Non-Manifold Boundary Representation for Solid Modelling

Stefka Gueorguieva David Marcheix

Laboratoire Bordelais de Recherche en Informatique (LaBRI) - Univ. Bordeaux I
351, Cours de la Libération, 33405 Talence, FRANCE

Abstract

The focus of current researches in geometric modelling is the extension of the representational power in order to encompass the non-manifold objects. In the present paper a non-manifold boundary representation is proposed. New types of topological elements are introduced to extend the dimension of the non-manifold condition at a face. This structure is based on the theory of simplicial complexes. The technique used for construction and manipulation of objects ensures the model validity at each intermediate stage of the design. An undergrounding set of basic operators is developed that according to empirical results is a complete and sufficient set for manipulation of objects described in terms of the elaborated structure.

Keywords : solid design, geometric modelling, non-manifold topology, feature based design, unhomogeneous objects, boundary representation, data-structures, algorithms.

1 Introduction

Solid modeling is based on informationally complete computer representations able to support the automatic calculation of any characteristic relevant to the geometric shape of the modelled object. A great deal of applications require the manipulation of non-manifold objects i.e. objects that are dimensionally non-homogeneous pointsets (referenced examples could be found in [13]). A classic problem are the Boolean operations of manifold objects that may yield non-manifold results (for illustration see [6]). A rich mathematical theory exists for manifolds however work to relate this theory to solid modelling is more recent.

The focus of this paper is the boundary representation of non-manifold solids. One can unambiguously define a solid by given the solid boundary description and topologically orienting it in such a way that the solid interior is defined for each point of the boundary. More precisely, the boundary surface subdivision into cells with dimension 0 (i.e. the vertices),1 (the edges),2 (the faces) embedded in E^3 is provided. Two parts are clearly differentiated:

• The topological model that encodes all information about adjacencies between topological elements.

• The geometric model that describes the precise shape and geometric location of the various elements.

Possibility to interact with the solid representation at different levels of the corresponding structure makes this solid modelling method very attractive. Moreover, the idea to enable this freedom of manipulation not only for solids but for arbitrary dimensioned entities provokes an important evolution of the boundary representation method ([9]).The aim is to encompass the manifold and the non-manifold domain. Several data-structures for representing non-manifold topologies have been proposed ([1], [2], [3], [15], [17]). Processing unhomogeneity of the dimension however implies complex test for model consistency. Theoretical formulation of validity tests exists (see [7] and [8]) but as long as we know no results on implementation are published.

In this paper we propose a boundary based non-manifold scheme. This structure permits the representation of d non-manifold conditions. The cases of $d = 0, 1$ are treated similarly to known data-structures while for $d = 2$ an extension is made through introduction of new types of topological elements. A set of basic algorithms for construction and manipulation of objects defined in terms of the developed structure is presented. According to our experimental results, their corresponding operators form a sufficient set. The paper is organized as follows: First, an example of form feature based design is given in order to illustrate the main requirements for the solid domain extension. A brief discussion of the existing approaches for treating the non-manifold conditions is given next. The basic concepts of the elaborated structure are introduced further. The set of operators for construction and manipulation is presented next. Finally, implementation results are commented.

2 Non-manifold modelling in feature based design

An example of CAD/CAM application that is strongly related to the manipulation of dimensionally non-homogeneous pointsets is the solid design with form features. According to [18], features characterize certain area of interest of the part model. In particular the geometric features, following [14], are used to represent some types of shapes or portions of the part model that may be important to the designer or to an application.

Let us consider the solid S on Fig.1. It could be created as a block with a set of geometric features: slots A and D, a boss B and a compound feature C. Many other examples are discussed in [4], [5], [11]. As it is shown by [14] editing solid models could be based on surface and/or volume features processing. The surface features (for example the boundary faces and the floor of a slot) are considered as collections of faces of part models and the volume features (such as bosses and holes) are

seen as full-dimensional pointsets of the part models or their complements.

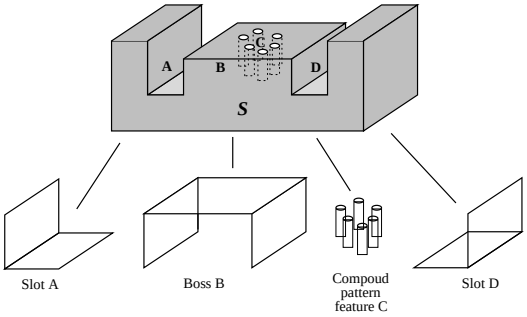


Fig.1.

Manipulating features permits also to pre-evaluate the computational expense of operations that add or subtract material or move and combine subsolids through Boolean operations (see the intentional feature defined by [14]). Feature validity is often expressed as system of constraints on referenced geometric elements. This permits a more rigorous and in the same time more comprehensive control of the final shape validity.

Let us see an illustration of feature editing by deletion and creation. The solid T_1 on Fig.2a is constructed by a subtraction of a block A and a slot B.

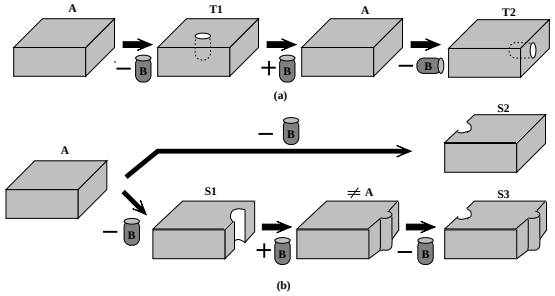


Fig.2.

T_1 may be transformed into T_2 by refilling T_1 with B to obtain A and then subtracting again B with the proper orientation. Following the same sequence of operations however would not permit to transform the solid S_1 from Fig.2b into solid S_2. In fact refilling S_1 with the slot B will not result in the original model A.

Another example of compound solid editing with feature is shown on Fig.3.

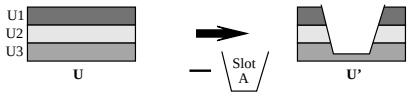


Fig.3.

A three layered object U, aggregated of the regions U_1, U_2 and U_3, is combined with a slot A. The goal is not to obliterate the internal structure i.e. one should differentiate the three layers in the result also.

These examples lead to the following conclusions:

• Feature based solid design encloses non-manifolds treating. The features as 2D or 3D aggregates could be naturally represented in terms of the undergrounding data-structures.

• Form features editing should be expressed as a set of tools for non-manifold objects manipulation and thus limiting cases of undesirable effects.

• Internal structures of composite objects have to be preserved no matter the feature based operation applied on them.

3 Treating non-manifold conditions

Following [17] non-manifold conditions occure when the solid boundary is not topologically flat i.e. there is a point such that each neighbourhood of this point is not homeomorphic to a disk in the relative topology. In the next section a more precise definition will be given, here our aim is to illustrate the more often encountered cases of point and edge non-manifold conditions.

Let us consider the solid on Fig.4a.

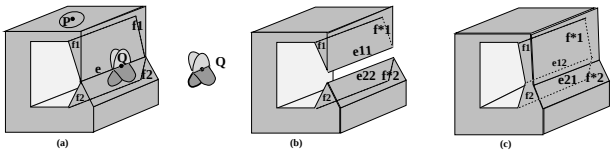


Fig.4.

Along the edge e the solid becomes infinitely thin i.e. for point Q on e one cannot determine on which side the solid interior lies. Each neighbourhood of Q is of the shown type and thus not homeomorphic to a disk. We will denote this case as edge non-manifold condition.

In the manifold domain we will fail to encode this case. One approach to treat this singularity is to consider that solids are topological manifolds but their embedding in the three dimensional space E^3 permits geometric coincidence of topologically separate elements. This approach is often applied in manifold restricted modellers giving a topological interpretation of the non-manifold conditions occurence. On Fig.4b faces f_i and $f*_i$, $i = 1, 2$, are mated and edges e_{11} and e_{22} though topologically distinct are embedded at the same geometric location.

The alternative interpretation is illustrated on Fig.4c where f_1 is matched with f_2 and $f*_1$ with $f*_2$. Thus edges e_{12} and e_{21} coincide geometrically. The choice of interpretation could affect model validity test as shown in [7].

Another non-manifold condition, when touching is in a point, is given on Fig.5.

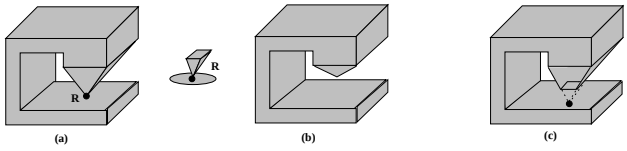


Fig.5.

The corresponding interpretations are Fig.5b where we unstick topologically the solid, and Fig.5c where the point intersection is replaced with a small enough face in order to preserve approximatively the real solid but eliminating the point non-manifold condition.

Maintaining solid validity is very difficult when this approach is applied for the resolution of non-manifold conditions. Numerical errors lead to difficult problems of interpretation of geometric

intersections. Processing them prove to be an exhausting task especially when topological distinct elements could occupy the same geometric location. Thus developing non-manifold representations is crucial from both application and implementation points of view:

• More and more CAD applications operate on non-manifolds thus they need the appropriated solid domain.

• The robustness of their implementation necessitates an explicit modelling of non-manifold objects.

4 Non-Manifold Boundary Representation

Several proposals for non-manifolds representation have been published in [1], [2], [3], [15], [16]. By contrast with the manifold domain where theoretical concepts are brought up to computational algorithms and modelling systems prototypes (see for example [10]), there is still a lack of similar counterparts in the non-manifold domain. Descriptions are often incomplete to permit a valuable comparative analysis.

In this section we introduce to the basic notions underlying the developed non-manifold representation. Comparisons are made with the "radial-edge" data structure of Weiler (see [17] and [16]) and the "non-manifold spine " structure of Desaulniers ([3]).

First we formulate the basic structure elements in term of the topological theory (see [7] and [8]). Next we give a more intuitive general description making a parallel with the radial-edge structure.

Let us see some examples of n dimensional manifolds for $n = 0, 1, 2$. The corresponding neighbourhoods are emphasized in bold style on Fig.6.

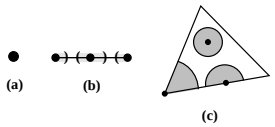


Fig.6.

Thus the non-manifold solid includes points that do not fulfill this condition. Such an object is shown on Fig.7.

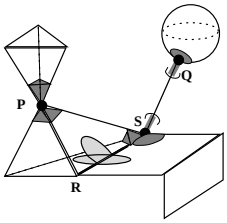


Fig.7.

This solid is a dimensionally non-homogeneous pointset with internal structure. It is composed by 3D pointsets (two tetrahedra and a sphere), a 2D pointset (a shell) and a wireframe linking the sphere and a tetrahedron. The corresponding non-manifold conditions are presented explicitly with their neighbourhoods. Such an aggregate could be represented using the simplicial complex.

Indeed, the solid could be defined as a 3-complex i.e. an union of n-cells, $n = 0, 1, 2, 3$, with disjoint interiors. In other words all the intersection information is encoded by the topological adjacency relationships. More over, each connected component of the solid interior should be homeomorphic to a 3-manifold. This permits a non-ambiguous definition of the solid interior (see [7]). Considering non-manifold solid as "immersion" of several manifolds and the corresponding definition is due to Hoffmann ([7]). However, as he points out, this approach comprises also geometric coincidence of distinct topological elements. This problem could be resolved by the introduction of new types of topological elements such that an unique topological element of given type is situated in given geometric location.

The Weiler's radial-edge structure uses the radial vertex and the radial edge to proceed the point and the edge manifold conditions.

Fig.8.

We could extract all edges embedded in given geometric position through the "edge use radial pointers" as shown on Fig.8. The same is true for the "vertex use radial pointers" to regroup vertices embedded in the same geometric point.

The structure proposed in [3] encodes the non-manifold points in a "non-manifold spine", illustrated in bold line on Fig.9, while representing them as "infinitesimal faces" in the corresponding solid components.

Fig.9.

Each term in the spine stick together geometrically equivalent elements.

The restriction as in the case above is that non-manifold conditions occuring in a point or in an edge are just proceeded.

Two questions are postponed:

• Could a non-manifold condition occure in a higher dimension?

• How does the internal structure preserving react on multiple consecutive modifications?

We define the following non-manifold structure to answer to these general requirements:

The radial vertex, radial edge and radial face elements are introduced to stand for classes of topologically different elements

embedded in the same geometric location and thus giving rise to non-manifold condition. Thus for given object and given point in E^3 there could be at most one radial vertex belonging to the object and located in the point. The same is true for the higher dimension elements edge and face.

One can easy construct examples of objects with non-manifold conditions at a vertex and at an edge, each one encoded by the corresponding radial element. It is sufficient to take trees of nested surfaces sharing a point or a contour as shown on Fig.10.

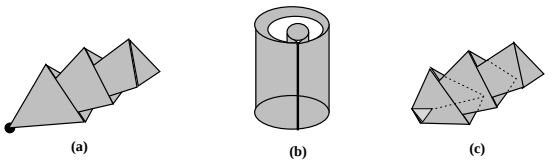


Fig.10.

The radial link is emphasized with a bold line. It should be pointed out that in some cases the radial link could be destroyed by replacing the geometrically identical topological elements with just one occurence of this element. In fact, this operation is a kind of manifold gluing and is applicable where gluing does not provoke a non-manifold condition.

For example let us combine two dangling faces f_1 and f_2 along a common edge. On the first stage we create a radial edge to stick together the corresponding edges e_1 from f_1 and e_2 from f_2 as shown on Fig.11a and Fig.11b.

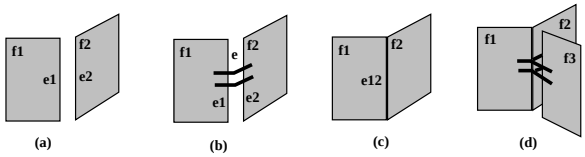


Fig.11.

If we want to preserve the internal structure of the resulting object i.e. as aggregate of two distinct components f_1 and f_2, we should maintain the radial link. In this case however, a manifold gluing could be accomplished by destroying the radial edge e and one of the corresponding edges e_1 or e_2 and thus creating a manifold shell (Fig.11c) composed of the two faces adjacent at e_{12}. If we want to combine three faces along an edge it obvious that manifold gluing could stick together just two of them and thus the radial link with the third one is unavoidable as shown on Fig.11c.

This reasoning is extending in 3D. Physically one could consider a space of higher than three dimensions. Let us take for example the E^3 extended with the temporal dimension. The radial face will express then a face non-manifold condition i.e. some kind of objects penetration (solid rigidity should be understood in a more general sense as defined in [12] such if interatomical distance authorizes material fusion from a macroscopic point of view). An illustration is given on Fig.12.

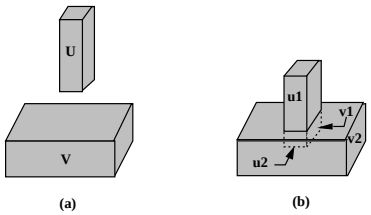


Fig.12.

After a "collision" the aggregate of U and V is composed of U_1, U_2, V_1, V_2 such that $U = U_1 \bigcup U_2$, $V = V_1 \bigcup V_2$, $U \bigcap V = U_2 = V_2$ where the pairs U_1 and U_2, V_1 and V_2 are manifold connected to form U, resp. V, and the faces of U_2 and V_2 are radially connected. We could save the internal structure by preserving four separate components and the corresponding radial links. If we want to destruct the internal boundary we will glue in a manifold way U_1 and V_1 and thus obtain a monolithic solid. In practice, gluing two volumes in a manifold manner along a common face provokes the face deletion.

After this general discussion we can give the definitions of the basic elements. We define:

A **n-cell** as a connected bounded set such that:

- for each point x of its interior there exists a neighbourhood homeomorphic to $D^n(x,r) = \{y \in R^n : \|x-y\| < r\}$;
- its boundary is divided into a finite number of lower dimensional cells, called the faces of the n-cell;
- a face belongs to the n-cell, if for each point x of the face there exists a neighbourhood homeomorphic to $D^{n+}(x,r) = \{x(x_1,\ldots,x_n) \in R^n : \|x\| < r \ \& \ x_n \geq 0\}$.

A **vertex** is a 0-cell, an **edge** is a 1-cell and a **face** is a 2-cell.

A **primitive** K_p^d $(0 \leq d \leq 3)$ is a connected finite set of d-cells, $K_p^d = \bigcup\{\sigma : \sigma \text{ is a d-cell }\}$ such that:

- if σ is a cell of K_p^d, then the faces of σ that belong to σ are elements of K_p^d;
- if σ and τ are cells in K_p^d, then $Int(\sigma) \cap Int(\tau) = \emptyset$;
- for each point x of K_p^d there exists a neighbourhood homeomorphic to either $D^n(x,r)$ or $D^{n+}(x,r)$.

A **complex** K^d $(0 \leq d \leq 3)$, is a finite set of cells, $K^d = \bigcup\{\sigma : \sigma \text{ is a cell }\}$ such that:

- if σ is a cell in K^d, then all faces of σ are elements of K^d;
- if σ and τ are cells in K^d, then $Int(\sigma) \cap Int(\tau) = \emptyset$.

The dimension of K^d is the dimension d of its highest dimensional cell.

An **object** is a connected oriented complex. It could be a volume, a shell or a wireframe entity, an isolated point or any combination of them. The object is built up from consistently oriented primitives, each one corresponding to a manifold object component. A **scene** gives the list of all objects in the modelling space.

Halfelements are used to reflect a specific orientation of the topological elements with respect to the object to which they belong. A **halfvertex** is an oriented vertex such that in a primitive, one could say on which side or in which sector around the vertex the object interior lies. A **halfedge** defines one of the two possible orientations of an edge in a primitive. Each edge is consisted of two halfedges. A **halfface** denotes one side of a face in a primitive. In a shell primitive each face has two halffaces. In a volume primitive the halfface stand for the chosen orientation towards the solid interior.

Radial elements encode classes of different topological elements embedded at the same geometric location. Thus a **radial**

vertex is the set of vertices of an object embedded in the same point of E^3. By analogy we define the **radial edge** and the **radial face**.

In order to facilitate the manipulation of edge lists as wireframe primitives or as boundary contours of faces we introduce the notions of **loop, halfloop** and **radial loop** in a similar way. Thus the object illustrated on Fig.7 is compounded by five primitives: three 3-cells (the tetrahedra and the sphere), a shell primitive and an wireframe primitive. Radial connections at points P and Q reflect the point non-manifold condition and radial edge at RS is caused to edge non-manifold condition.

The similitudes with the radial edge structure are situated at the levels vertex/ halfvertex/ radial vertex corresponding respectively to vertex/vertex use/ radial mate vertex pointer. By analogy edge/ halfedge/ radial edge and face/halfface. The radial face however, is a new level extending the non-manifold condition dimension. The structurization of the modelling space in scenes, objects and primitives gives an intuitively clear construction technique in terms of the topology of simplicial complexes. More over, in the developed structure the non-manifold domain naturally englobes the manifold one thus making easy the transformation of a non-manifold into a manifold representation if possible and desired.

In the next section the set of operators undergrounding the structure construction and maintenance is described. They verify the general requirements formulated in [16] for the boundary graph operators. The advantage is that they sound to be a minimal set thus giving a general technique for non-manifolds construction. More over, they guarantee the model validity at each step of the solid design. Some algorithms are fully original and make explicit the general concepts developed in [16].

5 Non-Manifold Operators

A set of operators is developed for non-manifold objects manipulation.

It's based partially on the Euler Operators (see [16]), however some constraints are implied.

• First, the operator set should provide enough tools for the construction and maintenance of the non-manifold boundary representation described in the previous section.

• Second, every operator should keep the topological validity and the coherence of the model.

The first requirement is naturally deduced from the fact that the operators process objects encoded in terms of the worked out structure. The second one permits a more intuitive object construction because it's preferable to pass through valid intermediate results in order to obtain the desired object than to perform invalid ones (i.e. that cannot be represented with the model in use) as done in [10]. More over, in this way no final validity test is needed as long as each step produces a valid result.

The operator set is subdivided into two categories :

• Low level operators that form a basis for the structure manipulation.

• High level operators that perform more sophisticated treatments and could be made out of basis operators sequences.

The basic operators would be discussed in detail while just some examples of the second class of operators would be given next. An example of non-manifold object construction is given in appendix B.

5.1 Low Level Operators

The class of low level operators could be regroup into two subsets. Firstly, operators to create and to destroy the structure basic elements. Secondly, operators for manipulation of radial elements.

5.1.1 Creation and destruction

This group of operators create and/or destroy basic elements (scene, object, primitive, face, loop, edge,vertex), and their half counterparts. The standard notations introduced by [16] are used. Elements are denoted as the abbreviations : s for scene, e for edge, he for halfedge .... Element identificator is denoted as the corresponding abbreviation suffixed by $"no"$. For example s for scene and sno for its identificator. Each element in the structure has an unique identificator. The letter $"M"$ and $"K"$ are used for the main operations of creation (Make) and destruction (Kill).

The order of operators enumeration correspond to the usually used sequence of object construction as a complex, consecutively increasing the dimension.

Explanations are given for the M-operators, the ones for K-operators could be deduced by simply applying the converse logic.

- SCENE

  Ms(*Id sno*)

  Ks(*Scene s*)

  Create (Fig.13a) [/destroy (Fig.13b)] a scene in the modelling space.

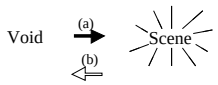


Fig.13.

- VERTEX

  Mvpo(*Scene s, Id vno, Id pno, Id ono, Id hvno, float x, float y, float z*)

  Kvpo(*Vertex v*)

  Create (Fig.14a) [/destroy (Fig.14b)] an object consisting of an isolated vertex primitive.

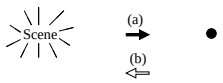


Fig.14.

- EDGE

  Mel(*Node n1, Node n2, Id eno, Id lno, Id heno1, Id heno2, Id lno*)

  Kel(*Edge e*)

  Create (Fig.15a.16a) [/destroy (Fig.15b.16b)] an edge and thus provoking the creation [/destruction] of a loop. These operators lead to two cases presented as follows. The class Node is an union of all structure elements. For these operators, $n1$ and $n2$ point on two differents halfedges in the first case (Fig.15) or on the same halfvertex in the second (Fig.16).

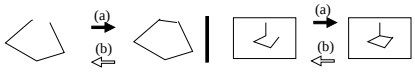


Fig.15.

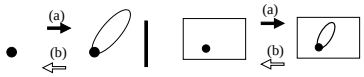


Fig.16.

MeKl(*Halfedge he, int what, Node n, Id eno, Id heno1, Id heno2*)

KeMl(*Edge e, Id lno, Id hlno*)

Create [/destroy] an edge to link [/unlink] two loops on a halfface where n points to halfedge for the case in Fig.17a or to halfvertex for the case in Fig.17b.

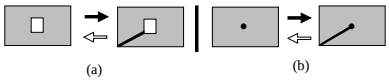


Fig.17.

MeKlpo(*Halfedge he1, Halfedge he2, Id eno, Id heno1, Id heno2*)

KeMlpo(*Edge e, Id lno, Id pno, Id ono, Id hlno*)

Create (Fig.18a) [/destroy (Fig.18b)] an edge to link [/unlink] wireframe primitives from two separate objects. For 18.a, the MelKpo gives rise to the destruction of the loop designed as second parameter, and the corresponding primitive and object.

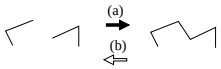


Fig.18.

MeKpo(*Halfedge he, Halfvertex hv, Id eno, Id heno1, Id heno2*)

KeMpo(*Edge e, Id pno, Id ono*)

Create (Fig.19a) [/destroy (Fig.19b)] an edge to link [/unlink] a wireframe primitive with a vertex primitive belonging to different objects. This causes the deletion of the primitive defined as second parameter and its corresponding object.

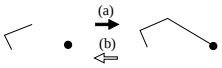


Fig.19.

MelKpo(*Halfvertex hv1, Halfvertex hv2, Id eno, Id lno, Id heno1, Id heno2, Id hlno*)

KelMpo(*Edge e, Id pno, Id ono*)

Create (Fig.20a) [/destroy (Fig.20b)] an edge to link [/unlink] two vertex primitives from separate objects. This results in creation of a loop and destruction of the primitive and the object corresponding to the second parameter.

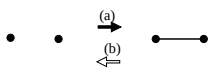


Fig.20.

- FACE

  Mf(*Loop l, Id fno, Id hfno1, Id hfno2*)

  Kf(*Face f*)

  KfMo(*Face f*)

  Create (Fig.21a) [/destroy (Fig.21b.c)] a face through transformation of wireframe into a shell. KfMo should be applied if the face destruction causes object creation (Fig.21c).

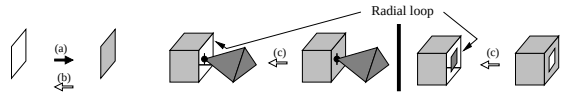


Fig.21.

- VOLUME

  MVolume(*Halfface hf*)

  KVolume(*Primitive p*)

  Create (Fig.22a) [/destroy (Fig.22b)] a volume primitive from a shell primitive.

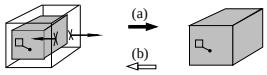


Fig.22.

5.1.2 Radial elements manipulation

These operators stand for radial element creation [/destruction] and thus making [/killing] the corresponding non-manifold conditions at vertex, edge or face.

Symbols presented in Fig.23a are used for the graphics illustrations of these conditions.

- VERTEX

  MRadv(*Halfvertex hv1, Halfvertex hv2*)

  KRadv(*Halfvertex hv1, Halfvertex hv2*)

  Create (Fig.23b) [/destroy (Fig.23c)] a radial vertex to make [/kill] a vertex non-manifold condition. For Mradv, two different cases could occure according to whether we duplicate a halfvertex from a given primitive or we combine two different halfvertices from separate primitives to create a non-manifold condition at the corresponding vertex. The higher dimensions are treated by analogy. For the radial edge it's shown on Fig.23d [/e] and for the radial face on Fig.23f [/g].

- EDGE

  MRade(*Edge e1, Edge e2*)

  KRade(*Edge e1, Edge e2*)

  Create (Fig.23d) [/destroy (Fig.23e)] a radial edge to make [/kill] an edge non-manifold condition.

- FACE

  MRadf(*Face f1, Face f2*)

  KRadf(*Face f1, Face f2*)

  Create (Fig.23f) [/destroy (Fig.23g)] a radial face to make [/kill] a face non-manifold condition.

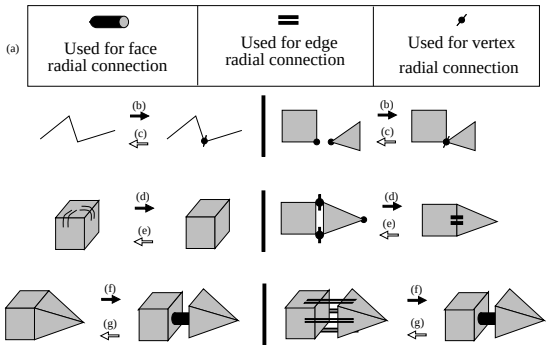


Fig.23.

5.2 High Level Operators

Some of the most often encountered high level operators are enumerated below. Each one is represented by its corresponding sequence of basic operators. It should be noted that for several operators, this sequence is not unique and could vary according to the object on which it is applied. Parameters of high level functions are omitted for more clarity.

- **Mevl()** $\Longleftrightarrow$ *(Mvpo + MelKpo)*
  Kevl() $\Longleftrightarrow$ *(KelMpo + Kvpo)*

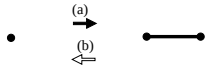


Fig.24.

- **Mev()** $\Longleftrightarrow$ *(Mvpo + MeKpo)*
  Kev() $\Longleftrightarrow$ *(KeMpo + Kvpo)*

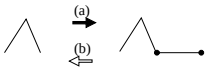


Fig.25.

- **MeKlp()** $\Longleftrightarrow$ *(KRadv + MeKlpo + MRadv)*
  KeMlp() $\Longleftrightarrow$ *(MRadv + KeMlpo + KRadv)*
  MeKp() $\Longleftrightarrow$ *(KRadv + MeKpo + MRadv)*
  KeMp() $\Longleftrightarrow$ *(MRadv + KeMpo + KRadv)*
  MelKp() $\Longleftrightarrow$ *(KRadv + MelKpo + MRadv)*
  KelMp() $\Longleftrightarrow$ *(MRadv + KelMpo + KRadv)*

  The mode of action of these operators is similar to the corresponding low level operators, except the fact that no object destruction occures because the primitives are already connected (i.e. belong the same object). So, the operator names are the same without the suffix "o".

- **MvRadv()** $\Longleftrightarrow$ *(Mvpo + MRadv)*
  KvRadv() $\Longleftrightarrow$ *(KRadv + Kvpo)*

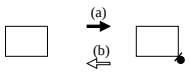


Fig.26.

- **MeRade()** $\Longleftrightarrow$ *(MvRadv + MvRadv +MelKp + MRade)*
  KeRade() $\Longleftrightarrow$ *(KRade +KelMp + KvRadv + KvRadv)*

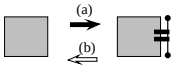


Fig.27.

- **MfRadf()** $\Longleftrightarrow$ *("face creation" + MRadf)*
  KfRadf() $\Longleftrightarrow$ *(KRadf + "face deletion")*

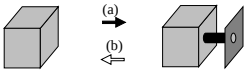


Fig.28.

6 Implementation Results

The representation that have been proposed is comparable to known representations [3], [10], [17] in terms of algorithmic complexity. Indeed, on average, one can consider that the non-manifold cases do not appear frequently enough to have significant influence on the algorithmic complexity.

In relation to memory, requirement of the elaborated data structure are higher in comparison with the manifold case. This drawback is widely compensated by the available information processing. A comparative analysis between the manifold structure described in [10], the non-manifold boundary structure from [17] and the structure proposed in this article, is made on the example of a cube representation in appendix A. The comparison results are sumerized below :

	Manifold Mantyla's structure	Non manifold Weiler's structure	Non manifold proposed structure
Memory Requirement	100 %	183 %	134 %

Finally, the code is written in C language on Silicon graphics workstations.

7 Conclusion

The current researches in the solid representation domain are directed towards the extension of the representational power to include the dimensional non-homogeneous objects known as non-manifold solids. The existing models are restricted to the treatment of d dimensional non-manifold conditions, $d = 0$ at a vertex and $d = 1$ at an edge. No generalizations are made for higher dimensions and especially in the case $d = 2$. Among other things as it was seen, this case appears when the topology of simplicial complexes is used as basic concept for the definition of non-manifold objects and especially for the validity test formulation. In this paper we propose a data-structure that resolves this problem. The corresponding tools for structure maintenance and manipulation are developed. According to the emperical results the set of underlying operators forms a basis and thus each object modification could be given in the form of a sequence of the basic steps. The intermediate results are valid in the terms of the elaborated model and thus the validity of the final entity is guaranted. The comparison in algorithmic complexity and memory requirements with the well known structures of [10] for the manifold case and of [17] for the non-manifold one shows that the gain of information processing power is made with no signicant memory use raise.

8 Appendix A

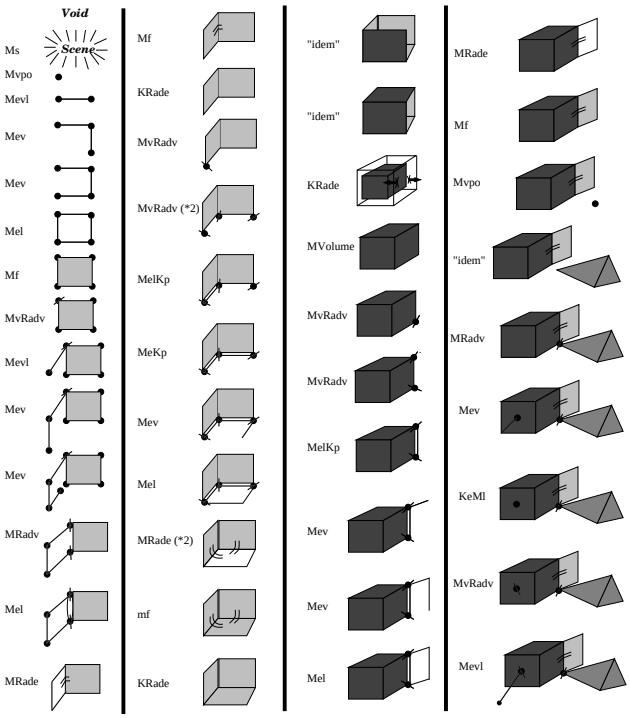
For homogeneous comparison we assume that no identificators and shape attributes are saved, the pointer size is $4Bytes$ and orientation or switch flag is encoded as an integer of $2Bytes$

Manifold Mantyla's structure		***Non manifold Weiler's structure***		***Non maifold proposed structure***	
Structure elements	*size of struct * number of elements (bytes)*	*Structure elements*	*size of struct * number of elements (bytes)*	*Structure elements*	*size of struct * number of elements (bytes)*
1 solid	5	1 model	3	1 scene	3
6 faces	30	2 regions	8	1 object	4
6 loops	24	2 shells	8	1 primitive	6
12 edges	48	6 faces	6	6 faces	18
24 halfedges	120	6 face uses	48	6 halffaces	30
8 vertex	24	6 loops	6	6 loops	18
		6 loop uses	48	6 halfloops	54
		12 edges	12	12 edges	36
		24 edge uses	264	24 halfedges	120
		8 vertex	8	8 vertex	8
		8 vertex uses	48	8 halfvertex	40
Total (bytes)	251		459		337
% relative to Mantyla's structure	100 %		183 %		134 %

(Memory requirement comparison on a cube)

9 Appendix B

The given example illustrates a non-manifold object construction. The object is composed of two volume primitives (a cube and a tetrahedron), one shell primitive (a dangling face), and one wireframe primitive (a dangling edge).



References

[1] F. Arbab. Set models and boolean operations for solids and assemblies. *IEEE ComputerGraphics&Applications*, pages 76–86, November 1990.

[2] G.A. Crocker and W.F.Reinke. An editable non-manifold boundary representation. *IEEE ComputerGraphics&Applications*, pages 39–51, March 1991.

[3] H. Desaulniers and N.F. Stewart. An extension of manifold boundary representation to the r-sets. *ACM Transactions on Graphics*, 11(1):40–60, January 1992.

[4] B. Falcidieno, F. Gianni, C. Porzia, and M. Spagnuolo. Configurable representation in feature-based modelling. In *Eurographics*, 1991.

[5] J.P. Gomes and J.C.G. Teixera. Form feature modelling in a hybrid csg/brep scheme. *Computer&Graphics*, 15(2):217– 229, 1991.

[6] E.L. Gursoz, Y. Choi, and B. Prinz. Boolean set operations on non-manifold boundary representation objects. *Computer-Aided-Design*, 23(1):33–39, January/February 1991.

[7] Ch.M Hoffmann. *Geometric and Solid Modeling: An Introduction.* Morgan Kaufmann, 1989.

[8] L.Ch. Kinsey. *Topology of surfaces.* Springer Verlag, 1993.

[9] P. Lienhardt. Topological models for boundary representation: A comparison with n- dimensional generalized maps. *Computer-Aided-Design*, pages 59–82, January/February 1991.

[10] M. Mantyla. *Geometric and Solid Modeling: An introduction.* Computer Science Press, 1988.

[11] M. Mantyla. A modeling system for top-down design of assembled products. *IBM Journal of Res. Develop.*, 34(5):637–659, Septembre 1990.

[12] A.G. Requicha. Representation for rigid solid: Theory, methods and systems. *ACM Computing Surveys*, pages 437–464, December 1980.

[13] A.G. Requicha and J.R Rossignac. Solid modelling and beyond. *IEEE ComputerGraphics&Applications*, pages 31 – 44, Septembre 1992.

[14] J.R. Rossignac. Issues on feature-based editing and interrogation of solid models. *Computer&Graphics*, 14(2):149–172, 1990.

[15] J.R. Rossignac and A.A.G Requicha. Constructive non-regularized geometry. *Computer-Aided-Design*, 23(1):21–32, January/February 1991.

[16] K. Weiler. Boundary graph operators for non-manifold geometric modeling topology. In *Geometric Modeling for CAD Applications*, 1988.

[17] K. Weiler. The radial edge structure: A topological representation for non-manifold geometric boundary modeling. In *Geometric Modeling for CAD Applications*, 1988.

[18] P.R. Wilson and M.J. Pratt. A taxonomy of features for solid modelling. In *Geometric Modeling for CAD Applications*, 1988.