ABCM Symposium Series in Mechatronics - Vol. 1 - pp.498-507
Copyright © 2004 by ABCM

Representation of Curves and Surfaces in B-Rep Solid Modelers

Wang Congli
Escola Politécnica da USP – Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos

Marcos de Sales Guerra Tsuzuki
Escola Politécnica da USP – Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos
mtsuzuki@usp.br

Abstract. In the last 20 years, CAD/CAM systems have developed incorporating an increasing volume of information related to the physical shape they represent. Historically, the representation techniques are classified in two groups: Solid Modelling and Geometric Modelling. The Geometric Modelling techniques have been developed with the automobile industry since the 50's, where they were used mainly to define the shape of car bodies. Now, they are applied in the naval and aeronautical industries and other areas. The Geometric Modelling techniques have as main support the control over the shape. The Solid Modelling techniques appeared in the 70's. They can create Computer Models with the capability of classifying any point in the three-dimensional space as being: internal, external or on the boundary of the solid. Solid models are very important, they support the calculation of mass properties (such as inertia, volume, and other properties) and the automatic generation of meshes for finite element analysis. However, the representation domain was restricted to polyhedrons, or, approximated polyhedrons of curved shapes (ex. spheres, cylinders, cones and others). Recently, in some commercial CAD systems, where Geometric Modelling techniques are combined with Solid Modelling techniques to represent computer models, the final model is not assure to be a valid Solid Model. In this article a new data structure will be defined based on non-manifold representation to support the representation of curves and surfaces in B-Rep solid models. This new data structure will support a synchronization between the geometry of the Geometric Model and an approximated polyhedral Solid Model. An approximated polyhedral Solid Model is obtained allowing the calculation of mass properties using several existent algorithms in the literature. And it will be possible to reduce the approximated polyhedral for a minimum representation whenever necessary. The face, in commercial Solid Modelers, has two functions: it represents the boundary of the solid and represents the geometrical shape of the contour. In this new data structure, we will separate these two functions, one element will represent the boundary of the solid and another topological element will represent the geometrical shape as free form curves and surfaces. We will explain that through an example, suppose that a face has three, five or more sides. In this case, it is not common for a commercial CAD system to support the representation of a face with three or more sides, as the polynomial expression becomes very complicated. Usually, those faces are subdivided in a number of four sided faces. That fact is an inconsistency between topology and geometry, as the number of sides in a face is defined by topology and geometry usually "forces" a face to have four sides. In this article, we will propose two structures, the principal data structure contains the original solid model with its geometry, and the auxiliary data structure represents an approximation of the solid model. We will define some methods to synchronize both data structures

Keywords. *Solid Modelling, Geometric Modelling, CAD/CAM, Mechatronics*

1. Introduction

Solid modelling and geometric modelling have been developed separately as pointed by Farin (1990). It is very common to associate polyhedral to solid modelling; and curves and surfaces to geometric modelling. Polyhedral models have some disadvantages when compared to geometrical models. First, the significant topological features of the model are obscured - for instance, a simple cylindrical hole is represented by many faces. Second, the geometry of the model is approximated - the level of geometric accuracy depends on the number of facets used to approximate each curved face. In the literature we have very few proposals to integrate these two topics. Toriya and Chiyokura (1991) described the proposal used in the solid modeler developed at Ricoh. Mäntylä (1988) described a similar approach used in the GWB, solid modeler developed at Helsinki University of Technology. Turner (1988) proposed a method to generate polyhedral solids from geometrical information, where the geometry is processed as an attribute of faces and edges. The support to both technologies solid modelling and geometric modelling is a must according to the actual technological needings. The polyhedral solid model is necessary, it assures that the solid is closed with coherent orientation, then it is possible to calculate mass properties, among other properties. The representation of curves and surfaces is very necessary to have access to the exact shape representation. This information is useful to calculate tool path for milling, among several other applications (Lee, 2001).

In this work, we will propose a data structure to integrate both technologies: solid modelling and geometric modelling. It will be proposed a synchronization method where it will be possible to generate easily a polyhedral solid as an approximation for a free form surface. It will possible to edit and manipulate the free form surface and the B-Rep solid will be updated to show such manipulation. We will show through some examples that a face in the B-Rep solid modeler has mainly two functions: represent the topological boundary of the solid and to represent the geometrical shape of the solid. The main idea in this new data structure is the separation of these two functions, simplifying the solid modelling and geometric modelling algorithms.

2. Geometric Modelling

Curves and surfaces have a very important role in engineering design and manufacturing. We will present here only Bézier curves and surfaces. This representation was developed by Pierre Bézier (1986) who worked at the French

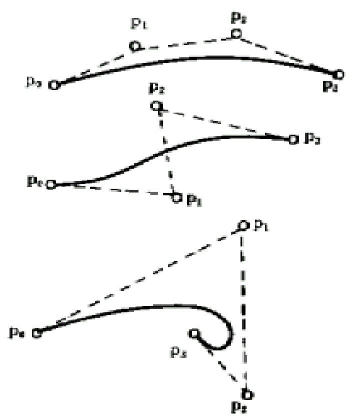

Figure 1. Examples of Bézier Curves.

company Regie Renault and developed the system called UNISURF. The same approach was developed by P. De Casteljau from the French company Citroen. However, as the UNISURF system was first published, this representation is associated to Bézier.

2.1. Bézier Curves

Bézier curves are defined by a set of control points. It is a very popular type of parametric curves. A Bézier curve is completely contained in the convex hull defined by the control points. It is easier for a designer to imagine the curve shape visualizing the convex hull. Examples of Bézier curves are shown in Fig. (1). The n-degree Bézier curve *Q(t)* is created from *n+1* control points and expressed as follows:

$$Q(t) = \sum_{i=0}^{n} B_i^n(t) P_i \qquad (0 \le t \le 1) \tag{1}$$

Where $B_i^n(t)$ is the Bernstein basis, given by:

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-1} \tag{2}$$

and

$$\binom{n}{i} = \frac{n!}{(n-i)!\,i!} \tag{3}$$

2.2. Bézier Surfaces

The Bézier surface is defined by a matrix of control points. A Bézier surface of $n \times m$ degree is represented by:

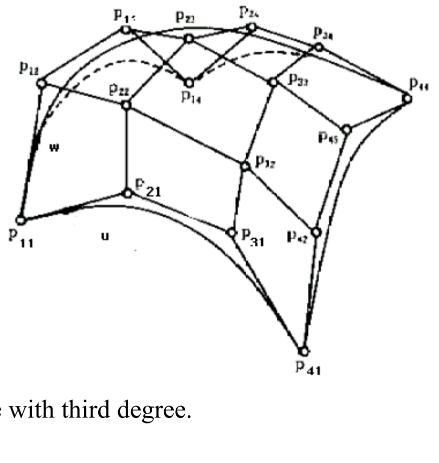

Figure 2. Example of a Bézier surface with third degree.

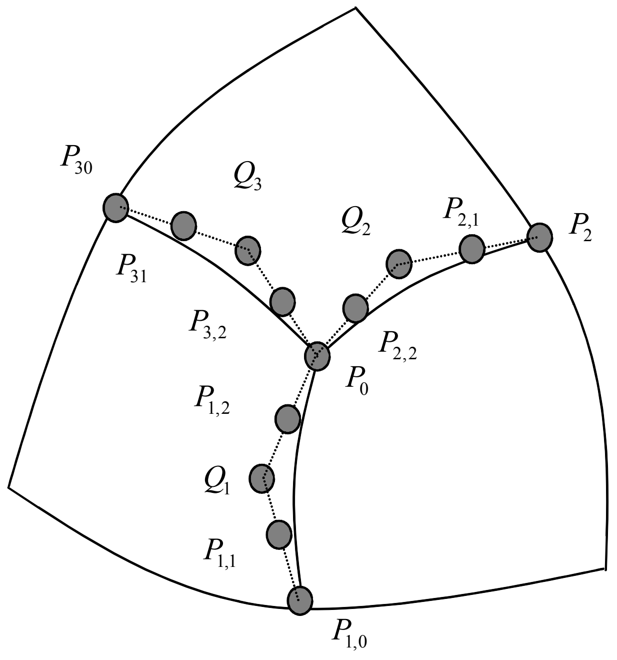

Figure 4. Case of a face with three sides.

$$Q(u,w) = \sum_{i=0}^{n}\sum_{j=0}^{m} B_i^n(u) \cdot B_j^m(w) \cdot P_{i,j} \tag{4}$$

Where, $P_{i,j}$ indicates a control point. If n+1 control points exist in the *u* direction and *m+1* control points in the *w* direction, we get a total of $(n+1)\times(m+1)$ control points. $B_i^n(u)$ and $B_j^m(w)$ are Bernstein basis. Fig. (2) shows an example of a Bézier surface.

2.3. Modelling Arbitrary Surfaces

Chiyokura and Kimura (1983) have shown that faces with three, five and six sides can be created in a solid when rounding operations are done. When such a surface is created its geometry cannot be directly defined. Kimura and Hosaka (1984) demonstrated the algebraic expressions to represent such free form surfaces. However, this is not the

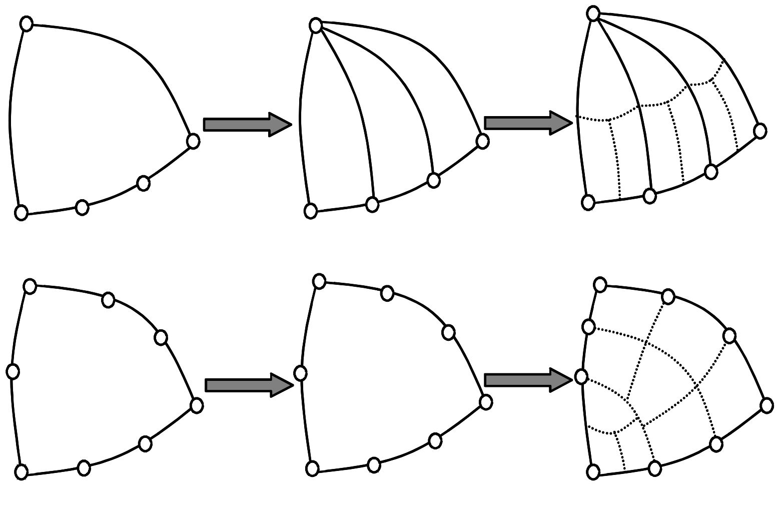

Figure 3. Generation of multi surfaces.

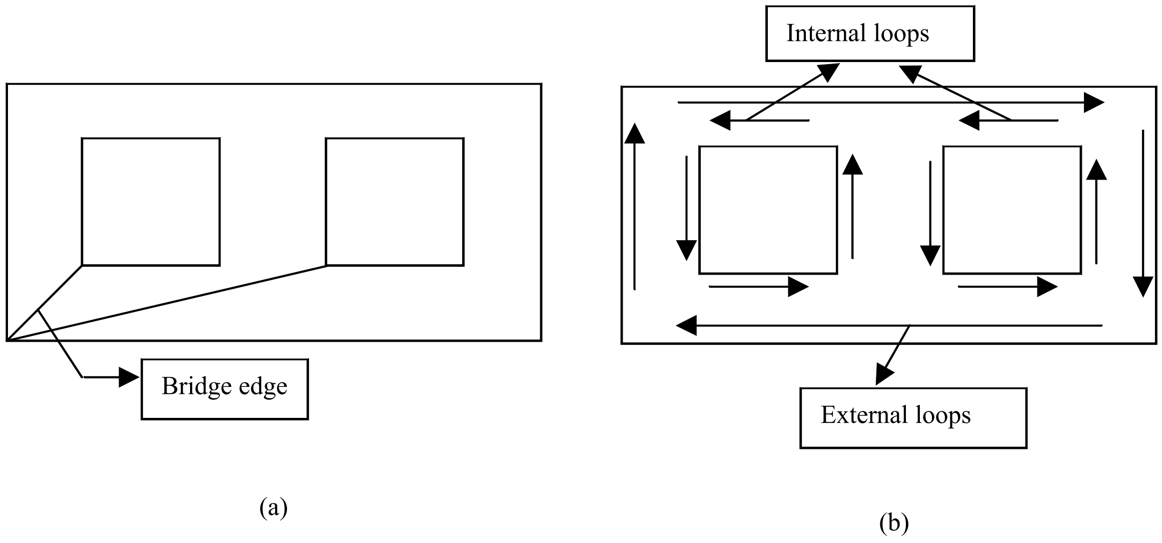

Figure 5. A face with internal holes, showing the definition of internal and external loops.

common approach, as the polynomial expression becomes very complicated. In the literature, we found a better approach where a face with n-sides can be transformed into n faces with four sides. Fig. (3) shows an example where a face with three sides is transformed in three faces with four sides. This technique has a collateral effect that is the creation of three extra points in the boundary of the original face. This way, the number of sides of the adjacent faces is increased by one.

Kado et al (1992) proposed an algorithm where faces with subdivided boundaries can be further subdivided, such that faces with four sides are created. Fig. (4) shows two cases: in the first case, one side of the triangular patch was divided in three segments; in the second case, all the three sides of the triangular patch were divided. In the first case, we got nine faces with four sides, and in the second case we got also nine faces with four sides. However, the drawback of generating extra vertices continues to happen.

Through these examples, we can see that when geometry must be added to the face, in some cases, it is necessary to pre-process the face such that it is divided in a number of four sided faces. Once the four sided faces are defined, the geometry can be associated to it. This is a necessary process to satisfy the forces between algebraic representation complexity and topology representation possibilities. So, we would like to represent as much free form surfaces as possible without increasing the algebraic manipulation complexity. We choose to represent only free form surfaces with four sides. This way, when the faces does not have four sides, it is necessary to divide it into a number of four sided faces.

Based on this discussion, it is possible to find two main functions for topological faces. They must define the topological boundary of the solid with n sides and they must associate to this topological boundary the geometry of the boundary. If these two functions could be separated, simpler algorithms can be defined for surface edition and manipulation. This work is focused on this problem.

3. Solid Modelling

B-Rep solid models emerged from the polyhedral models used in computer graphics for representing objects and scenes for hidden line and surface removal. They can be viewed as enhanced graphical models that attempt to overcome some problems by including a complete description of the bounding surface of the object. There are three primitive entities face, edge and vertex, and the geometric information attached to them form the basic constituents of B-Rep models. In addition to geometric information such as face equation and vertex coordinates, a B-Rep model must also represent how the faces, edges and vertices are related to each other. According to Mäntylä (1988), it is customary to bundle all information of the geometry of the entities under the term geometry of a boundary model and, similarly, information of their interconnections under the term topology. It is possible to say that the topology is a glue that tie together the geometry.

With the objective of algorithm simplification, mainly in the determination of the circuit of edges surround a face the half-edge entity was created. It was observed that the edge in the original winged-edge data structure had two main functions: represent the circuit of edges surround the face and to represent the real edge. The algorithm to determine the circuit of edges surrounds a face was very complex with several rules. Some researchers observed that separating these two functions the algorithm can become much simpler. This way, the modern solid modelers have one entity to represent the edge itself and another entity to represent the circuit of edges surrounds the face.

As the B-Rep model can become more complex, it is necessary to add new primitive entities: loop and shell. A face can have holes inside to represent protrusions or depressions. In this case a face has one outer loop and zero or more

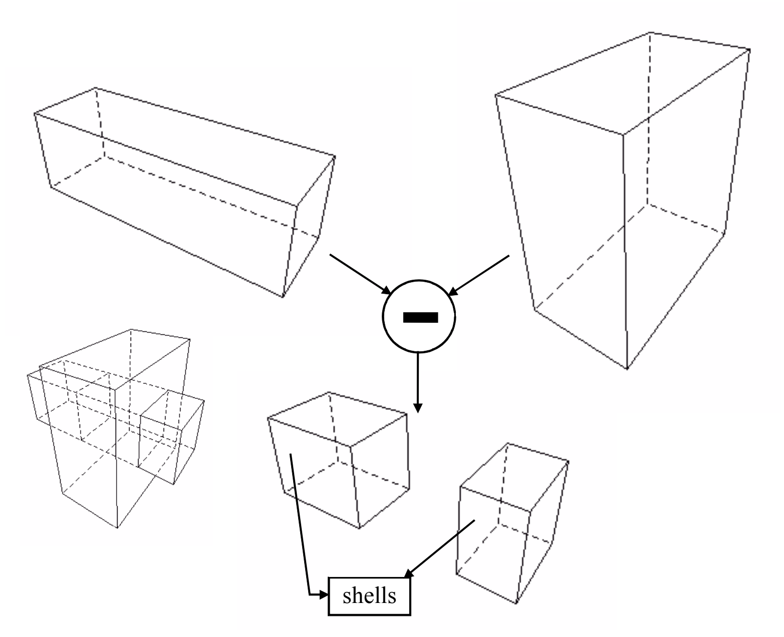

Figure 6. A case where a cube minus another results in one solid with two shells.

inner loops. A loop represents a sequence of half-edge (see Fig. (5)). A shell is a set of connected faces. Considering the case where a cube is subtracted from another cube such that the result is a solid with two shells (see Fig. (6)).

4. Sinchronizing Solid Modelling and Geometric Modelling

The synchronization between a solid model and the associated geometric model must support several procedures. Once the control points for a Bézier curve is defined, the polyline that approximates the Bézier curve can be calculated using expression (1). However, when a control point is moved, the polyline must be updated. How to determine the edges that belong to the polyline that approximates the Bézier curve? We will explain how to synchronize curves and surfaces expressions to their approximating entities (faces and edges).

Our proposal is based on the half-edge data structure. The data structure used in our solid modeler is based on the original half-edge data structure used by Mäntylä (1988) and Chiyokura (1988). The following primitive entities are presented in the data structure of our solid modeler: solid, region, shell, face, loop, edge, half-edge and vertex. A region defines a unique volume in space. In the example shown in Fig. (6), we have two regions, each region has one shell.

The work presented by Mäntylä (1988), Toriya and Chiyokura (1991) and Turner (1988) showed the same

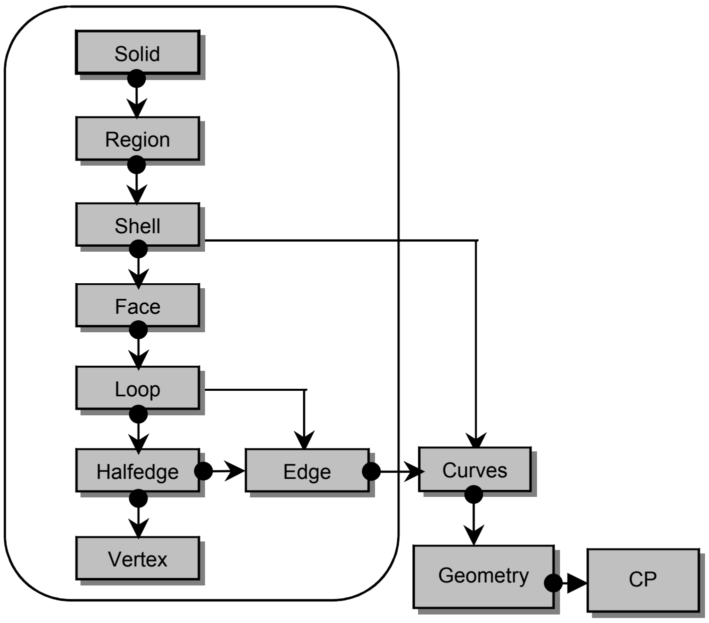

Figure 7. Data structure showing curve as edge´s attribute.

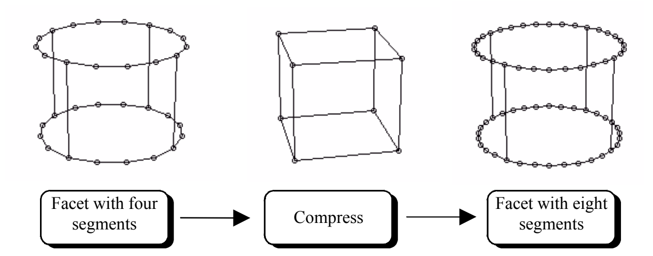

Figure 8. Mechanism showing how to increase the number of approximating edges.

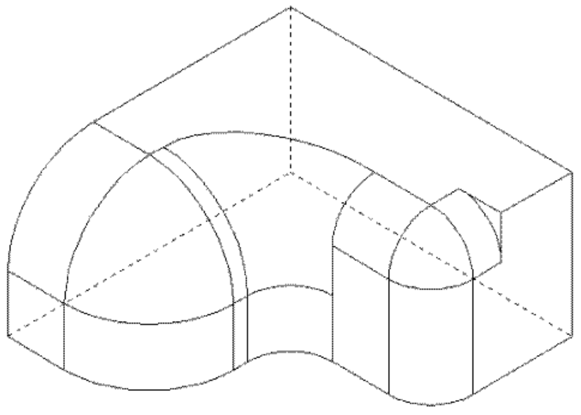

Figure 9. Solid with vertices and edges rounded, showing that non-four side faces are common.

approach to represent the geometry of curves and surfaces as an attribute of edges and faces. First, we will show that representing curve as attribute of an edge is a good approach. The adapted half-edge data structure is shown in Fig. (7). Thus, each edge has a pointer to a record specifying the exact mathematical representation of the underlying curve. This data structure gives full compatibility with the original half-edge data structure. It is possible to draw the solid traversing the list of edges, it is possible to obtain the adjacency information directly from the data structure.

As proposed by Turner (1988), a curve can be approximated by a number of edges; however, in some situations it is necessary to increase the number of approximating edges or even decrease such number. Fig. (8) shows an example where a cylinder is initially created with four approximating edges. In the slide of the middle, each curve is associated to only one edge and in the last slide each curve is approximated by eight edges. We need just two operators: one to compress the memory used to represent the curve, and another to approximate the curve by a number of edges. Such operators can be easily implemented using sequences of Euler operators.

Each approximating edge has a pointer to the original attribute, showing that it was originated from the same geometry. When the compress operator is applied it is necessary to traverse the circuit of half-edges surround the face searching for edges with the same geometric attribute. It will remain only one edge associated to such geometric attribute.

4.1. Associating Curved Surfaces to Faces in the B-Rep Data Structure

Turner (1988) proposed that, similarly to the curve associating approach, each face should have a pointer to a record specifying the exact mathematical representation of the underlying curved surface. Each edge should be tagged as facet edge or nonfacet edge. A facet edge is any edge that separates two facets of the same curved surface. A nonfacet edge is any other edge. Mäntylä (1988) and Toriya and Chiyokura (1991) proposed a similar approach. The problem stated in a previous section is very common, Fig. (9) shows a solid with rounded edges and vertices. As can be seen in this solid, several faces with non-four sides were generated. Thus, it is necessary to separate topological boundary representation from geometrical representation. This way, when geometry is associated to a specific face, this process does not affect the neighboring faces.

In this work we propose a data structure that has two extra levels to represent the geometry of faces with any number of sides and does not affect its neighboring faces. A face has a pointer to a record that is associated to a set of four sided faces. Then the face is not divided in the original B-Rep data structure. The face is first copied to this new data structure and then divided to define only four sided faces. In this way, the division does not affect the neighboring faces in the original B-Rep data structure. Every four-sided face in the new data structure has a pointer to another record where the four-sided face is represented as a polyhedral approximation. Such data structure is shown in Fig. (10).

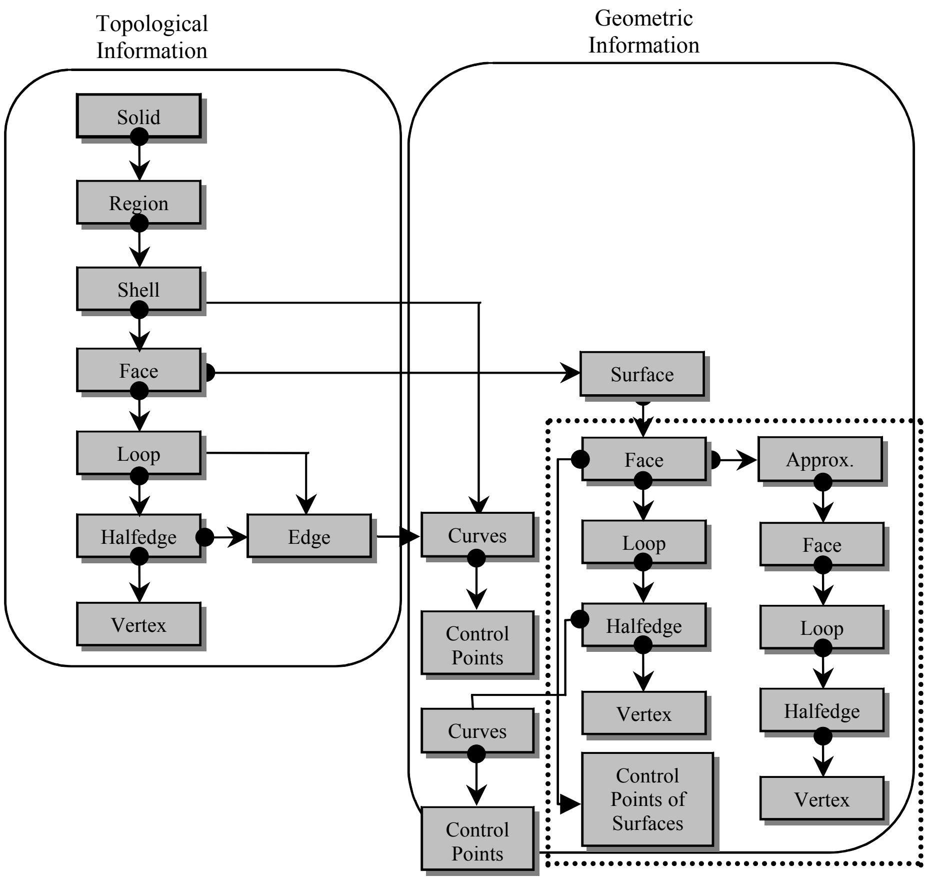

Figure 10. Proposed data structure with support to free form surfaces.

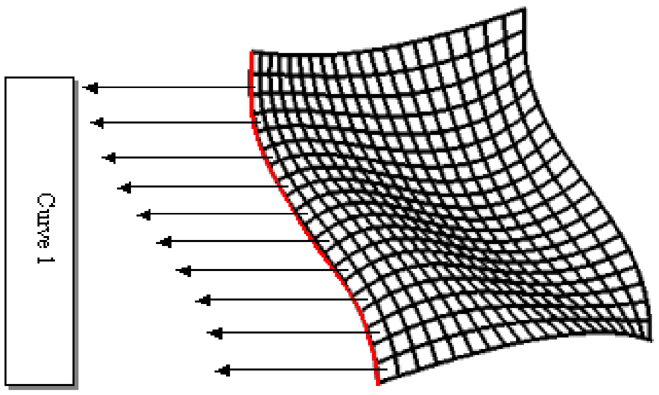

Figure 11. Example of a polyhedral approximation for a free form surface.

The geometry is defined following a sequence of definitions: first, the geometry of curves are defined; and, second, the geometry of surfaces are defined. The process is similar to the definition of the wings for airplanes or hull for ships. In both design process, firstly, it is created a wireframe structure with curves, and, secondly, the geometry of surfaces are associated to interpolate the wireframe structure. This way, the geometry of the solid is defined step by step. The curve level is very important to separate the wireframe model defined by the user from the model with geometric information associated to surfaces. In this CAD system, the geometry of surfaces is calculated automatically from its boundary. Thus, whenever necessary it is possible to divide the non-four-sided faces into four-sided faces without side effects.

The surface level is where the polyhedral approximation for a face is defined. It is necessary to consider the possibility that the polyhedral approximation from one face can affect the approximation of a neighboring face. Such problem is very similar from the previous one. Fig. (11) shows an example of polyhedral approximation definition. As we can see, another problem arises. When the approximation is done in the original B-Rep solid model, the edges that are approximating the curve are mixed with the edges that are approximating the surface. In our approach, the edges

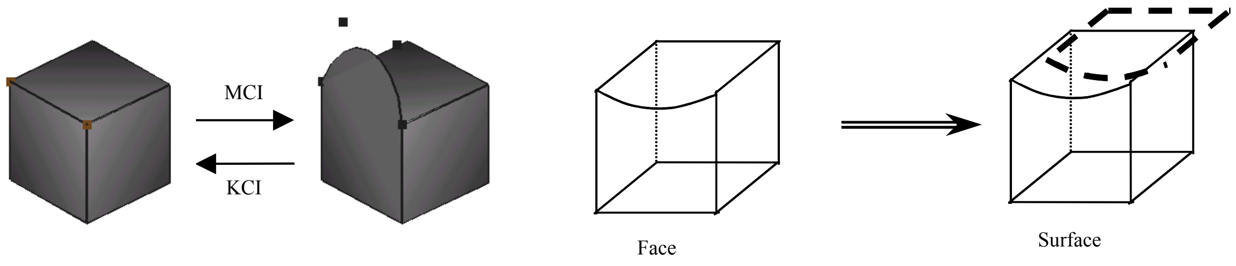

Figure 12. Example of operators to manipulate the proposed data structure.

that are approximating a curve are defined in the original B-Rep solid model. However, the edges and faces that are approximating a surface are defined in a separated data structure. This fact results in simplifications in the algorithms.

4.2. Operators to Manipulate the Proposed Data Structure

As a conventional solid modeler, we defined several Euler Operators to manipulate the creation and destruction of entities in the proposed data structure. The following operators are defined:

- **MCI** (Make Curve Information): is an operator to add curve information to an edge;
- **KCI** (Kill Curve Information): this is the opposite of MCI. It removes a curve from the solid. First, the curve is compressed such that it is associated to only one edge;
- **ADDPP** (Add Point to Polyline): it adds a control point to a polyline;
- **REMPP** (Remove Point from Polyline): it removes a control point from a polyline;
- **MSI** (Make Surface Information): it is an operator to add surface information to a face in the original B-Rep solid model;
- **KSI** (Kill Surface Information): it is an operator to remove surface information from a face in the original B-Rep solid model;
- **ADDPS** (Add Point to Surface): it is an operator to add a control point to a surface patch;
- **REMPS** (Remove Point from Surface): it is an operator to remove a control point from a surface patch;
- **MSuVF** (Make Surface Vertex Face): this is a similar operator to **MVSF** (Make Vertex Solid Face) that is used to define the first vertex and face in the surface data structure;
- **MSuEV** (Make Surface Edge Vertex): this is a similar operator to **MEV** (Make Edge Vertex) that is used to define an edge and vertex in the surface data structure;
- **MSuEF** (Make Surface Edge Face): this is a similar operator do **MEF** (Make Edge Face) that is used to define an edge and vertex in the surface data structure;
- **KSuVF** (Kill Surface Vertex Face): this is a similar operator to **KVSF** (Kill Vertex Solid Face) that is used to remove the last vertex and face in the surface data structure;
- **KSuEV** (Kill Surface Edge Vertex): this is a similar operator to **KEV** (Kill Edge Vertex) that is used to remove an edge and vertex from the surface data structure;

We created also some high level operators, that are a collection of Euler Operators. Below we describe some of those operators:

- **CopyFace**: this high level operator copy a face structure from the original B-Rep data structure to the correspondent surface data structure;
- **TransformThreeSidedIntoFourSidedFaces**: this high level operator creates three four sided faces in the surface data structure according to the algorithm proposed by Chiyokura and Kimura (1983);

We implemented another set of low level operators to manipulate the approximation data structure.

- **MAI** (Make Approximation Information): it is an operator to add approximation information to a face in the surface data structure;
- **KAI** (Kill Approximation Information): it is an operator to remove approximation information from a face in the surface data structure;
- **MApVF** (Make Approximation Vertex Face): this is a similar operator to **MVSF** (Make Vertex Solid Face) that is used to define the first vertex and face in the approximation data structure;
- **MApEV** (Make Approximation Edge Vertex): this is a similar operator to **MEV** (Make Edge Vertex) that is used to define an edge and vertex in the approximation data structure;

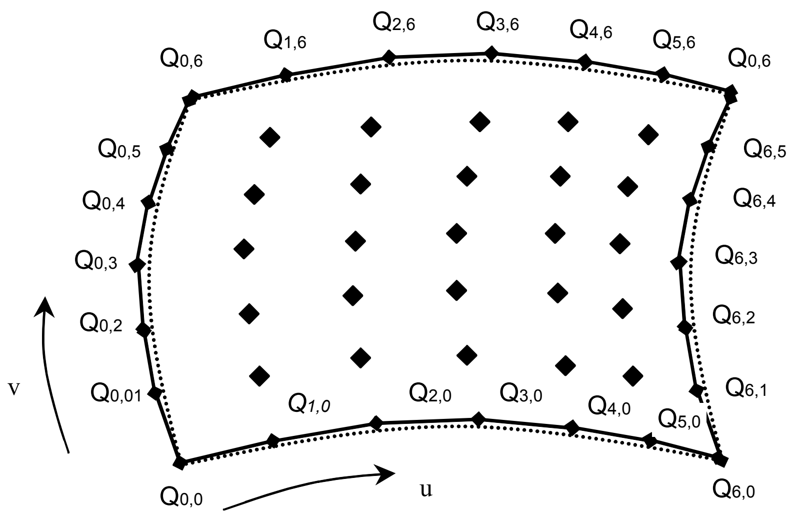

Figure 13. Example of vertex on free form surface calculation.

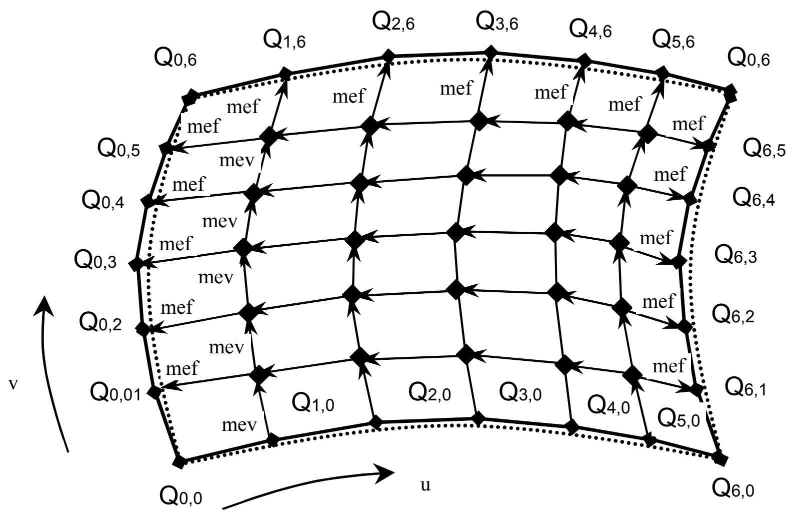

Figure 14. Example of a polyhedral approximation definition.

- **MApEF** (Make Approximation Edge Face): this is a similar operator do **MEF** (Make Edge Face) that is used to define an edge and vertex in the approximation data structure;
- **KApVF** (Kill Approximation Vertex Face): this is a similar operator to **KVSF** (Kill Vertex Solid Face) that is used to remove the last vertex and face in the approximation data structure;
- **KApEV** (Kill Approximation Edge Vertex): this is a similar operator to **KEV** (Kill Edge Vertex) that is used to remove an edge and vertex from the approximation data structure;

We created also some high level operators to manipulate the approximation data structure, that are a collection of Euler Operators. Below we describe some of those operators:

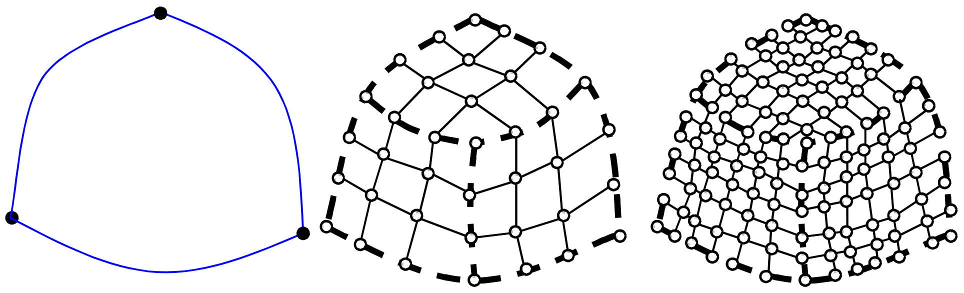

Figure 15. The three levels to represent geometry of a face. In the left the original B-Rep data structure. In the middle we have only faces with four sides. In the right the approximated faces.

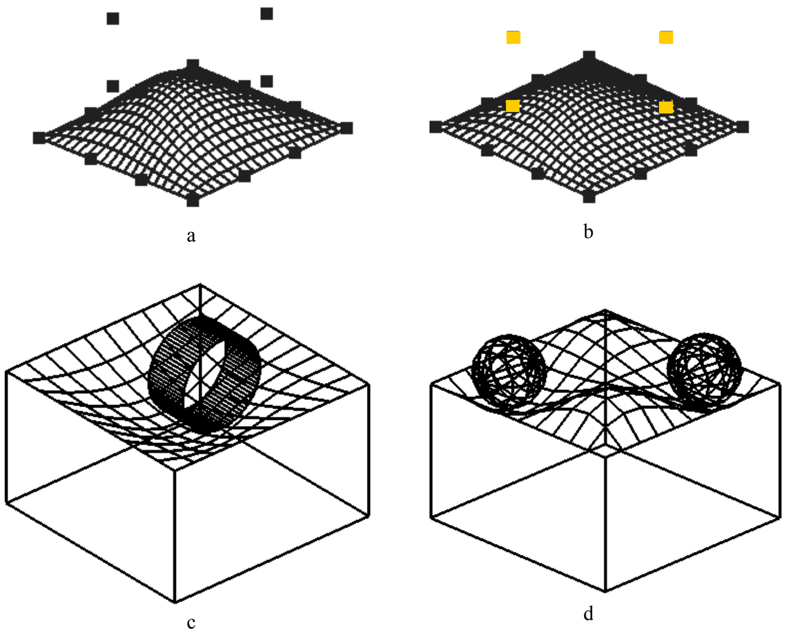

Figure 16. Shapes generated with the USPDesigner.

- **CalculateVertexOnSurface**: this routine calculates the coordinates of a vertex given its parameters on the free form surface. Fig. (13) shows an example with the calculated vertices;
- **CreateApproximationSurface**: this routine generate a sequence of MEV (Make Edge and Vertex) and MEF (Make Edge and Face) to define the polyhedral approximation for a free form surface. Fig. (12) shows an example of a polyhedral approximation definition.

Fig. (15) shows the three levels of the data structure: original B-Rep data structure, faces with four sides with attached geometry and polyhedral approximation data structure.

6. Conclusions

We have shown that polyhedral approximations may be synchronized to geometrical representation. The basis of the approach is a routine that calculate the polyhedral approximation and another to compress the data structure removing all approximating entities. We resulted in a simplified topological structure where a face does not interfere in its neighboring faces when geometry is associated. The proposed data structure was able to separate the face´s conflicting functions: topological and shape representation.

7. Acknowledgement

This work was partially supported by CAPES and CNPq under grant 300.224/96.

8. References

Bézier, P., 1986, “The Mathematical Basis of the UNISURF CAD system”, Butlerworths.
Chiyokura, H. and Kimura, F., 1984, A New Surface Interpolation Method for Irregular Curve Models**,** Computer Graphics Forum, vol. 3, pp. 209-218.
Farin, G. E., 1990, “Curves and surfaces for computer aided geometric design: a practical guide**”**. 2. ed., San Diego, Academic Press.
Hosaka, M., 1992, “Modelling of Curves and Surfaces in CAD/CAM”, Berlin, Springer-Verlag.
Kado, K., Shimamura, A., Inoda, K., 1992, A Surface Interpolating Method for a Car-Styling Designer’s CAD Work Tool**,** CG International ’92 June 22-26, Proceedings.
Kalay, Y. E., 1989, “The Hybrid Edge: A Topological Data Structure for Vertically Integrated Geometric Modelling”, Computer Aided Design, Vol. 21, No. 3, pp. 130-140.
Lee, S. H., Kim, H. S.; “Sheet Modelling and Thickening Operations Based on Non-Manifold Boundary Representations”, Proceedings of DETC’01 – ASME01 Design Engineering Technical Conferences, 2001.
Mäntylä, M., 1988, “An Introduction to Solid Modelling”, Computer Science Press.
Toriya, H.; Chiyokura H.; 1991, “3D CAD Principles and Applications”. Berlin, Springer-Verlag.
Turner, J.U., 1988, “Accurate Solid Modelling Using Polyhedral Approximations”, IEEE Computer Graphics & Applications, Vol. 8, No. 3, pp. 14-28.