A BREP Model and Mesh Errors Detecting Tool: TopoVisu

François Petit, François Guibault

A BREP model and mesh errors detecting tool: TopoVisu

François Petit and François Guibault

École Polytechnique de Montréal, Génie informatique,
2500, chemin de Polytechnique, Montréal (Québec), H3T 1J4 , Canada
{francois.petit,francois.guibault}@polymtl.ca
http://www.polymtl.ca/

Abstract. This paper aims to describe the methodology behind the computer aided design tool TopoVisu. The objective of this tool is to support analysts and designers in their work, mainly by helping detect errors in models and meshes. But the error detection is not the only feature of TopoVisu, it also provides a visualization environment to cope with data portability issues. The paper briefly describes the type of problems that may be detected through the tests implemented in TopoVisu, and the PDF 3D exporting tool implemented as a mean of documentation and communication of the models and their validation.

Keywords: Computer aided design, Boundary representation, meshes, errors detection, errors documentation.

1 Introduction

Computer aided engineering and design involve processes that can be decomposed in several steps. First comes the system analysis, then the preliminary design, during which a boundary representation (BREP) model is created for analysis and optimization purposes. In order to carry out analysis, several models with various levels of fidelity may be used. For high fidelity analysis, the geometric models must be discretized, meshes are therefore generated, and simulations are run on the meshes. According to the simulation results, optimizations can later be carried out on the BREP model to obtain the final numerical model that will be used in the subsequent phases of detailed design.

Along those steps, some errors can be generated, either by designer mistakes or by conversion among the software products used in the design chain. The less demanding way to correct these errors would be to correct them when they are made, or very soon afterwards. Otherwise, an error in design could sometimes be corrected only after several simulations. And the later an error is found, the harder (and costlier) it is to correct it. That is why there is a need for active detection of problems, mainly during the preliminary design phase.

This paper presents a software system that aims to help designers detect CAD and mesh generation errors early. Its main purpose is to propose a visualization environment for BREP models and meshes, to detect and highlight errors and

to allow documenting these errors and save the diagnostics in order to ease communication.

Among various verifications that are performed, the software can, for instance, measure gaps between coinciding topological entities, volumes of mesh cells, size of edges, etc. and compares them to tolerances in order to determine if a given geometric model is valid and watertight or if a mesh is of sufficient quality to allow simulation. When errors are detected, geometric or mesh entities are highlighted (by coloration and listing) in the graphic interactive interface. It's also possible to label each entity if a problem appears. Finally, the user can export his model, with its visualization parameters, to an Adobe PDF 3D file, which can be read easily on any computer with Adobe Reader installed.

The paper first discusses the verification processes that are carried out at three levels: 1) geometric, 2) topological, and 3) meshing, along with example problems for each. The paper also discusses visualization strategies used to help designers understand the problems detected, and how the visual cues may be saved and manipulated in the application-neutral format, PDF 3D.

The resulting software can be considered as a BREP model and mesh debugger, and is able to export its data to an open file format, which can ease analysis and communication during computer aided engineering.

2 Problem review

According to Matsuki [9], problems in CAD systems can be classified in two categories, quality of models and durability of data. The first category regroups problems at different stages of the model. Some are present in the BREP model, others have to be solved on the mesh.

2.1 Issues concerning the quality of models

BREP model issues. The BREP model issues can be summarized in one word: water-tightness. As the main principle for BREP models is to define an entity through its boundaries, it is necessary to have closed surfaces to delimit the volumes. Most of the problems consist in two entities that do not coincide while they should. For example, two connected surfaces can be slightly different at their common edge (see Fig. 1). In this case, the shell defining the volume of the turbine will be closed through the creation of edges on each surface, and declaring them as corresponding.

With this example problem, as with every manifold BREP model, it is possible to create a surface joining the existing borders, but such an approach increases the computing time and model complexity drastically. What's more, models used in TopoVisu are based on the STEP [7] concept of BREP models, so they can contain non-manifold entities (eg. internal faces in volumes), and thus attempting to connect several surfaces along a given edge worsens the problem. For these reasons, other solutions are used, which may also be applied

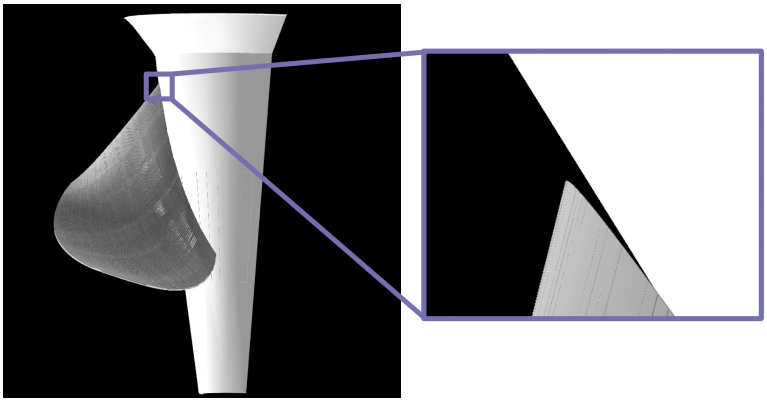

Fig. 1. Example of water-tightness problem: there is a small gap between the turbine blade and the hub.

in the more restrictive domain of manifold models. The existing solutions that inspired the development of TopoVisu follow.

It can be relatively simple to add information to the geometrical information. For example, most BREP models contain topological information. Entities (Vertices, edges, faces, ...) are used to mark up the corresponding geometrical objects. These entities are the core of the topology defined in the STEP [7] standard. This allows to verify the theoretical water-tightness, but if the surfaces do not match where they should, the quality of the mesh generated afterwards will be impacted.

To ease the adjustment of borders, it's possible to use a simplified model, such as the one proposed by [5]. In this model, the entities of lower dimensions (curves, points for 3D) are only defined as sub-objects (and one scalar equation) of higher dimension objects. This allows the suppression of edges and vertices by themselves but, as a drawback, it's not possible to access an explicit expression for these entities. So there can again be a loss in computing speed.

Segal [11] proposed to use tolerances to detect what entities should be merged. This solution, although useful in most cases, is not flawless. Geometric models of industrial complexity often include regions or parts which are too narrow to be united/separated correctly using only tolerances (see Fig. 2 for an example).

The approach proposed in complete topological models, such as those proposed in STEP [7], by Tanaka and Kishinami [14], and TopoVisu, is to add the topological information to the geometry, and verify model coherency through tests based on tolerances. That way, one can know beforehand what entities are connected, and the tolerances are only used to ensure that the geometric reality is consistent with the topological information.

This solution allows to have a BREP model of quality before mesh generation. But the mesh can still have problems due, for example, to a narrow geometry. To ensure that particularities of a geometry does not result in a simulation of poor quality, a mesh checking step is generally added before simulations are run.

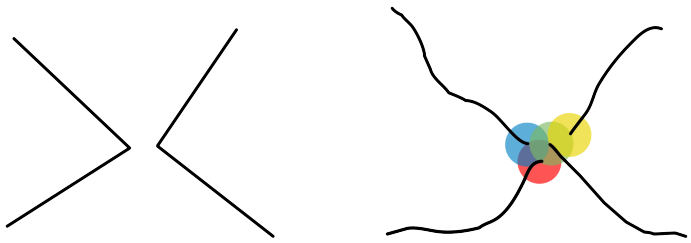

Fig. 2. Ideal case (on the left) and ambiguous case described with tolerances of a unique topological situation.

Meshing issues. Concerning the mesh generation, different strategies can be used to avoid errors. One approach is to have a BREP model as precise and with as much information as possible. Some examples of this strategy follow.

Gopalsamy, Ross and Shih [5] chose to add information to the BREP model in the mesh generation as a XML file. This file contains information about critical areas of the model, as the meshing method to use on each area.

Steinbrenner, Wyman and Chawner [12] presents the interactive mesh generation process they developed for the JULIUS project. The main idea is to make the program ask for help or directions according to its needs for each product. The main problem with this method is that it is not compatible with batch-processing.

Another approach to mesh generation is to consider that the BREP model contain errors, and that the meshing algorithm will have to ignore them first. Chong, Senthil Kumar and Lee [4] decided to begin by meshing the incorrect BREP model, and to solve its problems on the resulting mesh. The difficulty with this approach is that they never have a correct BREP model, while correctness may later on be required, for instance for manufacturing purposes.

Busaryev, Dey and Levine [2] present an algorithm based on the merging of areas with small defects (gaps, overlaps, and intersections) through protecting balls, directly on the BREP model, and then to mesh the model using these protecting balls. But with this method, there is a risk to have similar problems to the approach using only tolerances to close a BREP model.

What differentiate the approach presented in this paper is that the BREP models are created and used in a unique, integrated format. This approach ensures that no error can be generated once the BREP model has been checked. Moreover, meshes need to be generated in batch mode, so no interactive method can be used. Thus meshes need to be checked afterwards, prior to using them in simulations (which is generally also a batch process). As verification of the mesh is not performed during its generation, the need for a checking tool arises : TopoVisu, a BREP model and mesh debugger, is needed.

2.2 Issues concerning portability and durability of data

Another issue is that even if a product is in a correct state, either a good quality mesh or BREP model, it can be deteriorated through communication. As the different steps of the design process do not use the same tools, software and file formats, errors can be generated, and information can be lost during conversions. That is why Sum, Koch, Nyen, Domazet and San [13] defined an archive framework adapted to the design tools used by their team. Their objective was to unite every version of a product while still being able to produce program-specific "views" of the product. This archive would handle the specificities of each file format.

Short-term errors are not the only ones designers have to face. Another issue is the conservation of product data. Regli, Kopena and Grauer [10] try to determine what data is necessary and sufficient to ensure that product data will be valid, and well interpreted in the future. The answer is far from trivial as it is not safe to assume that any software or file format will still be in use in 20 years (and 20 years is actually quite short-term when compared to the life span of some planes or buildings). Their conclusion is that it is really difficult to estimate what is sufficient, so it is easier to just keep everything possible. They also advise to keep records of product data in different forms, such as a native file format, a standard file format and a visualization file format.

Our approach is similar as our models can be saved as ".pie" (our native file format) and they can be exported as STEP files, and as PDF 3D files, which correspond to the standard and visualization aspects. To keep a good portability of the tool, the meshes are described in a CGNS-compatible file format.

3 Tests run in TopoVisu

The main objective for handling errors during the design phase is to detect the errors as soon as possible. TopoVisu detects those errors by running tests at various steps of the design phase. Some tests are run on the geometry and the topology, while others are run later, when the mesh is generated. Although both series of tests check different criteria, they are similar in their process. The idea is to evaluate each entity of the model or the mesh, and then to display the results (either as a list of faulty entities, or as statistics) to the user.

3.1 Geometry and topology testing

The quality of the initial model is really important for the validity of the mesh that will be generated from it. As long as the mathematical expression of the surfaces and curves are valid, the main problems of a BREP model come from the topology. That's why the tests on the model have to consider both the geometry and the topology.

The verification process is quite straightforward. The topological entities can be represented as a tree, an abstract root node is parent to all volumes, each

volume is parent to the shells that bound it, each shell is parent to faces and so on. Each entity has a list of the other entities it interacts with (for example, each co-edge has a list of other co-edges built on the same edge). Thus, to verify the model, the entity hierarchy is recursively traversed and tests are run on each entity to verify model properties based on neighborhood dependencies.

Table 1 presents the geometry and topology tests that are implemented in TopoVisu, they are classified according to the dimension of the geometric entities concerned. This table summarizes the description of the tests detailed in [3].

Table 1. Table presenting the tests run on geometry and topology in TopoVisu.

Dimension - Name	Test	Other entities involved	Description of the tests
0 - Vertex	Position	Co-vetices	Each vertex is compared to all the other vertices that should coincide with it (co-vertices).
	Position	Edges ending on the vertex	The extremity of each edge is compared to the position of the vertex
	Position	Edges ending on the vertex, co-vertices	The extremity of each edge is compared to the positions of all the co-vertices
1 - Edge	Length	Co-edges	The length of the corresponding edges are compared
	Coincidence	Co-edges	Coincidence among co-edges is checked: points along the edges are evaluated, and corresponding positions on each co-edge are compared
2 - Face	Coincidence	Edges	Verify that each edge delimiting the surface lies on that surface
	Loop closure	Edges	Verify that the edges delimiting the surface form a closed loop
	Loop closure	Co-edges	Verify that the co-edges form a closed loop

Fig. 3 shows two examples of detected errors. The problematic elements are colored in red and listed in the "Topological errors" box. The errors are assigned to the topological element of the lowest dimension containing all the entities implicated in the error.

3.2 Mesh testing

At the end of the mesh generation process, the validity of the mesh must be assessed. In a valid mesh, each edge connects two nodes, each node is connected by edges, there is no folding, no node outside the geometry, no hanging nodes, etc. What designers have to check is the quality of this mesh. Cells that are too

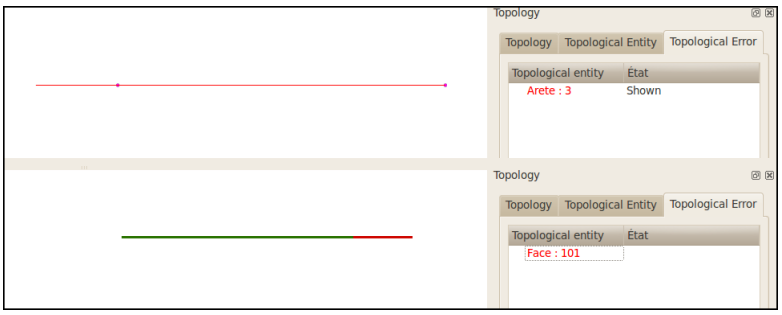

Fig. 3. Examples of topological errors: a vertex not corresponding to the end of an edge (top), and corresponding edge and co-edge with different lengths (bottom).

long, or angles that are too sharp may directly affect convergence during the subsequent simulation phases. To maximize the quality, each cell is evaluated through some of its geometric characteristics (which are calculated beforehand for each cell). Table 2 presents the mesh tests that are implemented in TopoVisu, for 2D and 3D cells. Each threshold mentioned is actually a pair of values which includes a warning and an error threshold.

Table 2. Table presenting the tests run on mesh in TopoVisu

Dimension	Entity concerned	Description of the test
2D	The length of each edge	Lengths are compared to low and high thresholds, and between each other, to avoid sudden change of scale.
	The area of each cell	Areas are compared to low and high thresholds, and between each other, to avoid sudden change of scale.
	The aspect ratio of each cell	Aspect ratio is compared to low and high thresholds, to limit error in the following simulation.
3D	The length of each edge	Lengths are compared to low and high thresholds, and between each other, to avoid sudden change of scale.
	The volume of each cell	Areas are compared to low and high thresholds, and between each other, to avoid sudden change of scale.
	The aspect ratio of each cell	Aspect ratio is compared to low and high thresholds, to limit error in the following simulation.

The cells that do not pass the tests are tagged as containing errors, and are colored in red. In addition, some thresholds (such as volume, angle, etc.) can be adjusted by the user to match stricter requirements from clients. In that case, more cells are highlighted, as shown in Fig. 4. At the end of the series of tests, a report is generated. This report notifies the user of the number of cells in the warning and error ranges, as well as the positions of the cells having extreme values for each criterion.

4 GUI Design

The main purpose of the proposed debugger is to support the designers and analysts in their work. A tool is really of help when it is instinctive in its use. That's why one of the first steps of development was to create a graphics environment to support the visualization of the models.

The core of the application is based on Qt, a multi-platform object-oriented framework. This framework proposes a widget-based development, which allows adding new features as the functionality of the tool needs to evolve. As Qt is a widespread framework, it provides a familiar look to the user and minimizes portability issues. Qt's familiar look was maintained in the creation of menus and sub-windows as can be seen in Fig. 4.

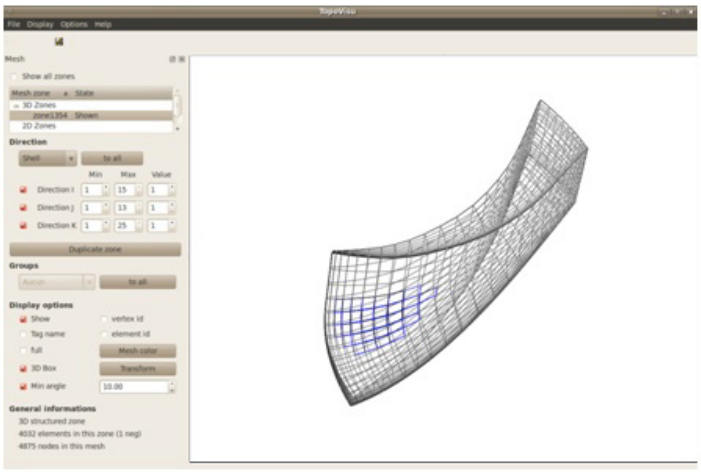

Fig. 4. Screenshot of the TopoVisu interface, with angle threshold activated. Cells with too sharp angles are colored in blue.

For the visualization of the three dimensional model, the widely used cross-platform OpenGL API was adopted. The philosophy of TopoVisu is to separate visualization from actual mathematical data. To achieve this, independent visual objects are constructed, which are then linked to the mathematical objects used in the BREP representation. Two symmetric structures are obtained, one containing the visualization information, and the other containing the BREP model. Each item of each list can be linked to its corresponding item in the other list.

Complex models can be hard to fathom as a whole. TopoVisu provides some ways to help its users have a better visual on BREP models and meshes. First, each graphical entity has its own color. Although initial colors are based on the type of entity they are attributed to (faces are blue, edges are green, etc.), it is possible to change the color of each element separately. This is particularly

useful in order to focus on a particular area. It's also possible to hide parts of the model if it is of no use for the user's task. Calculating the bounding boxes of the visible elements allows a precise focus of the camera.

Concerning meshes, several display modes are available. Users can choose to only display the frame of the meshed volume, or to add the cells on the external shell, or to display the whole mesh. As mentioned in 3.2, cells can be colored according to certain characteristics.

5 Export to PDF3D

As explained in [9] and [10], durability of CAD data is an unsolved issue. Nowadays, most legally accepted archives are on paper, and according to [8], some companies are attempting to move away from this strictly paper-based approach. This orientation isn't surprising given that paper plans are not used anymore during the design process.

The main problem in that area is to ensure the longevity of the data. One solution is to use a non application-specific format that is already used on a large scale. With such a format, we can be sure that support will exist for a long time, and that's why we decided to export our own data in the same format: the Adobe PDF file format.

In this file format, it is possible to insert 3D data in two types of formats: U3D and PRC. While the first format aims for three dimensional graphical scenes (so it contains geometry in an approximated form), the second can store geometry and topology in their exact mathematical form. That's why we chose the PRC embedded in PDF as a favorable transmission format, the embedding allowing any user with the freeware Adobe Reader to open the file and see the geometric model. It's also possible to manipulate the model in three dimensions in order to see every angle.

To create our PDF3D - PRC exporter, we decided to use an existing C/C++ component. This component was extracted from the open source software Asymptote [6]. This library only allows the creation of geometry (the visualization aspect of TopoVisu), the topology management still had to be implemented. It was developed according to the PRC file format specifications [1].

The exporting tool offers the possibility to save all or part of the geometry of the current open project. It saves all items with their current color to allow the highlight of a specific part. This can be particularly useful for a good visualization in a report for example.

6 Conclusion

The development of TopoVisu is still in progress and is probably to stay that way, in order to keep up with all the changes that come in the design industry. But the debugger is usable, and used, right now. The error detection tools are really helpful to designers. It would be nearly impossible to check models entirely by hand.

One advantage of our method is that the product has a good quality at both BREP model and mesh steps. This allows to reuse the model if meshing technology evolve drastically in the future, as opposed to the methods that consider the model to be faulty. Another strong point of TopoVisu is its strong visualization environment. The software is interactive, easy to use, and easy to learn. The results of the tests can be seen, and that is very helpful.

The software does not only help to design new products, it also allows to document and communicate easily on the designs. All the file formats we use are easily convertible to standards (STEP for BREP models, CGNS for meshes and PDF 3D - PRC for reports), as they follow the same methodology. Our objective, in the long run, is to be able to create easily shareable archives of BREP models and meshes, with highlighted errors and points of interest.

References

1. Adobe Systems Incorporated: PRC format specifications (Jan 2012), `http://livedocs.adobe.com/acrobat_sdk/9/Acrobat9_HTMLHelp/API_References/PRCReference/PRC_Format_Specification`
2. Busaryev, O., Dey, T., Levine, J.: Repairing and meshing imperfect shapes with delaunay refinement. In: SPM. vol. 9, pp. 25–33 (2009)
3. Camarero, R., Courchesne, O., Guibault, F.: Hybrid mesh validation and visualization. Proceedings of Numerical geometry, grid generation and scientific computing
4. Chong, C., Senthil Kumar, A., Lee, H.: Automatic mesh-healing technique for model repair and finite element model generation. Finite Elements in Analysis and Design 43(15), 1109–1119 (2007)
5. Gopalsamy, S., Ross, D., Shih, A.: API for grid generation over topological models. The 13th International Meshing Roundtable, Williamsburg, Virginia, USA pp. 221–230 (2004)
6. Hammerlindl, A., Bowman, J., Prince, T.: Asymptote sources website (Jan 2012), `http://asymptote.sourceforge.net`
7. ISO-10303: Industrial automation systems and integration – product data representation and exchange (1994), `http://www.iso.org/iso/home.html`
8. Kheddouci, F.: L'archivage à long terme de la maquette numérique 3D annotée. Master's thesis, École de technologie supérieure (2010)
9. Matsuki, N.: Problems in current CAD systems. International Journal of Product Lifecycle Management 4(4), 326–330 (2010)
10. Regli, W., Kopena, J., Grauer, M.: On the long-term retention of geometry-centric digital engineering artifacts. Computer-Aided Design 43(7), 820–837 (2011)
11. Segal, M.: Using tolerances to guarantee valid polyhedral modeling results. ACM SIGGRAPH Computer Graphics 24(4), 105–114 (1990)
12. Steinbrenner, J., Wyman, N., Chawner, J.: Fast surface meshing on imperfect CAD models. In: 9th International Meshing Roundtable. pp. 33–42. Citeseer (2000)
13. Sum, S., Koch, D., Nyen, C., Domazet, D., San, L.: Development of a framework system for tool integration in a product information archive. Computers in industry 30(3), 225–232 (1996)
14. Tanaka, F., Kishinami, T.: Step-based quality diagnosis of shape data of product models for collaborative e-engineering. Computers in Industry 57(3), 245–260 (2006)