Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 163 (2016) 162 – 174

Procedia Engineering

www.elsevier.com/locate/procedia

25th International Meshing Roundtable

A Marching-Tetrahedra Algorithm for Feature-Preserving Meshing of Piecewise-Smooth Implicit Surfaces

Brigham Bagley[a], Shankar P. Sastry[a], Ross T. Whitaker[a]

[a]*Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT 84112*

Abstract

For visualization and finite element mesh generation, feature-preserving meshing of piecewise-smooth implicit surfaces has been a challenge since the marching cubes technique was introduced in the 1980s. Such tessellation-based techniques have been used with varying degrees of success for this purpose, but they have consistently failed to reproduce smooth curves of surface-surface intersection when two surfaces intersect at sharp angles. Such techniques attempt to discretize all surfaces within a given cell in a single pass by computing edge-surface points of intersection for each edge in the cell and use predefined stencils to generate the surface mesh elements. This approach limits the number of surface-edge intersections on every edge to just one (or some small finite number) because the number of stencils grows exponentially with the number of surfaces. In our tessellation-based approach, we discretize only one surface in each pass over the tetrahedral cells and retetrahedralize the affected cells for the next surface during the next pass. As a result, we manage to preserve sharp features in the domain, and our algorithm scales almost linearly with the number of surfaces. As in the isosurface-stuffing algorithm, we locally warp the initial tessellated domain to ensure that a high-quality surface mesh is generated.

Peer-review under responsibility of the organizing committee of IMR 25

Keywords: marching tetrahedra, feature preserving, surface meshing;

---

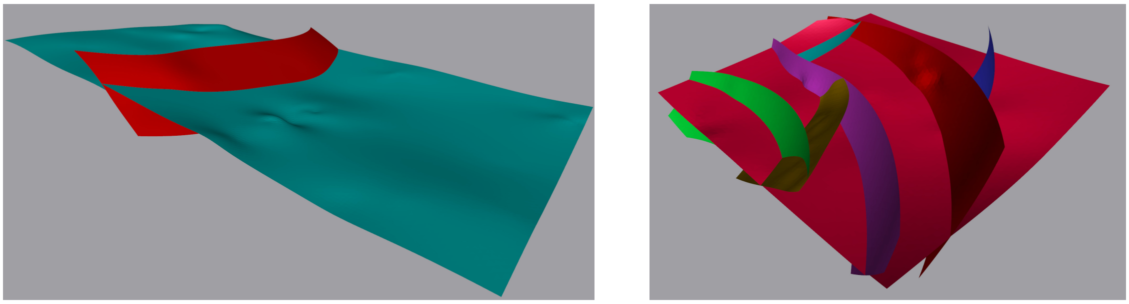
1. Introduction

The generation of surface meshes for implicitly defined surfaces has been studied over the last few decades [21] due to the applications of the meshes in areas such as biomedical simulations (from patient images) [9], constructive solid geometry (CSG) [5], and geometric modeling and visualization of complex real-world shapes (statues, for example) [17]. Our motivation to develop an algorithm for this purpose arose from our need for geometric modeling and simulation of subsurface geology. The primary challenge is to mesh our implicit functions quickly and preserve sharp features where two surfaces or more intersect. In subsurface geology, a horizon is a layer of soil parallel to the crust whose physical characteristics slightly differ from neighboring layers. There are discontinuities in horizons caused by massive tectonic forces during earthquakes or other geological events. Such meshes are necessary for analyzing

---

* Brigham Bagley eMail: brig@sci.utah.edu

Peer-review under responsibility of the organizing committee of IMR 25
doi:10.1016/j.proeng.2016.11.042



(a) A Simple network (b) A Complex Network

Fig. 1. Examples of typical domain seen in structural geology. We wish to capture the curve of intersection between two surfaces in our surface meshes. Existing techniques have been unable to do that when the intersection is "sharp".

water tables, extracting natural resources, analyzing impact of an earthquake, etc. Another example of such domains is layered, composite materials in semiconductor industries, where multiple materials are deposited layer by layer for manufacturing a computer chip. In this paper, we describe an algorithm to generate a surface mesh for geological structures using implicit functions whose level sets define the surface. The algorithm is generic enough to be used in other applications such as CSG.

The nature of the domains we wish to mesh is shown in Fig. 1. In the first example, we have one horizon split by a fault. The horizon surfaces on the either side of the fault are not continuous. In the second example, a complicated geological structure is shown. There are multiple faults dividing a single horizon into many different compartments, each having their own equations defining the surface.

Our technique builds upon prior tessellation-based algorithms such as the marching cubes [21], isosurface stuffing [19], and dual contouring techniques [17,20]. Such techniques build a surface mesh by first constructing a background mesh and then determining the edges though which the surfaces pass. These techniques preserve sharp features with difficulty because they restrict the number of surfaces passing through an edge to just one. As a result, some approximations are made to the geometry that result in elimination of sharp features. It is possible to extend these algorithms for preservation of sharp features by allowing multiple surfaces to cross an edge, but the combinatorial explosion in the number of possible configuration makes it very hard to implement them. Some details about these algorithms are provided in Section 2. In the techniques above, all the surfaces are resolved simultaneously in a single pass over the tetrahedral or the cubical elements in the background mesh.

Our technique is based on the isosurface-stuffing algorithm [19], which was developed for a single surface. We extend the technique for multiple surfaces, and show that it is possible to preserve sharp features in the domain at the cost of mesh quality. As in the isosurface-stuffing algorithm, we employ a tetrahedral warping strategy to retain higher quality. Section 3 provides a brief description of the isosurface stuffing algorithm.

Our algorithm overcomes the difficulty of preserving features by constructing each surface sequentially and subdividing each affected tetrahedron before constructing the mesh for the next surface. Although we make multiple passes over the set of tetrahedral elements, only a single surface is consolidated to the mesh, and we are able to preserve sharp features, in contrast to other techniques thus far. A detailed explanation of our algorithm is provided in Section 4. As expected from tessellation-based techniques (see [19]), our implementation is fast, and (depending on the material intersections) our meshes are of good quality (explained in detail in Section 5). In order to further improve the quality of the triangles in the surface mesh, we use a smart Laplacian-based smoothing technique to move the vertices around.

Our algorithm is defined for implicit surface functions and will need extensions for application in the biomedical field, where several scalar fields provide different materials in the domain rather than an implicit surface. We briefly discuss the possible changes in the algorithm and other future research directions in Section 6.