MODELING TECHNIQUES TO GENERATE 3D MESHES

Coelho L.C.G.[1,2], Gattass M.[1,2] and Martha L.F.[1,2]
[1]Pontifical Catholic University of Rio de Janeiro - Brazil (PUC-Rio), [2]TeCGraf (PUC-Rio)

INTRODUCTION

Among the various approaches to the modeling of surface patches for generating tridimensional finite element meshes for plate, shell, or solid analysis, two aspects are determinant: the user interface methodology and the internal representation used in the geometric algorithms. The paper presents the authors approach for the complete generation of 3D meshes. The interface is heavily based on direct manipulation techniques for three fundamental aspects: creation of forms, especifying 3D transformations, and changing viewing parameters. The data structure and geometric algorithms use the adaptative polygonalization idea for parametric curve and surface representation. The attributes are attached to the geometric model using an extensible database that can be addapted for a great variety of finite and boundary element analyses (stress, temperature, flow, and ship stability have been implemented). The techniques presented here, when combined, produce a powerfull, flexible and easy to use preprocessor for numerical analysis.

THE INTERFACE

The construction of the user interface is based on a toolkit, called Interact, that uses an object oriented paradigm to model 3D interaction tasks. The classes of this implementation, shown in Figure 1, are based on an user interface library for two dimensionall applications [6]. The `Task` class is used to do selections, to enter points, and to transform objects. The `Plane` class and the use of the visualization libraries presented in [1] are used to simulate the 3D environment. The `Canvas` class is responsible for the drawing area methods. It represents the necessary methods and attributes to work with the standard OpenGL^{TM} library.

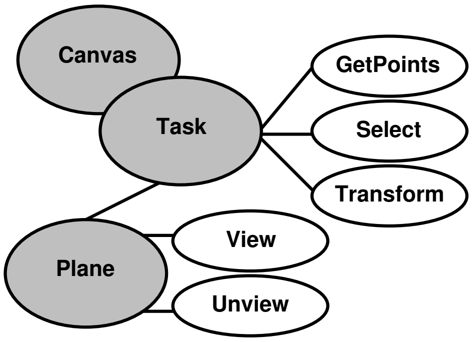


Figure 1: Class organization of the interface.

The implementation of methods related to the tasks creates a layer of abstraction that manages the the event programming, like mouse and keyboard controlling. The implemented modeler uses these classes for the construction of curves, surface meshes, and solid meshes, as well as other simpler tasks like changing the point of the observer, scaling the scene, etc. The input of 3D coordinates needed in the construction of curves is done with an auxiliar interface plane presented in [2], also available for direct manipulation transformations.

The direct manipulation technique used to implement the tranformations in 3D space is based on the ideas presented by van Emmerik [3]. The flexibility of the snap-to-grid plane and snap to other objects is acomplished by the interface classes with virtual methods. The modeler implements derived classes that are adapted to the nature of the entities. Figure 2 also shows the interface of a mesh generator program, called mg, that was created with the Interact toolkit.

Pilot tests were performed with the user interface of the modeler and this tests were used for redefine the interface modes, as well as to enhance the feedback of possible operations. The full description of these testes can be found in [5].

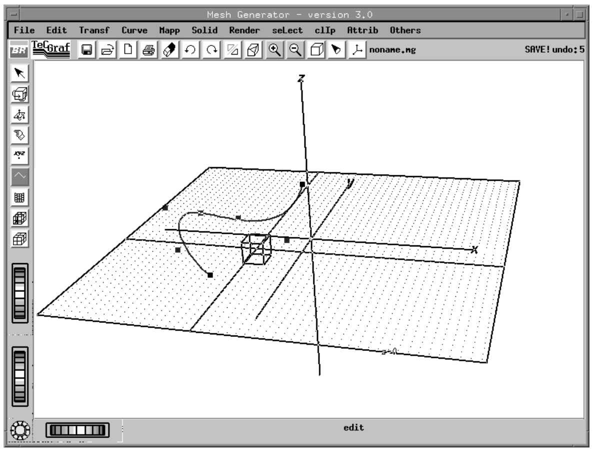


Figure 2: Creation of a Bspline.

GEOMETRIC MODELING

The modeling is based on the creation of curves that form the basis for generating surface patches. For plate and shell analysis, the surface patches are integrated in a single finite element model. For the generation of solid meshes the user creates surface patches and applies the available solid mesh generation methods. In that case, the final model is constructed integrating the individual solid meshes.

Curves

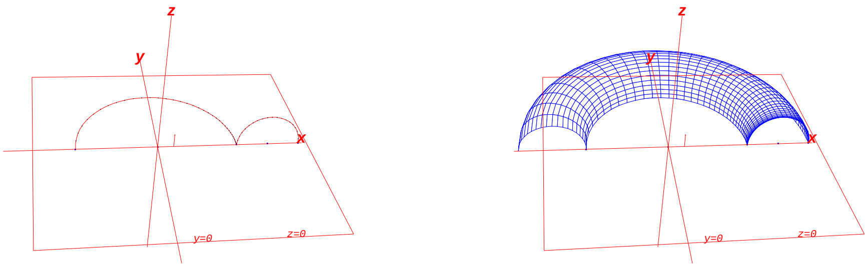
To create a curve in the environment, the user chooses a form from the available types: interpolating Bspline, arc of circle, or a polygonal line. The points are entered by positioning the cursor in the canvas area. The feedback of cursor position is a crosshair-line that walks on the interface plane and a 3D point coordinates is shown on a message window. Each entered point is added to the geometric description of the curve. The atraction to existing coordinates is done with the extreme points of the curve to garantee the adjacency, if desired. A tolerance represented by a cursor cube also controled by the user, determines the area of atraction. Figure 2 shows the screen of the modeler during the creation of a Bspline curve.

All the curves are internally stored with their parametric descriptions, which involves the points given by the user and the type of the curve. In addition to the parametric description, an adaptative polygonal representation of the curve is evaluated, as described by de Figueiredo [7], and also stored for further drawing, picking, subdivision, and intersection algorithms. The adaptative polygonal representation is common for all types of curves and these algorithms have a single implementation. The gaps generated by the use of the polygonal representation in the subdivision and intersection algorithms are controled by different tolerances used in “screen space” and in “object space”.

Surfaces

The creation of surface patches is done with existing curves. The user chooses a method (the implemented ones are: translational sweep, rotational sweep, generic sweep, parametric four-sided, Bezier three-sided, Delaunay triangulation, and transition mapping according to some predefined parametric pattern), pre-selects the curves that completely define the surface mapping, and creates the surface patch by pressing the corresponding button. To exemplify this process, Figure 3 shows the generation of a quarter toric shell by the generic sweep method. Figure 3 (a) shows the two arcs created for this purpose. The first selected curve defines the shape to be swept. The second denotes the path, the subdivisions, and the position of the cross sections. Figure 3 (b) shows the generated surface patch.

Similarly to curves, the parametric and adaptative polygonalization representations are stored for surfaces. The implementations presented in [4] were used to accomplish the interpolations necessary to build the parametric representation. The polygonalization of surfaces is obtainned according to the algorithm described by Vlassopoulos [8]. To perform the intersection bewteen two patches of surfaces the polygonalization representation is used to generate trimming curves. The meshes are reorganized further to interpolate the points that define the subdivision of the trimming curves. The surfaces are not reparametrized because this would cause distortions in the original modeling. Figure 4 (a) shows a visual intersections between surfaces. Figure 4 (b) shows the trimming curves generated with the polygonalization representation.



(a) arcs. (b) gsweep.

Figure 3: Generic sweep built with two arcs.

(a) surface patches. (b) trimming curves.

Figure 4: The generation of the trimming curves.

Solids

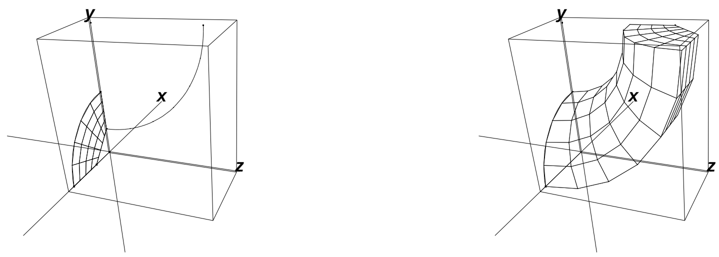
The solid methods implemented in the modeler correspond to sweep methods, combining surface patches with curves. To exemplify this, Figure 5 (a) shows a curved model defined by a four-sided spline surface mesh swept along an arc with uniform subdivision in Part (a). In Figure 5 (b) the boundary of the resulting solid mesh is shown.

ATTRIBUTES

The present modeler is linked to the extensible system for attribute customization presented by Carvalho and Wawrzynek [9]. This system provides a high level interface to assign simulation attributes to the geometric and discretization entities. The modeler and the attribute library are completely independently an the interface only deals with generic references (implemented in C with void pointers) and dimensional descriptors. At each attribute attachment, the library asks for lists of selected entities.

EXAMPLES

Two complete models generated by the program are presented. The first, shown in Figure 6, consists of a boundary element model of a semi-submersible oil platform used for ship stability analysis. The second model represents a mechanical part for finite element analysis (Figure 7).



(a) surface patch and arc. (b) solid mesh.

Figure 5: Solid sweep mesh built with a surface pacth and an arc.

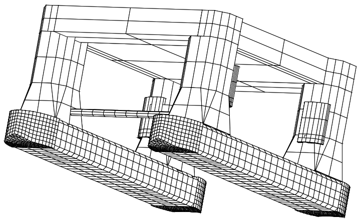


Figure 6: Boundary element mesh of semi-submersible PXVIII, Copyright ©Petrobrás.

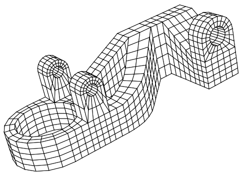


Figure 7: Finite element mesh of mechanical part.

CONCLUSION

The modeler has power of expression as we can see by the examples. The results presented in [5] show that the interface has been remodeled successfully. The order of the intersection algorithm is quadratic with respect to the number of facets of the adaptative representation. We are about to add topological informations to the data structure in order to enhance the time consuming.

REFERENCES

[1] B. Castier, L.F. Martha and M. Gattass, *A Taxonomy for Interactive Manipulation and Viewing of 3D Objects* (in portuguese), Proceedings of the VII SIBGRAPI[1], 1994, 149-156.

[2] L.C.G. Coelho, L.F. Martha, C.S. de Souza and M. Gattass, *Mesh Generation in Space with Direct Manipulation and Object Orientation* (in portuguese), Proceedings of the XIV CILAMCE, 1993, 1275-1284.

[3] M.J.G.M. van Emmerik, *A Direct Manipulation Technique for Specifying 3D Object Transformations with a 2D Input Device*, Computer Graphics Forum, 1990, vol 9, 355-362.

[4] L.F. Martha, *Topological and Geometrical Modeling Approach to Numerical Discretization and Arbitrary Fracture Simulations in Three-Dimensions*, Phd Thesis at Cornell University, Ithaca, N.Y., 1989.

[5] L.C.G. Coelho and C.S. de Souza, *Comunication of Problems and Geometric Solutions in a 3D Interface* (in portuguese), Proceedings of the VIII SIBGRAPI, 1995, 183-190.

[6] M.M. Carneiro, *Interact: An Interaction Model for Graphic Editors* (in portuguese), Msc Thesis, Catholics University of Rio de Janeiro, Informatics Department, 1995.

[7] L.H. de Figueiredo, *Adaptative sampling of parametric curves*, Graphics Gems V, 1995, 173-178.

[8] V. Vlassopoulos, *Adaptative polygonalization of parametric surfaces*, The Visual Computer, 1990, 291-298.

[9] M.T.M. de Carvalho and P.A. Wawrzynek, *An architecture for configurating geometric modelers: computational mechanics application* (in portuguese), XXVII SEMISH, Latin-American Informatics Conference, Canela RS, Brazil, 1995.

Adress for Correspondence

Luiz Fernando Martha, Departamento de Engenharia Civil, Pontifícia Universidade Católica do Rio de Janeiro, Rua Marquês de São Vicente, 225, Gávea, 22453-900, Rio de Janeiro, RJ, Brazil.
Email: `lula@icad.puc-rio.br`

[1]The Brasilian Computer Graphics Congress.