Automatic and Parametric Mesh Generation Approach

Alan M. Shih, Sankarappan Gopalsamy, Yasushi Ito, Douglas H. Ross, Mark Dillavou, and Bharat K. Soni

Department of Mechanical Engineering
University of Alabama at Birmingham (UAB)
Birmingham, AL 35294, U.S.A.
E-mail: {ashih, sgopals, yito, dhross, dillavou, bsoni}@uab.edu

Abstract - In this paper, we will discuss the approaches and technologies associated with the development of the Geometry-Grid Toolkit (GGTK) and MiniCAD. GGTK is a software library that provides core functionalities in geometry, topology and mesh generation and MiniCAD is a software system with graphical user interface built on top of GGTK. The application of GGTK/MiniCAD to the development of customized packages for automated analysis and parametric design involving complex geometry is demonstrated. The progress realized in the development of automatic and parametric mesh generation technologies for high fidelity simulations is presented. Examples are presented to demonstrate the benefits and effectiveness of this library. The future vision and development plans for this library are also outlined.

Keywords: NURBS, facetted surfaces, topology, parametric modeling, geometry and grid template.

I. INTRODUCTION

Geometric modeling and mesh (grid) generation are important preprocessing steps in many engineering simulations, such as Computational Fluid Dynamics (CFD) and Computational Solid Mechanics (CSM). In many cases, the objects in such simulations are designed and modeled using Computer-Aided Design (CAD) systems in terms of smooth curves and surfaces based on some design parameters that determine the shape and size of the objects. The CAD models are then passed to grid generators in order to obtain surface and volume grids needed for CFD and CSM computations [FAR, 99]. These simulation methods are being used to provide optimized designs without resorting to creating physical instantiations of a product. In order to create optimized designs, the parameters must be varied and the results studied to steer further optimizations. The separation of CAD and grid software requires that multiple CAD models with a modified geometry be built and then imported into the grid software for meshing. This process of modeling and then recreating the grid over essentially the same model is time consuming and wasteful. This paper does not address how to find the optimal parameters but rather presents a method of varying these parameters in an efficient manner to create grids suitable for the simulation methods. We propose a system that has the ability to vary geometric or grid parameters and rebuild the grid without having to modify the geometry in some external software. The system also allows the user to easily create a template with the parameters to define relationships and easily modify their values. We are not aware of any unified geometric modeling and grid generation system that supports such parametric modeling and grid generation.

Secondly, there is increased interest and necessity in dealing with objects from bio-medical applications, such as human lung and brain. The geometric models for such objects are obtained from scanned data and they are generally modeled as faceted surfaces consisting of large collections of triangles. Generating grids over these objects that are suitable for CFD or CSM analysis is a relatively new area and is not supported by present day CAD and grid generation systems.

This paper presents a framework for developing geometry and grid templates that supports traditional CAD geometry as well as geometry represented by faceted surfaces. This framework is being developed at the Enabling Technology Laboratory (ETLab), University of Alabama at Birmingham (UAB). The framework is based on the Geometry and Grid Toolkit (GGTK), and MiniCAD – a unified CAD and grid generation system built on top of GGTK.

Figure 1 shows the architecture of the framework. The various components of the framework are explained in the following sections. Section II describes the geometry and grid generation tools for models defined by smooth
curves and surfaces using Non-Uniform Rational B-Splines (NURBS). It talks about a topological representation scheme to represent watertight geometry needed by grid generators. Then it gives an outline of various functions for generating grids over topological entities. Section III talks about grid generation over models defined by faceted surfaces. It also gives an outline of our unstructured volume mesh generation methods. Section IV explains the graphical user interface (GUI) and functionalities provided in MiniCAD that are needed for template development. Section V explains the issues in developing the template framework. Finally section VI gives conclusions and future directions for template development.

II. MODELS DEFINED BY SMOOTH SURFACES

GGTK has functions to (a) create and edit geometric models in terms of NURBS curves, surfaces and volumes, (b) define a topological representation of the model in terms of vertices, edges, faces, and volumes, and (c) specify and generate required grids over the topological entities. These functionalities are put under three components – geometry, topology, and grid, which are explained briefly below. More details can be found in [GOP, 03], [GOP, 04] and the GGTK user manual [GGT, 03].

A. Geometry Component

The main geometry entities are 3-D point and NURBS curves, surfaces, and volumes. While the 3-D point is given a simple structure storing the three coordinate values, the data structures of NURBS entities contain NURBS data – order (degree+1), knot vectors, and control points – as per standard definitions [PIE, 97]. The main functions contained in the geometry component are given below.

**NURBS functions:** This module contains almost all standard functions [PIE, 97] on NURBS curves, surfaces, and volumes such as point evaluation, computation of derivatives, split, join, trim, transformations, knot insertion, degree elevation, and reduction of control points. Apart from these, it has many methods for curves and surface creation such as conics, surface of revolution, interpolation, lofting, blending, and transfinite interpolation.

**Basic geometry functions:** This module contains basic utility functions on 3-D points and vectors, and transformation functions.

**Advanced geometry functions:** The geometry module contains many advanced functions such as (a) projection of point and curve onto a NURBS surface and (b) surface-surface intersection. It also has limited capability of reading IGES and STEP entities and converting them to GGTK entities.

B. Topology Component

In this component, data-structures and functions are given for the topological representation of a geometrical model in terms of vertices, edges, faces, and volumes. Topological representation is used to represent and work with "water-tight" geometry, i.e., geometry without "holes". The topology data-structures and functions have been designed based on the new geometry representation scheme proposed by Chew *et al* [CHE, 02]. The main objective of this scheme is to represent geometry suitable for grid generation for CFD and fracture mechanics applications. This scheme is based on boundary representation, where a 3-D volume is described by its bounding faces. A boundary face is defined by an underlying surface and bounding edges. A boundary edge is defined by an underlying curve and two bounding vertices. Since a boundary edge should lie on the surface, it is required, in this scheme, that the underlying curve be defined in the parametric domain of the face.

**Topology Functions:** The topology component has topology creation functions such as "add trim NURBS surface", which will automatically create a topological face, its boundary edges and vertices. It has functions to check to see if a model is watertight or not. Then it has Application Programmer Interface (API) functions [GOP, 04] needed by grid generators to query and evaluate the underlying geometry through the topological entities.

C. Grid Component

The main entities of the grid component are edge grid, face grid, and volume grid. These are linked to the corresponding topological entities. The face grid and
volume grid will have additional classifications into structured, unstructured, and generalized grids. Through their links with the topological entities, the grid entities are linked to their underlying geometric entities. Hence in addition to the 3-D coordinates of the grid points, the grid entities have the capability to store their parametric values with respect to their underlying geometries. This may be used in grid adaptation.

**Grid generation tools:** GGTK contains the following sets of functions for grid generation: (a) packing functions to discretize a curve and generate an edge grid, (b) structured face and volume grid generation by elliptic, transfinite interpolation and reparametrization methods, and (c) unstructured face grid generation by an advancing front method. It has also functions for computing grid quality and grid interpolation.

**Grid specification:** GGTK has support for specifying various types of grids using Extended Markup Language (XML) as explained in [GOP, 04] and generating the specified grid over a topological model. Figure 2 shows structured volume grids over the six blocks of a shear coaxial element injector. The grids were specified using XML and automatically generated by GGTK.

### III. MODELS DEFINED BY FACETTED SURFACES

A NURBS representation of a surface gives flexibility for modifying. However, there are many cases when a facetted surface is given as input geometry, which cannot be easily converted to NURBS. In addition, most of commercial CAD software systems support output as facetted surfaces in stereolithography (STL) format, which is the de facto standard in rapid prototyping. Mesh generation based on a facetted surface representation is another powerful and straightforward approach. GGTK has two approaches to obtain triangular meshes suitable for numerical simulations from facetted surfaces. One is a direct advancing front method [ITO, 02], and the other is a mesh decimation/refining method coupled with mesh quality enhancement methods, such as node smoothing and edge swapping based on the Delaunay property [ITO, 05].

#### A. Direct Advancing Front Method

The direct advancing front method gives controllability for local mesh density, which is often important for numerical simulations. First, degenerate facets are removed by swapping edges and removing too small edges. They are sometimes found in STL files, and prevent correct estimation of surface values, such as normals. This modified facetted surface is used as a background mesh, and a new surface mesh is created on it (Figure 3a). Second, geometrical features are extracted based on a folding angle at each edge (Figure 3b). A user can change parameters to specify desired geometrical features, and add new ones using GUI. Third, the user specifies node distributions on the geometrical features, which form an initial front for the direct advancing front method. Forth, surface triangulation is performed directly in 3-D to evaluate the quality of triangles easily (Figure 3c). Node locations are calculated using a triangular shape function.

![Figure 3](image-url)

**Figure 3.** Mesh generation for a mechanical object: (a) input as a facetted surface; (b) geometrical features, most of which are based on folding angles and some of which are added manually using the GUI environment; (c) surface mesh generated using the direct advancing front method; (d) a cross-section of a tetrahedral mesh

#### B. Mesh Decimation/Refining Method

This approach targets especially surfaces extracted from medical images, for example, computed tomography (CT) and magnetic resonance imaging (MRI) data. Surface meshes suitable for numerical simulations are generated semi-automatically from facetted surfaces. Medical images can be processed using open source libraries, such as the Insight Segmentation and Registration Toolkit (ITK) [ITK] and Visualization ToolKit (VTK) [VTK], and facetted surfaces are obtained using the Marching Cubes method [LOR, 87]. They contain many skewed facets, and the number of facets depends on the resolution of original medical dataset (Figure 4a). In addition, geometrical features are often obscured and they cannot be properly defined only based on a folding angle at each edge.

In this approach, a uniform mesh is created first through adding nodes to the original triangulated surface, smoothing nodes and swapping edges based on the Delaunay property to avoid truncation errors in the latter
Mesh decimation is then performed based on local surface curvature and volume thickness to generate high-quality volume meshes (Figure 4c, d).

Figure 4. Surface mesh generation for a human lung: (a) original surface (352,904 triangles): the bronchial tube is manually removed and the resulting hole is closed using a small number of stretched triangles; (b) a uniform mesh; (c) after decimation #1 (26,702 triangles); (d) after decimation #2 (11,222 triangles)

C. Unstructured Volume Meshing

From a closed triangulated surface mesh, tetrahedral meshing can be performed using an advancing front method [ITO, 04] (Figure 3d). The quality of resulting meshes is enhanced using angle-based node smoothing and face swapping based on the Delaunay property. To perform viscous flow simulations, hybrid mesh generation is available using an advancing layer method (Figure 5).

IV. MINICAD

MiniCAD is a unified system for geometric modeling and grid generation, built on top of GGTK. It provides GUI environment for model creation, editing, topological representation, and grid generation. It has features for parametric modeling and template development. These are briefly explained below.

A. GUI Environment

The standard layout for MiniCAD, shown in Figure 6, has a typical menu bar, a hierarchical object browser, a 3-D view port and creation tabs for geometry, topology and grid. The menu bar includes File, Edit, View, Window and Help drop downs with the common methods implemented. The 3-D view port can be split into multiple view ports and supports orthographic or perspective projection.

The object browser supports expand and collapse of the hierarchy and the modification of the object name. Objects can also have a window opened to modify their display parameters (such as color, wire-frame) or a window to show information (such as number of control points or grid quality) another window allows for setting of transformation values. Translation and scaling can also be adjusted interactively. Methods for input to create or modify objects include numeric entry, selection of objects in the 3-D view or browser and a command line.

Selection of objects for use in creation of other objects can be done either by adding them to text boxes in the creation tabs or through an interactive creation process where the user is prompted for the required input or selection. Selection in the 3-D window supports crossing and surround methods. The selection list can be modified based on the mouse button used.

B. Model Creation and Grid Generation

The GUI supports 2-D and 3-D geometry surface creation. Volumes are defined using boundary representation. Grids are built by passing the geometry through an explicit topology structure to ensure that a watertight grid is created.

Geometry: Geometry objects include points, curves, surfaces and volumes. Curve types are line, circle, ellipse, Bezier, NURBS by control point or interpolation, parametric curve, parametric line and surface/surface intersection. Modification of curves includes reverse, split and join. Geometric surface creation includes bilinear,
rotated curve, extruded, transfinite interpolated, lofted, interpolated. Modification of surfaces includes reverse, split, join and extend. Volume types are extruded and revolved.

**Topology**: Topology objects include vertices, edges, faces and volumes. However, only volumes and faces are created interactively. The required edges and vertices are created automatically. The volumes are created empty or by passing a geometry volume. If an empty topology volume is created faces can be added using complete surface, trimmed surfaces or planar curves. The topological volume is displayed in the browser in a hierarchical manner showing volumes, faces, edges and vertices at their appropriate level. Topology volumes can be checked for watertightness. Interface for methods to manually create a watertight volume by manipulating tolerances are also available.

**Grid**: Grid objects include edges, faces, volumes and models. A grid model must be created to hold other grid objects. Grid edges are created on topology edges. Grid faces are created on topology faces. Grid volumes are created on topology volumes. Grid edge number and packing parameter changes will immediately change the grid edge and display. An update button press is required to have the grid face or volume match the grid edge change. All methods in GGTK for structured and unstructured face and volume grid creation are supported.

### C. Support for Parametric Modeling

Parametric modeling for CAD packages takes two forms, *history based* and the ability to have *inter-relations between parameters* which can be interpreted as constraints [MAR, 96]. MiniCAD supports both of these forms. The history based form is divided into two methods. The first method is to remember the creation values for an object, which works for objects created inside the MiniCAD environment. The second method is to remember the creation relationships between objects, which we define as upstream-downstream relationships. The other form is through relationships between object parameters specified by equations associated to object parameters.

**History using creation values** and allowing their modification is based on a class structure which has each object type storing the parameters that define it (for example, a circular arc stores radius, start angle and end angle). This type of information will typically not be available if the geometry comes from an external source (such as an IGES file). We are proposing methods to deal with this in future work.

**Object relational history** is instantiated through upstream – downstream relationships. For example the circle has an additional component to store, which is a pointer to the point that defines its center. Upstream geometry is used to create an object. Downstream geometry is an object created using this object. Any change to an upstream geometry will rebuild downstream geometry (e.g., the diameter of a cylinder based on a circle increases if the circle diameter is increased). These relationships can be removed either globally or per object by removing history.

If the system stopped at the geometry level then we could just store the constructor creation information for updates. Because we want to maintain links to the topology and then the grid that is built on the topology it is sometimes necessary to store the completed geometry with a more complicated relationship than for a purely geometric modeling system. For example, if we only maintain the curves and the extrude operation to create a volume most of the geometry that should maintain a link to the downstream topology and grid would be rebuilt and loose its connectivity. However, this does also currently limits us to only supporting geometric changes that do not alter the topology.

Topology has an upstream-downstream relationship with the geometry. A topological object may have multiple geometric instances. For example, if an edge is the common boundary of two faces, then the edge will have two instances of geometry curves – one with respect to each face. In such cases, a change to either geometric instance will require rechecking of the topology.

The upstream-downstream relationship between topology and grid objects is currently simple because they are one to one. Updating the topology due to an underlying geometry change (assuming no change in topological structure) will pass the same geometric information to the grid.

**The second form of parametric modeling** is the inter-relationships between parameters. In MiniCAD, this comes from the ability of parameters to hold and execute an equation. The equation associated to a parameter is used to define the parameter value through a relationship to other parameters. These equations are executed in the python interpreter. The user has the ability to use the full functionality of the python language in these equations allowing any type on conceivable relationship to be maintained. Examples of parameters are point x, y or z coordinates, circle radius, number of grid points or grid packing parameters.
V. TEMPLATE DEVELOPMENT

Template development in MiniCAD is primarily through the selection of parameters and creating the relationships among them. Templates were originally created programmatically in MiniCAD as plug-ins, which required a developer's time and knowledge to create them. The new template system allows the end user to quickly create their own templates by dragging items to the templates panel and setting equations. MiniCAD also supports the ability to create a custom interface for a template that would only expose the GUI interface that the developer wanted.

The template framework leverages the power of the Python scripting language which gives the user a lot of flexibility when creating templates. Any user can create templates without any programming knowledge, but to harness the full power of templates, a basic understanding of the Python scripting language is required. Templates support full drag and drop along with auto-completion to help the user create templates quickly. As the user types equations, the auto-complete drops down a box of all the object's attributes that can be used as templates.

To create the new template, the user begins by creating or loading geometry into MiniCAD. The user can then select the geometry in the graphics window or browser and drag it to the template panel. The template panel will show all the object's available parameters as the user creates the template (Figure 7). Almost any mathematical equations can be modeled, and the user can use multiple equations to make up a single template parameter. Multiple parameters then make up the entire template.

Templates can be saved and loaded from a MiniCAD XML file which allows one user to create a basic template, save it, and give it to another user to use. The XML files are human readable, and all template parameters are clearly marked at the top, so the user can quickly change parameters in the file itself.

Templates in MiniCAD are extended using the Python scripting language. Python allows the use of powerful equations to express templates. It also gives the user flexibility such as the following. There could be a
parameter a user would want to change over a range of values and create a new grid at each interval. Python allows the user to do this quickly, and easily.

MiniCAD can also be extended with plug-ins. MiniCAD Plug-ins are currently divided into two groups; plug-ins that extend functionality and plug-ins that provide new workspaces. Plug-ins can either provide one of the two or both types. Creating a plug-in that extends MiniCAD functionality is done by creating a python script and loading it through MiniCAD. The python script can access any of MiniCAD functions and can also include its own functions.

Creating a custom environment: Commonly, a developer will want to leverage all the functionality of MiniCAD, but use it for a specialized purpose. The developer can do this by creating a new workspace plug-in for MiniCAD. MiniCAD has a default workspace which includes all the GUI items and the graphics window. MiniCAD’s GUI has a very modular design, and each GUI item can be used separately, so when a developer is creating a new workspace, they can include MiniCAD GUI items like the graphics window, or create their own. For example, a student was designing a new program for stent creation and wanted to leverage all of MiniCAD’s functionality, but didn’t want the complexity of MiniCAD’s interface. The student created a new workspace which included only MiniCAD’s graphics window, and a specialized panel for stent creation.

VI. CONCLUSION AND FUTURE DIRECTIONS

In this paper we have presented a unified geometric modeling and grid generation system that can be used to build geometry and grid templates for specific objects. We have outlined the basic geometric functions and grid generation tools needed for the system. We have also described the underlying graphical user interface that supports parametric modeling. We have explained the use of Python scripting language in the template development. We see that using this framework, we are able to build templates for objects like the shear coaxial element injector easily. As future work, we will use this framework to build templates for various other objects and add to the framework any new features that is needed.
Parametric modification of imported geometry: One challenge is the parametric modification of imported geometry. In the case of geometry built in MiniCAD the parameters are clearly available for modification using the first method of the history, however, in the case of imported geometry or meshes parameters related to construction of the original model are not present or cannot be easily extracted. We plan to create a method to control geometric transformations of the object that will have the properties of local influence, that it can modify a b-rep volume without edges separation and provide a clear geometric meaning.

Grid over multiple topological faces: Our current grid structure has a one to one correspondence to the topological structure. To give the user sufficient flexibility in the specification of a grid location it must be able to be defined either over a portion of a face or over portions of multiple faces.

Grid point modification: In some grids it is necessary to give the user fine control over the grid structure down to the grid point level. Grid point modification would allow the user to adjust a single grid point with either a 3-D or parametric location.

ACKNOWLEDGEMENTS

This research was supported in part by the following projects: (a) Improvements in Turbomachinery Grid Generation Tools, sponsored by the NASA Marshall Space Flight Center; (b) Automated and Parametric Dynamic Geometry-Meshing Tools and Technology for Thrust Chamber Assemblies, sponsored by NASA CUIP Program; (c) Mesh Improvement Algorithms and Software Technology, sponsored by DOD MSRC PET Project.

REFERENCES


