Octree Detection of Closed Compartments

P. Brunet, I. Navazo, A. Vinacua
Universitat Politècnica de Catalunya
Barcelona, Spain

Abstract

The present paper addresses the problem of detecting closed compartments produced by a set of planar faces in the space. The topology of the set is general, and edges in the final piecewise planar surface can belong to one, two or more faces; boundary representations for non-manifold solids are particular cases. An octree structure (dubbed *compartment Octree*) that defines a 3D graph through the volume defined by the set of faces is proposed, and it is shown that a seed propagation algorithm on the graph can be used to detect the existing closed compartments. The algorithm can either compute the total number of compartments or detect if the set of faces define a closed solid volume, the outside part being considered as a separate compartment.

1. Introduction

Several solid representation schemes have been developed for Geometric Solid Modelling [ReV-83]. Choosing one or other is closely linked with the application at hand, its domain, features and interface with the user.

In some applications it is necessary to store volume properties or volume information of the objects that we want to model. Some examples are medical imaging [Mea-85], geological information [Bru-89], computing integral properties [LeR-82] or detecting assemblies in solids composed of different materials such a layers in silicon devices [RoC-89]. When this kind of volume information is needed several solid models are more suitable than others. In particular, octree representations are shown to be specially useful for this class of applications.

An octree is a tree that codes the adaptive recursive subdivision of a finite cubic universe. In this structure, each node is terminal or has eight descendants. The volume and, if necessary, the external geometry of the solid are stored in the allowed terminal nodes. The root of the tree represents the universe, a cube with an edge of size 2^n. This cube is divided into eight identical cubes, called octants, with an edge length of 2^{n-1}. Each octant is represented by one of the eight descendants of the root. If an octant cannot be considered terminal (Grey node), it is divided into another eight identical cubes which are represented as descendants of the octant in question. The previous process is repeated recursively until octants can be considered terminals. The class of terminal nodes is a function of the application [Bru-89][BJN-90]. For example, in classical octrees terminal nodes are related to cubes that are totally inside the solid (Black nodes) or totally outside (White nodes).

This paper discusses an octree structure that can be used to solve some questions related to the volume of a solid. We use the word object in a very wide way, so we are thinking about dangling or not dangling and manifold or non-manifold 3D objects [Req-77]. The only information we need is a set of planar faces (externals or internal with respect to the volume) that constitute the 3D object that we want to analyze (Fig 1).Subsets of these faces can

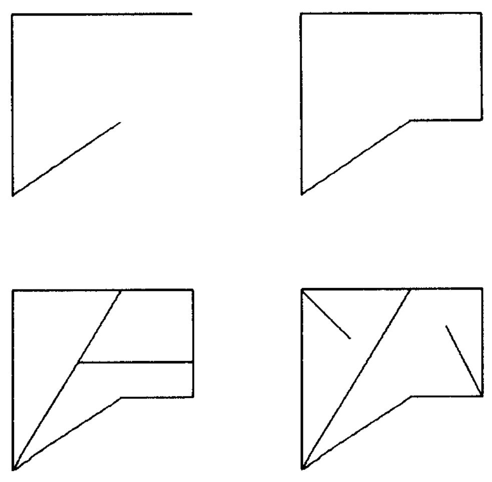

Fig 1: Possible input data (2D)

close isolated volumes. We name each of these volumes a *compartment*. More precisely, if S stands for the given set of faces, then each of the connected components of $E^3 - S$ (where E^3 stands for the three dimensional euclidean space) is a compartment.

The proposed octree structure defines a 3D graph through the volume and can answer questions as how many compartments does the object have, detect if the initial set of faces bound a closed volume, obtain the connectivity graph among

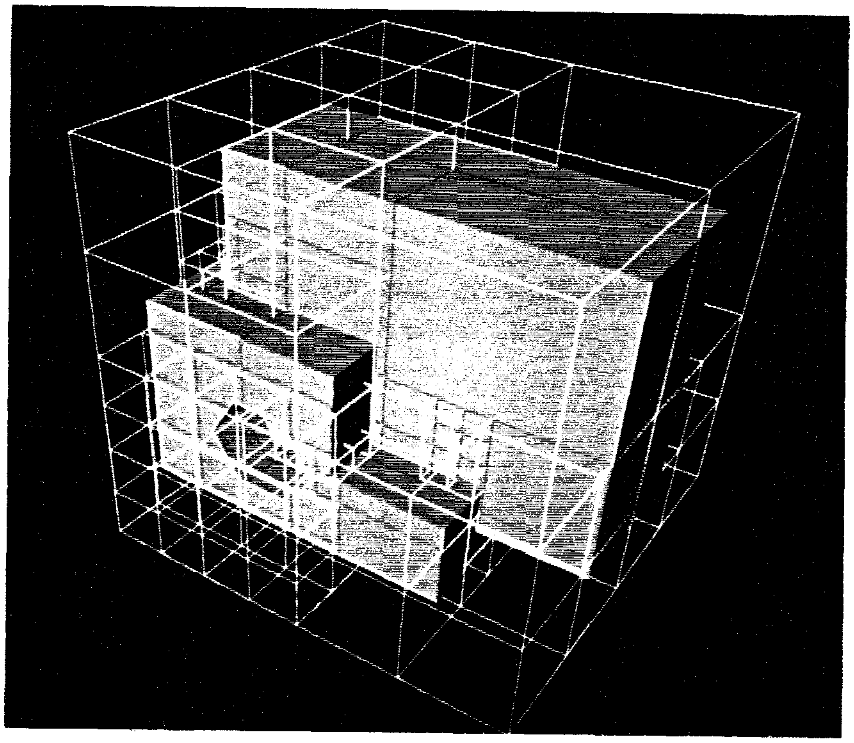

Fig 2: Mesh vertices and the 3D graph produced by the octree representation

the compartments and calculate the volume of a compartment.

In order to build the octree, an algorithm like that presented in [BrN-85] can be used. The only information needed is a set of faces. This information can be extracted from the BR, CSG or SCR [RoO-89] model of the solid. When the octree has been built, a seed propagation algorithm can be used to answer the stated questions.

Some previous works related to this paper are presented in [Sam-89a]. From a colored classical octree (Black nodes have a property associated to them, like color) algorithms for connecting nodes with the same property are presented. We can obtain this kind of octree from a set of faces but we have an approximate representation of the volume. [Bru-89] presents a special type of octrees, denoted mixed octrees, that can handle solids that have their volume subdivided by planes.

The paper is organized as follows. Section 2 introduces several assumptions that the octree representations must satisfy and presents algorithms that solve the stated problems. Then it gives a definition of an octree scheme satisfying those properties, and others that make computation with it easier. Section 3 contains the necessary proofs. We discuss some extensions in the conclusions

2. An Approach to the Problem

Let us suppose that an octree is generated, based on the geometry of the set of faces (Fig 2). Vertices of the cubes corresponding to terminal nodes in the octree will be noted as mesh vertices. Neighbor terminal nodes are those having geometric contact between their corresponding cubes. Neighbor terminal nodes can be either vertex-connected, edge-connected or face-connected. Two vertex-connected terminal nodes have associated cubes that share one of their vertices. In edge-connected terminal nodes, there is no vertex-connection but at least one of the vertices of one of the cubes is on one of the edges of the other cube. Finally, face-connected nodes have associated cubes with coincident faces, but they are neither vertex-connected nor face-connected (Fig 3). Specific algorithms that give the face, edge or vertex-connected nodes for a given node of the tree have been proposed, [Sam-89b].

The set of planar faces induces a local compartimentation in every cubic volume corresponding to a terminal node of the octree. The connected regions of the space within every terminal cube will be noted as *local compartments*. Note that two points in different local compartments may belong to the same compartment, as it is possible that they are connected through a path external to the node, (Fig 4). We shall call *perimeter* of a local compartment to the intersection of the local compartment with the boundary of the node. This perimeter is formed by several polygons each of which results of intersecting the local compartment with one of the faces of the node. We will call each of these polygons a *face* of the perimeter. A face of the perimeter of a local compartment is called *significant* if it intersects more than one edge of the node. A local compartment that does not contain any of the eight vertices of its associated node is called *volatile*. Let us make now the following assumptions,

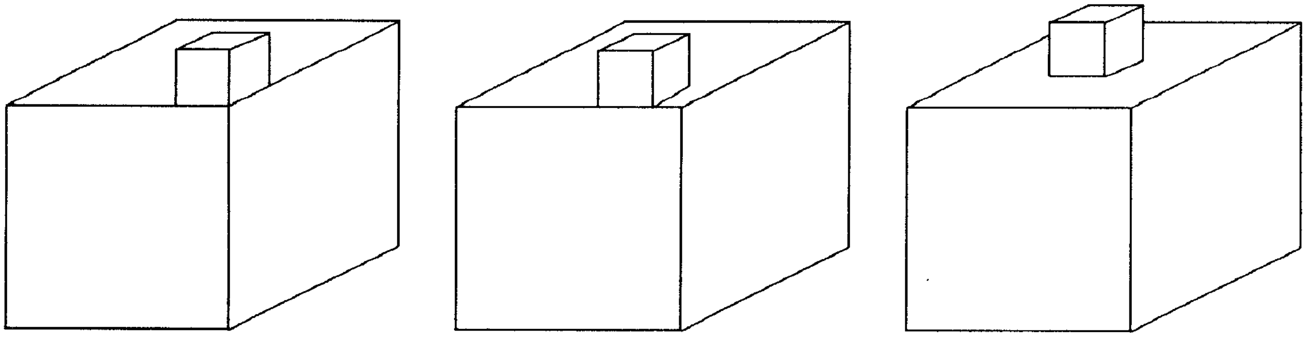

Fig 3: Neighbor node connectivities. Examples of vertex-connected, edge-connected and face-connected nodes.

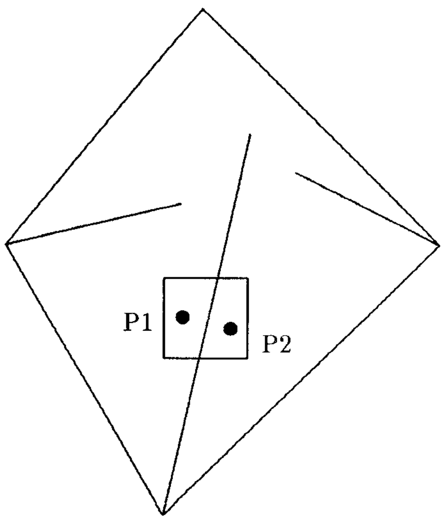

Fig 4: Two points P1 and P2 belonging to different local compartments may belong to the same compartment

A1- The octree representation is such that for every compartment, there exist at least one mesh vertex belonging to it.

A2- For every terminal node, the octree representation keeps a classification of its mesh vertices in different sets, such that vertices belong to the same set if and only if they are in the same local compartment.

A3- For any two contiguous compartments (that is for any pair of compartments that have a portion of a face in common) there is a terminal node in the octree having at least one vertex in each of the two compartments.

A4- Removing all the volatile local compartments does not break any of the compartments into disjoint compartments, nor does it change the number of compartments.

This last assumption guarantees that all the relevant information is actually captured by non-volatile compartments, and is therefore crucial. It is not easy to verify directly in this form, so we will actually check that the perimeter of the local volatile compartments that our scheme generates are connex, and that significant faces of volatile local compartments cannot be adjacent. In fact, several local compartments sharing significant faces may define a duct that establishes certain connection chanel. However, one can never obtain such a duct by adjoining local compartments by their non-significant faces.

Note also that, since no compartment can have zero neighbors, A1 is implied by A3. If these assumptions are satisfied, the following problems can be solved by means of the octree representation of the set of faces.

2.1. Several Problems

Problem 1: Obtain the number of compartments associated to the initial set of faces. (All faces are inside the root cube —universe— of the octree; the outside part of the object is also considered as a compartment which always contains the 8 vertices of the root cube).

Because of the assumptions A1, A2,and A4, the algorithm in Fig 5 gives a solution of this problem, based on a seed process through the octree nodes.

In the last procedure, the seed first propagates from a mesh vertex to all vertices in the same set in the node being visited. Then, not yet visited neighbor nodes n' to the present node n are considered, using face, edge and vertex-connectivities. However, it must be observed that finding a vertex w of n' in the same local compartment of m involves geometric computations in face and edge connected nodes, as the geometry of the set of faces restricted to the common face or edge must be taken into account (Fig 6).

In this figure, mesh vertex $V1$ propagates the seed to vertex $V3$ in Fig 6–a, or to vertex $V4$ in Fig 6–b.

We will show in section 3 that the seed propagation algorithm 2.1 in every closed compartment is complete, in the sense that given any two mesh vertices of the same compartment, a seed process starting at the first of them always reaches the second.

Problem 2: Detect if the initial set of faces bound a closed —solid— volume.

After algorithm 2.1, problem 2 is solved by the test $nc = 1$, as shown by the following algorithm,

```
nc := 0
Consider all mesh vertices as non-marked
While there is at least one non-marked mesh vertex do
        search for the first non-marked mesh vertex
        mark it
        pick terminal node that has this point as a vertex
        propagate (mesh_vertex, node)
        nc := nc + 1
end do
procedure propagate (m, n) is
             for each vertex v of n in the same set as m do
                 mark v
             end do
             for each not yet visited connected neighbor n' do
                 if find vertex w of n' in the same
                   local compartment of m then
                                                  propagate (w, n')
                 end if
             end do
end procedure
```

Fig 5: Algorithm 2.1

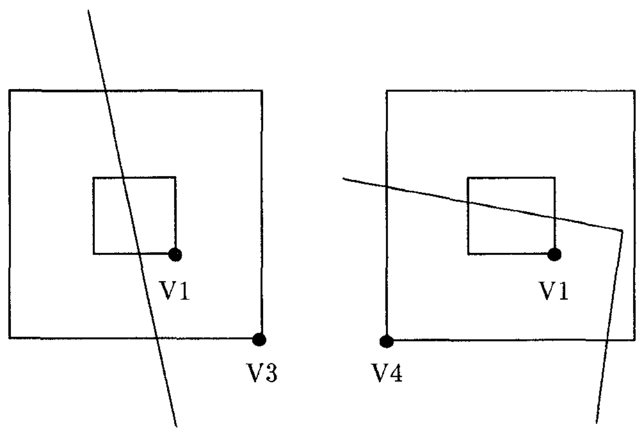

Fig 6: Propagation in face-connected nodes depends on the geometry of the compartment walls. V1 propagates to V3 in figure 6–a, and to V4 in figure 6–b.

case of nc
- 1: There is no closed volume defined by the set of faces
- 2: The set of faces bound a closed volume
- ≥ 3: The set of faces bound a closed volume with $nc - 1$ compartments

end case

Problem 3: Obtain the connectivity graph among the compartments defined by the set of faces. Vertices of the connectivity graph are the compartments obtained from the solution of problem 1; an arc exists between two compartments if they are contiguous (separated by a planar face of the initial set).

Because of assumption A3, this problem can be solved by a tree traversal after the seed process described in the algorithm 2.1, as for every two contiguous compartments there exists at least a terminal node of the octree with sets of mesh vertices corresponding to these two compartments.

Then,

```
for every terminal node in the octree do
    if not all vertex sets belong to the
         same compartment then
       create a graph arc for every two contiguous
         vertices of the node
       belonging to sets with different marks if it
       does not yet exist
    end if
end do
```

2.2. The Implementation Proposed

As has already been stated, the propagation of the seed information through face and edge connectivities —which is essential for algorithm 2.1— must be based on the local geometry of the set of faces. As a consequence, terminal

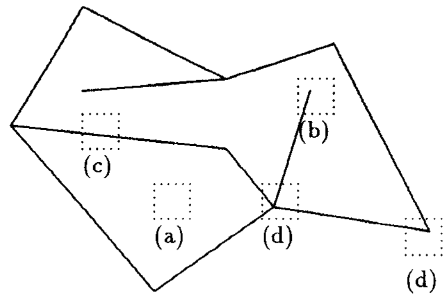

Fig 8: Terminal octree nodes can be homogeneous (a, b), face nodes (c) and junction nodes (d)

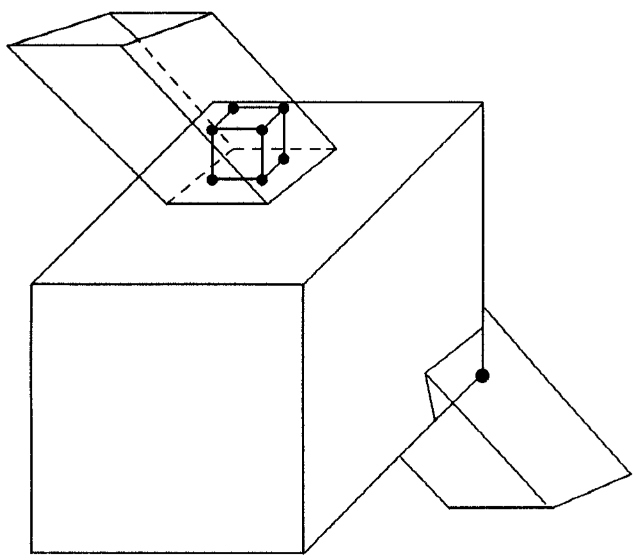

Fig 7: Propagation between face connected nodes is based on the local geometry. It can involve complex operations that produce incoherences and lack of robustness. In the figure, vertices of the small node propagate only to the bottom vertex of the largest node

nodes must keep geometric information of the planar faces. In complex configurations, such as the one shown in Fig 7, propagation can become cumbersome. Robustness must also be guaranteed in cases where independent decisions on different compartments may become incoherent.

In order to avoid these problems, we propose an octree definition with the following node types (fig 8), which we will call *Compartment Octrees*:

Homogeneous node: It corresponds to a cube that, because of the clipping process in the recursive division of the space, it contains either no face of the boundary representation, or only one face together some of its edges. (Either no part of the surface is inside or it contains a part of it; but in this last case, the surface does not divide the inside part of the cube, which remains connected)

Face node: It corresponds to a cube containing one face of the boundary representation, but no edges.

Junction node: It corresponds to a cube containing two or more faces, all converging to a single vertex or an edge inside the cube.

Grey node: Every cube in the subdivision process that cannot be assimilated to any of the previous types, and therefore must be subdivided.

By defining the *valence* of a node as the number of local compartments in the part of the solid inside the cube, we can say that homogeneous nodes have *valence* = 1, while the valence of face nodes is 2 and junction nodes have valence greater or equal than two.

This definition obviously fulfills assumption A2. Assumptions A3 and A4 will be verified in section 3. As observed above, A1 is implied by A3. Moreover, as will be shown in the following section, if two mesh vertices $V1$ and $V2$ of neighbor face, edge or vertex-connected nodes belong to the same compartment, then $V2$ can always be reached from $V1$ using only vertex connectivities from node to node, through a traversal of the 3D octree graph. Then, as a consequence, problem 1 can also be solved by the algorithm in Fig 9.

The main difference between algorithms 2.1 and 2.2 is that algorithm 2.2 does not use geometric operations in the propagation from node to node; this process is performed simply through common mesh vertices. For solving problems 1, 2 and 3, it is sufficient that terminal nodes keep the classification of mesh vertices in different sets, as required by assumption A2, without keeping further geometric information of the planar faces. Point-compartment classification geometric tests must be done only during the octree construction, and the relevant information for the algorithm is kept in the classification of mesh vertices into sets in terminal nodes. Consequently, robustness of the seed algorithm is guaranteed.

On the other hand, if the codification of terminal nodes associates to every set of vertices the value of the volume of the corresponding local compartment, then the following problem can be solved as a corollary of problem 3,

Problem 4: Compute the volume of each of the compartments defined by the set of faces.

This problem can only be solved approximately, since volatile local compartments will not be accounted for. However, this error can be arbitrarily reduced by modifying the definition of junction nodes for this problem, replacing it by:

```
nc := 0
Consider all mesh vertices as non-marked
while there is at least one non-marked mesh vertex do
        search for the first non-marked mesh vertex
        mark it
        pick terminal node that has this point as a vertex
        propagate (mesh_vertex, node)
        nc := nc + 1
end do

procedure propagate (m, n) is
            for each vertex v of n in the same set as m do
                mark v
            end do
            for each marked vertex w of node n do
                for each not yet visited node n' sharing mesh vertex w do
                    propagate (w, n')
                end do
            end do
end procedure
```

Fig 9: Algorithm 2.2

Junction node: It corresponds to a cube containing two or more faces, all converging to a single vertex or an edge inside the cube. *Moreover, either each local compartment is non-volatile, or the side of the node's cube has a predefined size* ε

This difficulty may not be circumvented by requiring that each junction node have at least one vertex in each of its local compartments, as there may be points such that any neighborhood of them intersects more than eight compartments, forcing infinite subdivision or else producing an approximation as above.

Note also that this solution of problem 3 computes in this case not only the volume of the existing compartments, but their connectivity graph. As a consequence, modifications of the volume due to changes in "porosity" in faces that separate contiguous compartments are straightforward.

The computation of more complex volume properties requires that terminal nodes keep geometric information of the local set of faces.

3. Validity of the Proposed Scheme

In the previous section we introduced several problems and algorithms to solve them using an octree satisfying assumptions A1, A2, A3 and A4. Along the way, we have made several claims that we will now prove. Essentially, these are: that a Compartment Octree satisfies A3 and A4, that the seed propagation is complete, and that dealing with vertex-connectivities is enough for Compartment Octrees. Each of the following subsections deals in turn with each of these problems.

3.1. Compartment Octrees Satisfy A3 and A4

Let us first prove A3. That is, we want to see that for every pair of neighboring compartments there is a node in the octree that has at least one vertex on each of the two compartments.

Proof: If the two compartments are separated by the polygon P, then look at a terminal node which contains a portion of this polygon. If it is a face node, then it has at least one vertex on each of the two compartments as required. If it is a junction node, then either it fulfills the requirement or there is at least one more polygon going out of the node through the same facets. This can happen in several ways (see Fig 10). In all but one, (depicted in Fig 10–a) there is a facet in the junction node that is crossed by both polygons but has no point common to both of them. In this cases, since the polygons cannot converge to a vertex or edge on the neighboring node, that node at this level must be grey, and is subdivided into face nodes, one of which contains a portion of P and thus fulfills the requirement imposed by A3.

On the other hand it is clear that not all of P may be contained in junction nodes with a configuration like that in Fig 10–a, so the proof is complete.

Observe that by this argument, if we chop off a compartment all the portions consisting of local compartments in junction nodes that are not represented by any of the vertices of the node, then we obtain a smaller volume, but it is still connected.

As for assumption A4, its proof has been mentioned in passing in the previous argument. For a junction node (the only kind that may have volatile local compartments) to have a significant face on a volatile local compartment, there must be more than one polygon going out of the node through the facet containing such a face, and since they cannot converge again in the neighboring node to a vertex or an edge, that node cannot be another junction node.

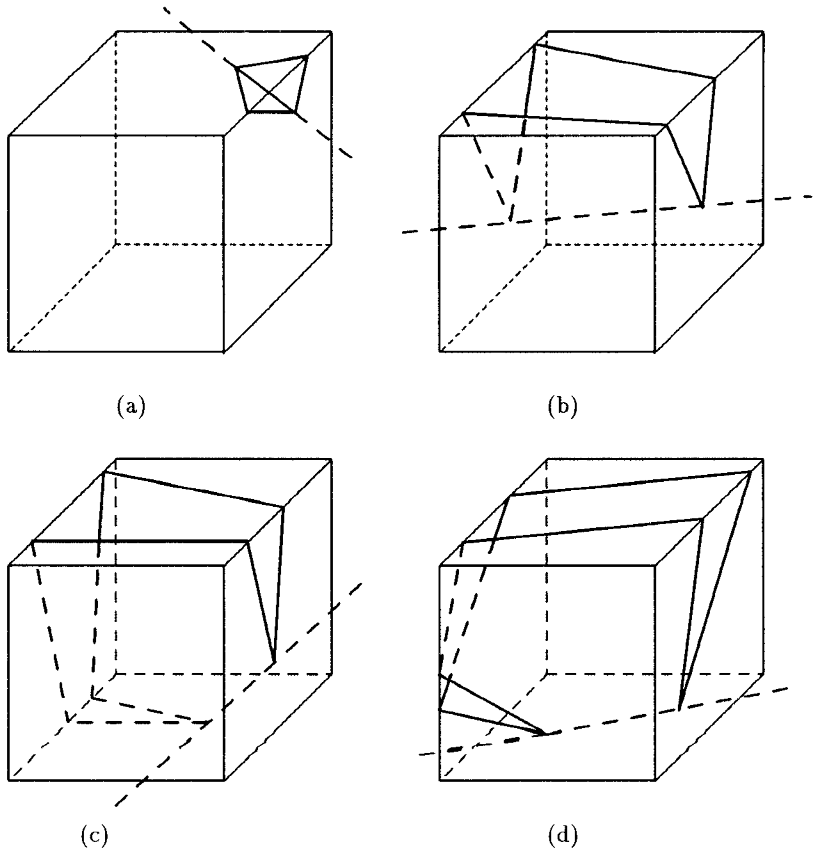

Fig 10: Examples of junction nodes having no "Grey faces" (a) and having them (b, c, d)

3.2. Seed Propagation is Complete

Recall that by this we mean that mesh points that belong to the same compartment ought to end up having the same mark.

To prove it, we will need to introduce some notaion and definitions. First, note that since the relation "is connected to" between neighboring nodes is reflexive, the seed propagation algorithm (2.1) is also reflexive. That is, if from V the seed can propagate to W, then so can it go from W to V. Hence, it is easy to see that the seed propagation algorithm defines an equivalence relation '$\sim$' among the mesh points. Namely

$$V \sim W \iff \text{the seed propagates from } V \text{ to } W.$$

Consider now a closed connected compartment C. By assumption A4, if we remove all the volatile local compartments contained in C, we will obtain a closed connected subset o C. That is, by removing those pieces we will not break C apart. Let C' denote this simplified compartment. Obviously a mesh point belongs to C if and only if it belongs to C', as the parts removed did not have any mesh points of their own. Now take the set of all mesh points in C' and break it up in equivalence classes modulo '$\sim$'. We would like to show that there is only one such class. To prove it, we will define the *span* of a class α. Given a mesh point m, let N_m denote the set of all the terminal nodes in the octree that have m as a vertex. Also let n_m denote the local compartment contained in node n that contains the vertex m. Then we define the span σ of a class α as

$$\sigma(\alpha) = \bigcup_{m \in \alpha} \bigcup_{n \in N_m} n_m \subset C'.$$

Supose now that there are different equivalence classes. Then there must be some pair (α, β) such that $\sigma(\alpha)$ and $\sigma(\beta)$ are contiguous (share a portion of a facet through which they are connected), or otherwise we would have broken C' into disjoint components. But if $\sigma(\alpha)$ and $\sigma(\beta)$ are contiguous in the sense just stated, then the seed propagation algorithm 2. would have propagated the seed from a vertex in α to a vertex in β and therefore $\alpha = \beta$. Therefore there cannot exist more than one equivalence class in a compartment, or in other words, the algorithm is complete, which finishes the proof.

3.3. For Compartment Octrees, Vertex Connectivities Suffice

In the preceding section, it has been pointed out that correctly propagating the seed to neighboring vertices that are edge or face-connected requires using geometric information that has to be carried along in the nodes and carefully evaluated in order to produce the same result no matter the order in which the seed propagation is carried out. It has also been stated that with Compartment Octrees, not only assumptions A1 through A4 hold, but also the fact that for these special octrees vertex-connections also give a complete seed-propagation algorithm, attaining a better robustness and no longer requiring the storage of the geometric information, although some applications may require it nonetheless for purposes other than the compartment detection. We now turn to verifying this statement.

To show this, we will actually show that, for these octrees, following vertex connectivities gives the same result as following also edge and face connectivities, and then the previous argument proves completeness in this case too.

In order to avoid confusion, in the following "faces" or "polygons" will stand for the data faces that define the compartments, whereas "facets" will denote the six faces of a terminal node. We will also use "restricted polygons" or "restricted faces" to refer to the intersections of the faces with a plane or a line.

We will need for our argument the notion of induced quadtree on a plane. Let ρ be a plane parallel to a coordinate plane and passing through the root cube of the octree. Let ρ^+ and ρ^- denote the two planes parallel to ρ lying on both sides of it at a distance ε, where ε is any arbitrarily small positive number. In fact we need it to be smaller than the closest mesh point not lying on ρ. The octree restricted to either of these planes is a quadtree on the plane. Consider the quadtree on ρ obtained by shifting back onto ρ the two quadtrees thus obtained on ρ^+ and ρ^- and merging them. This is what we will call the induced quadtree on ρ. This construction is necessary because we will want to focus on cases where ρ is the support plane of a facet, and thus the quadtree on ρ^+ and on ρ^- will often differ. Our definition above amounts to locally choosing the finer of the two in each region of ρ.

The first important fact about these induced quadtree is that it is a "Compartment Quadtree" for the plane ρ and the traces of all the given polygons on ρ, although it probably has been overdivided (that is: nodes that were perfectly valid terminal nodes for such a quadtree have been subdivided nonetheless).

This fact is not trivial, but follows from the fact that a facet of a homogeneous node as defined in section 2 is either a homogeneous node of a quadtree associated to the restricted faces, or it is a face node (Fig 11). The facets of a face node are in turn homogeneous or face nodes of such a quadtree (Fig 12) and facets of a junction node are either homogeneous, face, junction or grey nodes of such a quadtree. However, when they are grey (Fig 13) there are more than one polygon going out across that facet, and they do not have any points in common on the facet. Therefore these polygons cannot converge to a common vertex or edge on the neighboring node of the octree, and thus the neighbor in that direction on the same subdivision level of the octree must be a grey node, and its subdivision will induce on the facet's quadtree the corresponding subdivision yielding smaller face nodes. This is the role of merging the quadtrees induced on slightly shifted planes, so that locally we always end up with the finer of the two quadtrees.

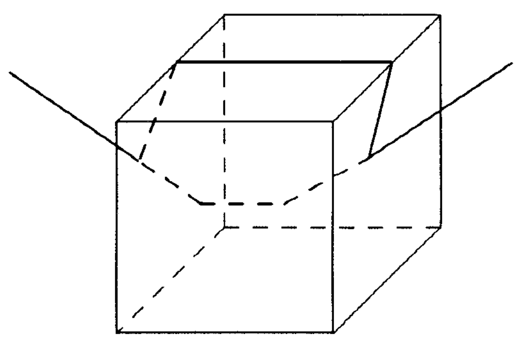

Fig 11: Facets of a homogeneous node are homogeneous or face nodes of the induced quadtrees

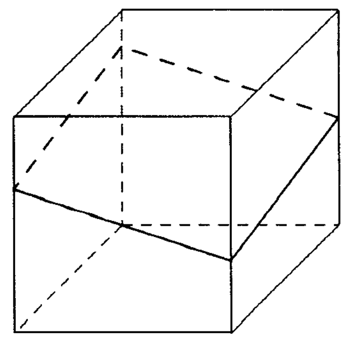

Fig 12: Facets of a face node are either homogeneous or face nodes are either homogeneous or face nodes of the induced quadtree

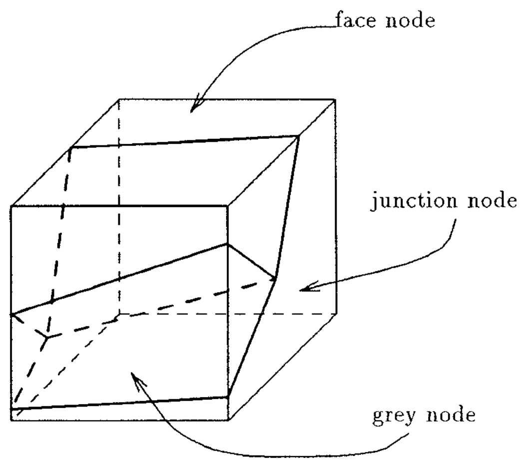

Fig 13: Facets of junction nodes include all possibilities

A similar argument shows that the restriction to an edge is a "Compartment bintree" corresponding to the traces on the edge of the given polygons, probably over-subdivided. All nodes here are either homogeneous (and both ends are in the same compartment) or face nodes (and both ends lie in neighboring compartments).

Now of course in the case of these bintrees the only possible connectivity is the vertex connectivity, and it is clear from the previous enumeration of possible cases that the vertex-based seed propagation is complete in this case. Also these bintrees will fulfill assumptions A1 through A3, and in particular will contain at least one mesh point in each compartment that intersects the edge.

Now go up in the number of dimensions. Let us look at the induced quadtree on the plane that supports certain facet of a terminal node in the octree. Now edge connectivities are possible, but when they happen, the edge on which they appear is an edge of the original octree, on which the germ propagation along the restricted bintree is complete, as we have just shown. So the seed will propagate properly across this kind of connectivity, and the process, on this quadtree, will be complete.

Now given a node of the octree, with a vertex in a given compartment, the germ will properly propagate to all mesh points on the edges converging onto that point, and on all the mesh points on the facets containing that point, and thus to all nodes having edge or face contact on either of these. If the compartment contains part of another facet not containing this vertex (nor any other of the terminal node under consideration), then the edge joining that face with one of the three facets containing the vertex must contain a mesh point inside the compartment, to which the seed arrives because the propagation on the facet is complete, and from which the seed propagates to all and any other mesh points on this last facet (see Fig 14).

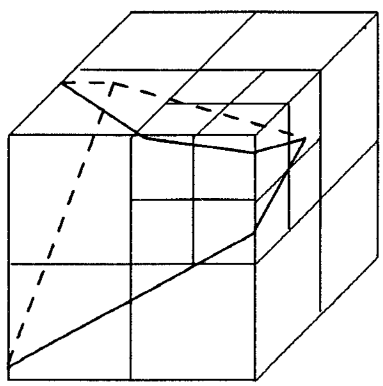

Fig 14: For junction nodes, the subdivision on the neighboring nodes produce the necessary mesh points to properly propagate the information

Thus, if two nodes of a compartment octree are face or edge-connected, the propagation of the seed through vertex connections will eventually reach the connecting vertex on the facet or edge, and will thus propagate correctly into the neighboring node. This completes the proof.

4. Conclusions

In the present paper, an octree structure for detecting closed compartments in sets of faces in the space has been proposed. Compartment octrees derive from extended octrees, and adapt the space subdivision to the geometry of the given set of faces. The octree generation is based on clipping and localization of geometric information. A subsequent seed propagation algorithm which involves no geometric computations can be used to obtain the number of closed compartments, to detect if the set of faces enclose a closed volume, to obtain the connectivity graph among compartments, or to compute the volume of every compartment if information from local compartments is kept in terminal nodes during the generation process.

The robustness of the algorithm is ensured by the fact that no geometric operation is performed after the octree structure has been obtained. The proposed structure can also be used for several related problems. We can mention, for instance, the detection of dangling faces in potential boundary representations. Dangling faces lead, during the octree generation, either to homogeneous terminal nodes with one face, or to face or junction nodes having two identically colored sets after the seed propagation.

In order to extend the use of the proposed octree structure to more complex problems such as the computation of the precise volume and other volumetric properties, it would be necessary to keep explicit geometric information in terminal nodes. This information is also necessary when an incremental evaluation of compartments is required.

5. Acknowledgements

The authors would like to thank Jaroslaw Rossignac for the initial ideas and discussions that lead to the developments in this paper.

6. References

[BJN-90] Brunet P., Juan R., Navazo I., Octree Representations in Solid Modeling, in *Progress in Computer Graphics*, G. Zobrist Ed., Abblex Publ. Co., in press

[BrN-85] Brunet P., Navazo I., Geometric Modelling Using Exact Octree Representation of Polyhedral Objects, *Proc. of Eurographics'85*, North Holland, Nice 1985, pp 159–169

[Bru-89] Brunet P., 3-D Structures for the Encoding of Geometry and Internal Properties, *NATO Advanced Research Workshop on Three Dimensional Modeling with Geoscientific Information Systems*, Santa Barbara, California, December 1989

[LeR-82] Lee Y.T.L., Requicha A.A.G., Algorithms for Computing the Volume and Other Integral Properties of Solids, *Comunications of the ACM*, vol. **25**,9 pp 642–650, September 1982

[Mea-85] Meagher D., The Application of Octree Techniques to 3-D Medical Imaging, *Proc. of the Seventh Annual Conference of the IEEE Engineering in Medicine and Biology Society*, Chicago, Illinois 1985, pp 612–615

[ReV-83] Requicha A.A.G., Voelcker H.B., Solid modelling: A historical summary and contemporary assessment, *IEEE Computer Graphics and Applications*, vol. **2**,2, pp 9–24, October 1983

[RoC-89] Rossignac J., O'Connor M.A., SGC: A dimension independent model for pointsets with internal structures and incomplete boundaries, *IBM Research Report, RC 14340*, January 1989

[Sam-89a] Samet H., *Applications of Spatial data Structures*, Addison–Wesley Publishing Company, 1989

[Sam-89b] Samet H., Neighbor Finding in Images Represented by Octrees, *Computer Vision, Graphics and Image Processing*, vol. **46**, pp 367–386, 1989