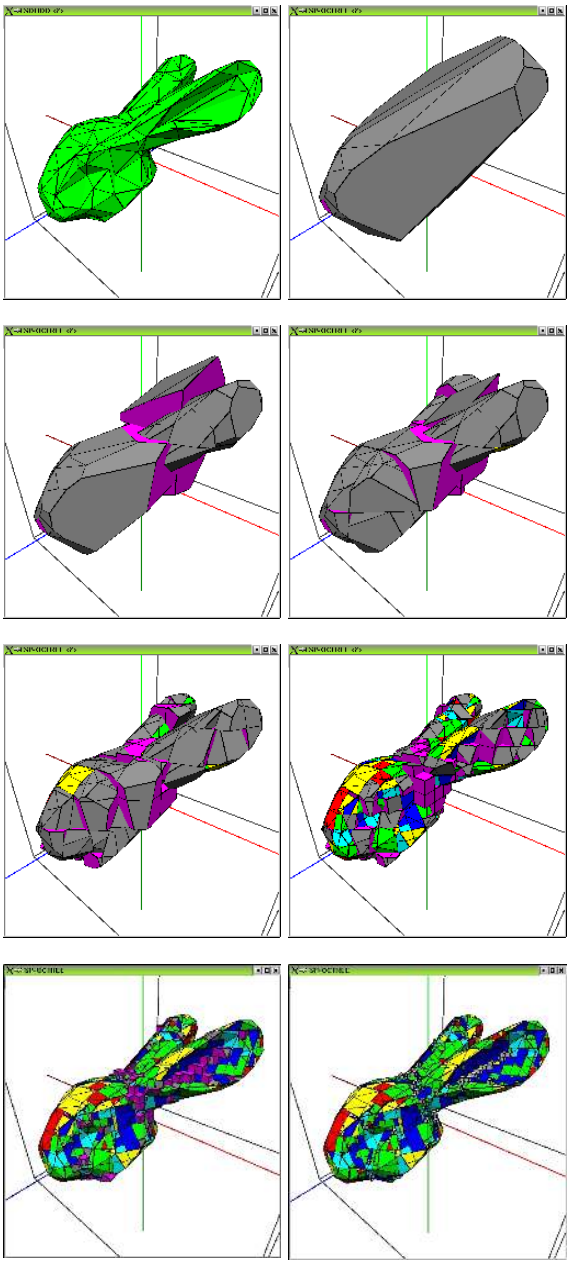

Image 1. Bunny head model: adaptative visualization by levels

This mechanism allows us to make an adaptative visualization according to the level of the tree that we represent.

As we maintain boundary information of the solid in the internal nodes of the tree (that it is part of the convex-hull of the represented solid in each node), we have an approximated representation of the modelled solid in each level of the tree.

The quality of the approximation improves as we are descending in the tree until arriving at the leaf nodes, where we have the exact representation of the solid.

In image 1 we have the visualization obtained for each level in the representation of a bunny head. The first image is the original solid (B-rep), and from the left to the right and from top to down we have the representation of each level. Faces color indicate the type of the node containing the face. Color grey shows planes in GREY nodes, colors green and blue show CONVEX and CONCAVE nodes respectively, and colors yellow and red show planes stored in upper levels of the tree but displayed in CONCAVE and CONVEX nodes respectively.

In this example we can see how the zones of the boundary that need a greater division in the model are those in which a greater concentration of concave edges exists.

4. PROGRESSIVE TRANSMISSION

Based in the previous visualization process, we can use this structure to make a progressive transmission of the tree level by level, so that the receiver of the model can visualize it and operate with it from the beginning of the transmission, without having to receive the complete model.

For each level, we transmit the nodes of the Octree that represent the model and the information of the boundary planes that appear in each node of that level (the equation of the planes).

The next algorithm shows the global process of transmission of the model:

```
Send (Level, BoundingBox)
Send (Planes in RootNode, NodeType)

If (RootNode=GREYNODE)
    Next_Level<-RootNode
    for each Level
    {
        Actual_Level = Next_Level
        Empty (Next_Level)

        while (Actual_Level ¡= EMPTY)
        {
            Node<-Get(Actual_Level)
            for each Child of Node
            {
                Send (Planes in Child, NodeType)
                if (Child=NODEGREY)
                    Next_Level<-Child
            }
        }
    }
```

Next_Level is a list that stores the GREY nodes sended in the actual level whose children must be send in the next level.

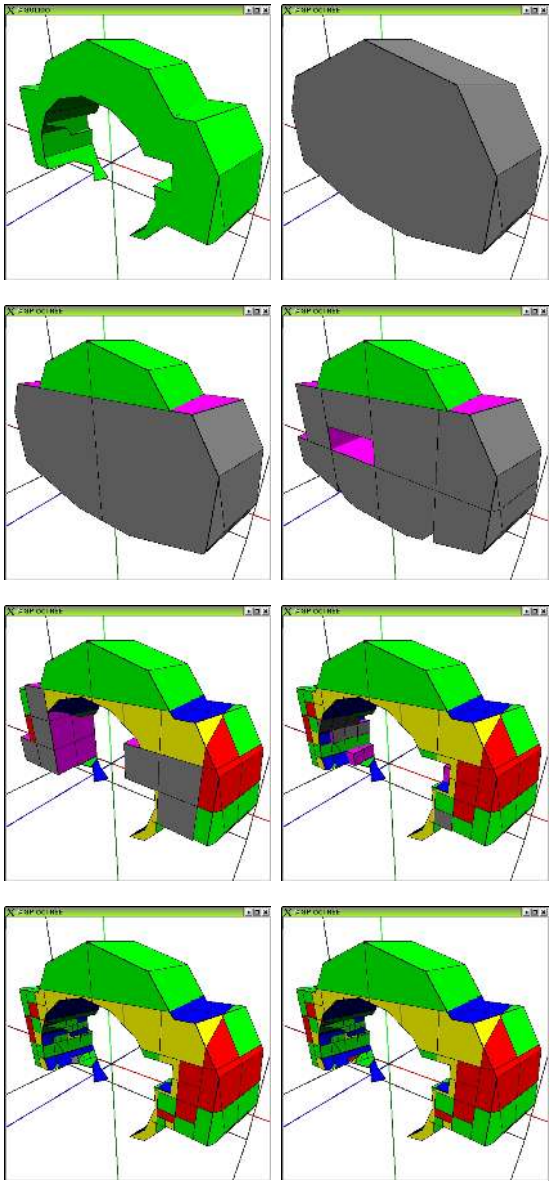

Image 2. Mechanical piece: visualization in each step of transmission

Get function obtain a pending GREY node in the actual level that has been just transmitted to send the information of its children.

Empty function initializes the list of pending GREY nodes to send in the next level.

The reception process is similar to this one, building the corresponding data structure for every received node.

In image 2 we can see a solid represented by a tree of 7 levels, for which at the first level we need only to transmit the information of 13 planes of the 43 which form the boundary of the solid. In level 1, the information of 2 new planes will be transmitted plus the nodes of the tree of that level.

The next table show the information transmitted at each level and the total size transmitted at this level for the mechanical piece.

Level	Transmitted size	Global size
0	724	724
1	104	828
2	572	900
3	636	1536
4	884	2420
5	1512	3932
6	596	4528

Table 1. Mechanical piece transmission (bytes)

In image 3 we can see a model of a bunny represented using a tree of 11 levels, we have an approximated representation of the modelled solid in each level of the tree.

The following table show the information transmitted at each level and the total size transmitted at this level for the bunny model.

Level	Transmitted size	Global size
0	37194	37194
1	201	37395
2	507	37902
3	2827	40729
4	14038	54767
5	28196	82963
6	25819	108782
7	5804	124586
8	11455	136041
9	11498	147539
10	10613	158152

Table 2. Bunny model transmission (bytes)

We must notice that at no moment we need to transmit geometric information of the polygons that form that boundary, but only the equation of the planes. In this sense, we need a decoding postprocess to reconstruct the geometry from the set of planes transmitted in each level.

As we can see in the examples, in the internal levels of the tree, where it only appears part of the information of the solid boundary, we have represented the solid with different levels of detail. This allows us to accelerate operations on the model (for example point's classification or intersection test with a ray).

Finally, another advantage is that this approach can be applied not only with triangular meshes, but also with any polyhedral mesh.

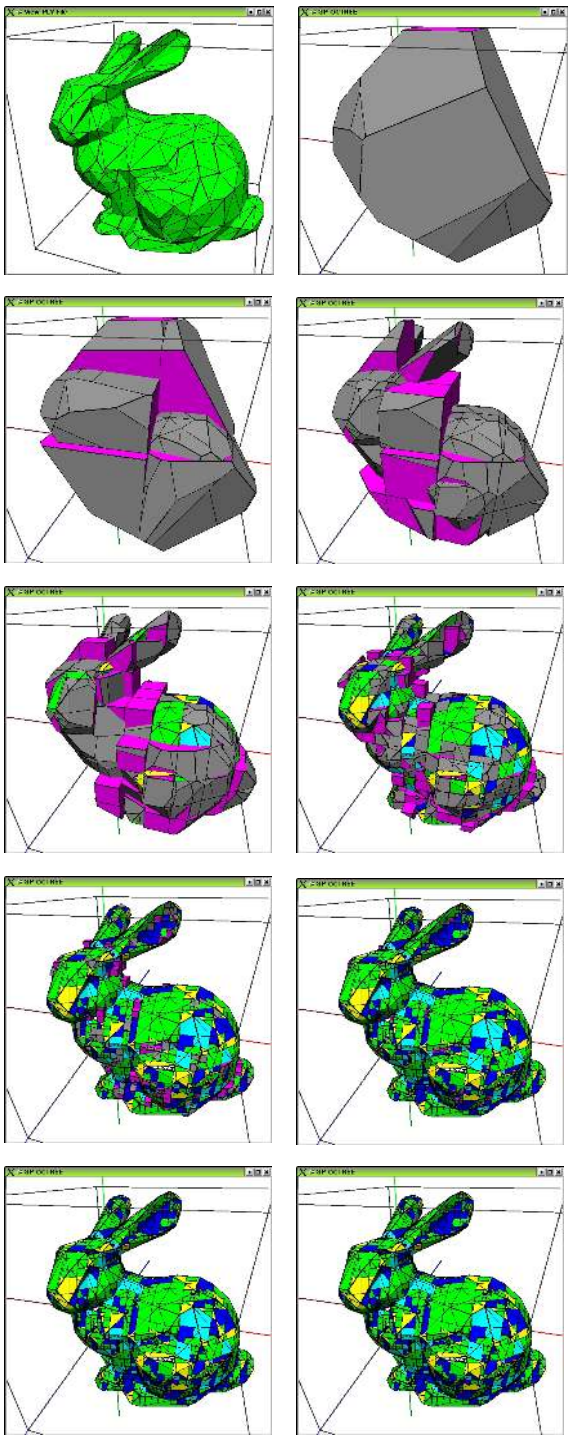

Image 3. Bunny model: adaptative visualization by levels

5. CONCLUSIONS

In this work, the use for progressive transmission of a new solid representation scheme has been presented. This scheme is based on an extension of the concept of classical Octree, introducing part of the boundary information of the represented object,

both in the terminal and in the internal nodes. The proposed method allows an exact representation of polyhedral objects.

Its use in progressive transmission allows the receiver of the model to visualize it and operate with it from the beginning of the transmission, without having to receive the complete model.

The number of levels that appear in the tree depends only on the concave edges that appear in the solid, whereas in other extensions of classical Octrees the number of nodes depends on the number of vertex and edges of it.

In addition, we continue maintaining the properties of arrangement of the classical Octrees, and, due to the own orientation of the planes inserted in each node, it is easy to make the interrogation and visualization of the model.

We are making a detailed comparative study with other representation schemes, in space, computation time and operations complexity. Also, we are studying the possibility of using the scheme for objects whose boundary is not plane.

Finally, we are studying the utility of the scheme as an indexing method to accelerate the calculations and the operations in B-Rep representation scheme.

6. ACKNOWLEDGMENTS

This work has been supported by the *"Ministerio de Ciencia y Tecnología" (Spain)* and *by FEDER* under contract TIC2001-2099-C03-02.

7. REFERENCES

[Bru85] Brunet, P.; Navazo, I.: *Geometric modelling using exact octree representation of polyhedral objects*. Eurographics'85, (1985).

[Bru90] Brunet, P.; Navazo, I.: *Solid Representation and Operation Using Extended Octrees.* ACM Transactions on Graphics, Vol. 9, nº 2. (1990).

[Can02] Cano, P.; Torres, J.C.: *Representation of Polyhedral Objects using SP-Octrees*. Journal of WSCG, vol. 10 (1) pp: 95-101. (2002).

[Car85] Carlbom, I.; Chakravarty, I.; Vanderschel, D.A.: *A hierarchical data structure for representing the spatial decomposition of 3D objects.* IEEE Computer Graphics & Applications 5,4, pp:24-31 (1985).

[Fuc80] Fuchs, H.; Kedem, Z.; Naylor, B.: *On Visible Surface Generation by a Priori Tree Structures.* ACM *C*omputer Graphics, 14(3), (1980).

[Fuj84] Fujimura, K.; Yamaguchi, K.; Kunii, T.: *Octree-related data structures and algorithms*. IEEE Computer Graphics and Applications, pp: 53-59, (1984).

[Gar99] Garland, M.: *Multiresolution Modelling: Survey & Future Opportunities*. EUROGRAPHICS'99 State of the Art Report. EG, (1999).

[Gar82] Gargantini, I.: *Linear octrees for fast processing of three-dimensional objects.* Computer Graphics and Image Processing, 20, (1982).

[Mea82] Meagher, D.: *Geometric modelling using octree encoding.* Computer Graphics and Image Processing, 19(2):129-147, (1982).

[Tau99] Taubin, G.: *3D Geometric Compression and Progressive Transmission.* EUROGRAPHICS'99 State of the Art Report. EG, (1999).

[Thi87] Thibault, W.C.; Naylor, B.: *Set Operations on Polyhedra Using Binary Space Partitioning Trees.* ACM Computer Graphics, 21(4), pp: 153-162, (1987).