SAND2008-1883C

COMPARING TECHNIQUES FOR TETRAHEDRAL MESH GENERATION

M. A. S. Lizier
Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, Brazil
lizier@icmc.usp.br

J. F. Shepherd
Sandia National Laboratories - Albuquerque, NM - USA
jfsheph@sandia.gov

L. G. Nonato
Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, Brazil
gnonato@icmc.usp.br

J. L. D. Comba
Instituto de Informática - Universidade Federal do Rio Grande do Sul, Brazil
comba@inf.ufgrs.br

C. T. Silva
SCI Institute - University of Utah - Salt Lake City, UT - USA
csilva@sci.utah.edu

Abstract

The growing importance of subject-specific modeling and simulation in medical applications has increased the need for automatic techniques for creating high-quality meshes directly from medical data. We discuss the main aspects related to volumetric mesh generation from iso-surfaces. We take a practical approach, and the main focus of this paper is evaluating processing pipelines using widely available tools. In our processing pipelines, we are trying to evaluate both the surface mesh and tetrahedral mesh quality, and their interactions. For the iso-surface extraction, we explore a number of widely available tools, in particular, Afront[15], CGAL[2], Macet[5], Dual Contouring[9] and Marching Cubes[11]. For tetrahedral mesh generation, we explore using TetGen[16], NetGen[13] and CAMAL[3]. We use VisTrails [1], a provenance-enabled workflow system, for assembling the processing pipelines, and comparing the results. Our plan is to make our processing pipelines available for the community, so that our results can be fully reproduced by others. In fact, our hope is that the comparison methodology used here makes it easier for others to build and compare alternative processing pipelines.

Introduction

Tetrahedral mesh generation from volumetric data is one of the most intricate problems in modeling and simulation for medical applications. Difficulty comes from several complex steps that are prone to numerical error which comprise the processing pipeline (see figure 1). For instance, the iso-surface extraction step, typically employed to generate the polyhedral surface mesh that bounds the simulation domain, can dramatically impact the subsequent step of tetrahedral mesh generation, since this step is very sensitive to badly-shaped elements in the input surface mesh. A great deal of effort has been spent to solve each of these tasks independently, but little work has been done on evaluating the different combinations of algorithms in each of these steps to determine how to obtain the best results for a given situation.

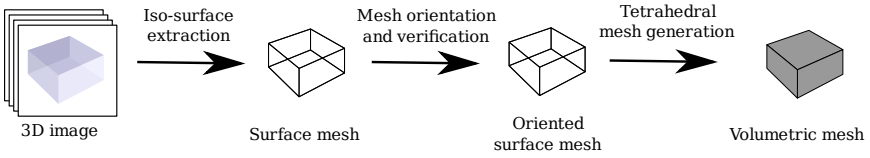

Figure 1: The traditional pipeline for tetrahedral mesh generation from volumetric data (disregarding the volume pre-processing).

In this work, we present an appropriated framework that allows a more elaborate investigation of the several aspects involved in this pipeline, as well as, allowing for combination of possible solutions at each step. Our work uses Vistrails [1], a provenance-enabled workflow system, for supporting the comparative analysis of the several different configurations. Since each algorithm comprises a building block in the workflow, exploring other alternatives such as replacing building blocks is naturally accomplished. The system also provides complete provenance on the changes made to the processing pipeline by the user. Also, comparison of different pipelines is easily performed since output can be sent to a spreadsheet that allow simultaneous exploration of the results using the same views. Most importantly, all the results generated in a given experiment can be easily reproduced elsewhere by simply sharing the Vistrail description, which allows others to add new comparisons or other components.

In our current analysis we are considering five different isosurface extraction algorithms (Afront[15], CGAL[2], Macet[5], Dual Contouring[9] and Marching Cubes[11]) and three tetrahedralization methods (TetGen[16], NetGen[13] and CAMAL[3]). One of the interesting results of our work is to show that a provenance-enabled, workflow analysis tool, such as Vistrails, is of great usefulness in managing a large-scale comparison experiment. We illustrate the effectiveness of Vistrails by summarizing the main results of comparing different combinations of polyhedral and tetrahedral mesh generators.

Mesh Generation Tools

In this section, we briefly describe the algorithms utilized in the comparisons. Our intent here is not to supply a detailed review about each technique, but highlight the computational and mathematical mechanisms these techniques are based on, thus making clear their requirements and behavior.

Polyhedral Surface Mesh Generators

Afront: Afront is an advancing-front triangulation algorithm that can be applied to many different meshing scenarios. It makes use of a guidance field to determine triangle sizing that is adaptive to the curvature of the input surface, but also maintains smooth gradation to prevent poor quality triangles from being created. The prototype implementation can be used to generate optimized meshes from other meshes (remeshing), from volumes defined on regular and irregular grids (isosurface extraction), and from point sets. It can be used to mesh either the entire input surface or just a local region, and can preserve sharp features

that are annotated ahead of time (see Schreiner, et al., [15]).

Marching Cubes: Marching cubes is a well-known algorithm for extracting a triangle mesh of an isosurface from a 3D scalar field. The algorithm proceeds through the scalar field, taking eight neighbor locations at the corners of each cube within the base mesh, and determining the triangles needed to represent the part of the isosurface that passes through each cube. The individual triangles are fused into the desired surface. For efficiency, each cube in the base mesh is assigned a template of predetermined triangle arrangements from a precomputed list of 256 possible configurations based on where the isosurface intersections within the cube occur. Finally, each vertex of the generated triangles is placed in the appropriate position along the cube's edge by linearly interpolating the two scalar values connected by that edge.

Macet: Marching Cubes is a popular choice for isosurface extraction from regular grids due to its simplicity, robustness, and efficiency. However, one of the key shortcomings of this approach is the quality of the resulting meshes, which tend to have many poorly shaped and degenerate triangles. This issue is often addressed through post-processing operations, including smoothing. While these improve the mesh, they do not remove all degeneracies, and incur an increased and unbounded error between the resulting mesh and the original isosurface. Rather than modifying the resulting mesh, the Macet algorithm modifies the grid on which Marching Cubes operates before the mesh is generated. Considering the edges of the Marching Cubes grid that contain intersections (active edges), the algorithm applies transformations that change edge locations to improve the quality of the resulting triangles. These edge transformations are embedded in the core of the Marching Cubes algorithm and incur in an overhead of up to twice the time of the original algorithm, but greatly increase the quality of the worst triangle in the extracted isosurface.

Dual Contouring: Dual Contouring is a feature-preserving, iso-surfacing method to extract crack-free surfaces from both uniform and adaptive octree grids. This technique can be seen as an hybrid of the Extended Marching Cubes [10] and SurfaceNets [8] as it make use of Hermite data and quadratic error function minimization to position the vertices of the surface mesh (as Extended Marching Cubes) and the dual topology to connect such vertices (as SurfaceNets) . Dual Contouring tends to generate better quality triangles than Marching Cubes while still being very effective in representing sharp features, rendering this surface mesh generation technique a good alternative to the popular Marching Cubes.

CGAL: CGAL is a C++ library that provides easy access to efficient and reliable geometric algorithms, including a surface mesh generation technique based on Delaunay triangulation [2]. The algorithm implemented in CGAL uses the notion of ϵ-sample, a concept widely employed in surface reconstruction from unorganized points, to build triangulated surfaces that are topologically equivalent and geometrically close to the original surface.

Tetrahedral Mesh Generators

We have utilized two freely available tetrahedral mesh generation tools for ease of use and availability. We have also included comparisons with a commercially available tetrahedral

mesh generation package.

TetGen: TetGen corresponds to a suite of techniques to generate different tetrahedral meshes from three-dimensional point sets or domains with piecewise linear boundaries. In particular, we use the module that computes the Constrained Delaunay Tetrahedralization (CDT) from the isosurface mesh produced by one of the algorithms described in the previous section. The TetGen implementation of the CDT is based on the incremental edge flipping algorithm proposed in [6]. Since this algorithm preserves triangles in the boundary, the edge flipping test is very sensitive to the quality of the input mesh, and in some situations numerical problems are so severe that no CDT is generated.

NetGen: NetGen is an automatic 3D advancing-front tetrahedral mesh generator that accepts input from constructive solid geometry (CSG) or boundary representations (BRep) from the STL file format. NetGen contains modules for mesh optimization and hierarchical mesh refinement. Netgen is open source based on the LGPL license, and is available for Unix/Linux and Windows.

CAMAL: The CUBIT Adaptive Meshing Algorithm Library (CAMAL) [3] contains several of the CUBIT [3] project's mesh generation algorithms. CUBIT's goal is robust and unattended mesh generation of complex geometries, scalable to millions of elements and thousands of parts. CUBIT is best known for its pioneering work on automated quadrilateral and hexahedral mesh generation, but also maintains robust triangle and tetrahedral meshing technologies. The tetrahedral mesh generation tool included in CAMAL is TetMesh-GHS3D[12], a package to automatically create tetrahedral meshes from of closed triangular surface meshes, with little or no user interaction. The implementation is based on the algorithm described in [7], and corresponds to another variation of a Constrained Delaunay Triangulation, and is, therefore, sensitive to the quality of the input mesh and may be prone to generate tetrahedral meshes with slivers if the boundary mesh does not maintain high quality.

Comparisons

In this section we present a summary of the main results we have obtained in comparing the different configurations for the tetrahedral mesh generation pipeline. Besides showing these quantitative and qualitative results we show how useful Vistrails is in the organization and management of the tests we have carried out. In fact, Vistrails makes the task of handling the pipeline much more friendly and intuitive, as the whole pipeline can graphically be depicted. Furthermore, the user can interact with the graphical workflow, changing components or creating new branches. Figure 2 shows the Vistrails interface and the workflow we have devised to verify how TetGen, NetGen, and CAMAL behave when Afront is used as the surface extraction algorithm.

Figure 2 shows (highlighted on top of the workflow window) the box representing the Afront algorithm. It is important to point out that by only changing the Afront box to another surface mesh generation tool (Marching Cubes, for example), a completely new set of tests can completed. Another important benefit provided by Vistrails is that all the

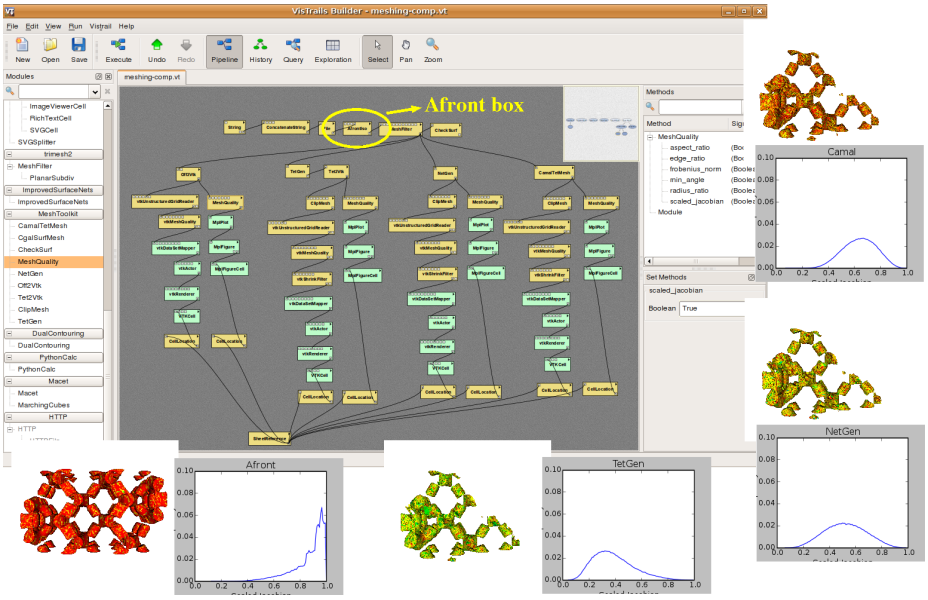

Figure 2: Vistrails interface: a) workflow employed to verify the behaviour of TetGen, NetGen, CAMAL when Afront is used as surface mesh extractor; b) The box representing the Afront algorithm.

tests conducted in previous run can be stored so as to keep the complete history of the experiments.

Table 1 shows some results selected from the tests we have carried out. Figure 3 shows the meshes of the silicium data set. The tests have been run on an AMD Opteron 242 processor (1.6 GHz) with 2 GB of RAM memory. NetGen has turned out to be less robust than TetGen and CAMAL, as noted in the silicium data set, where the NetGen mesher was not able to produce an appropriate tetraedral mesh for each of the given surface meshes. We can also see from table 1 that the combination Afront-CAMAL tends to generate a larger number of tetrahedra than other combinations, and the worst tetrahedron (we have used the Scaled-Jacobian measure, provided by Verdict Library [14]) is generated when Marching Cubes is used as surface mesh extractor.

Conclusion

We have provided comparisons for several surface (triangle) and volume (tetrahedra) mesh generation tools on a collection of models. These tools were implemented in a provenance-enabled workflow system utilizing VisTrails which enabled quick and easy comparisons for timing, sizing and quality between the various methods. The tools highlighted are used "out-of-the-box" and no additional optimization of the mesh was performed. Improved results for each of the tools may be possible. Because of the growing need for freely available mesh generation tools for research purposes, this comparison was viewed as timely to provide data for varied research efforts to find and implement tools that will provide adequate results, specifically for biomedical mesh generation activities.

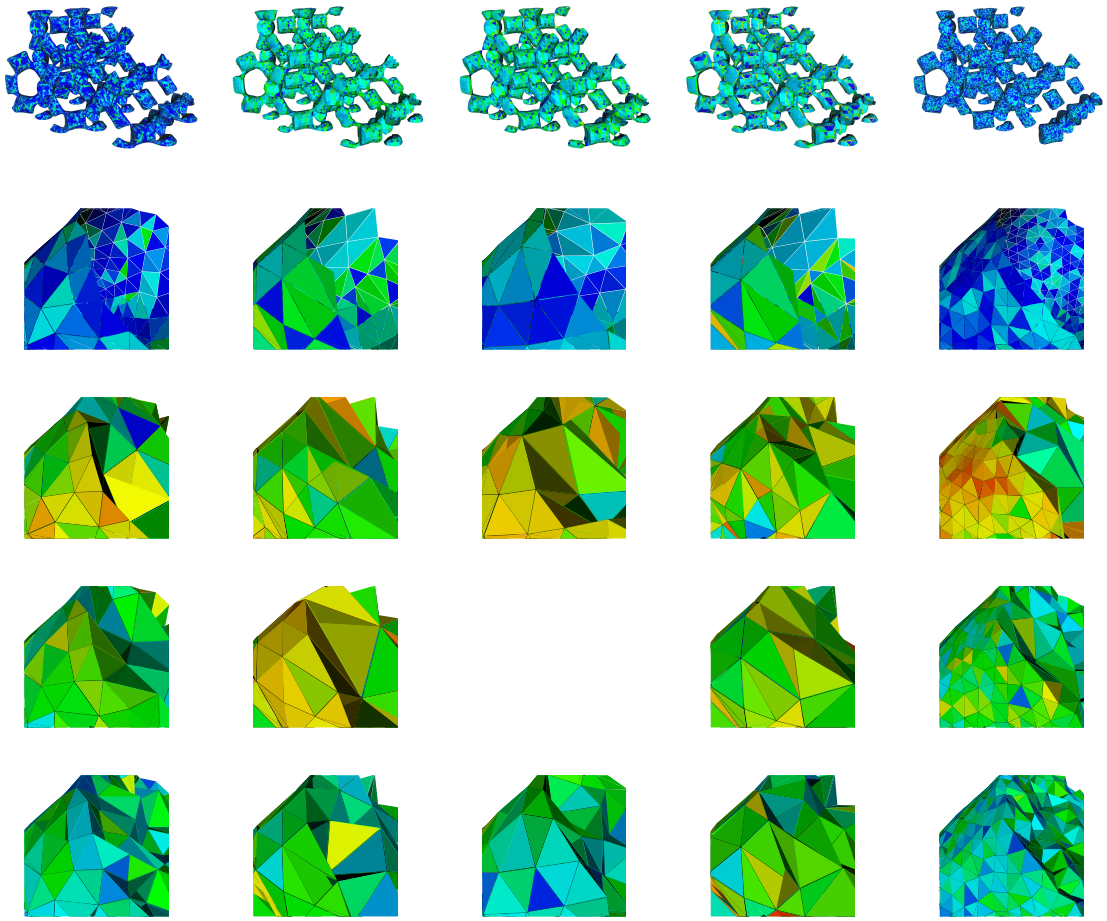

Figure 3: Silicium meshes from left to right: Afront, Macet, Dual Contouring, Marching Cubes, CGAL. Top to botton: Surface mesh, zoomed surface mesh, TetGen, NetGen, CAMAL.

Acknowledgments: C. Silva is funded by the National Science Foundation (grants CCF-0401498, EIA-0323604, OISE-0405402, IIS-0513692, CCF-0528201), the Department of Energy, and an IBM Faculty Award. Mario A. S. Liziér is funded by CAPES-Brazil and L.G. Nonato is funded by CNPq-Brazil (grant 308292/2006-5). J. Comba is funded by CNPq-Brazil (grant 485853/2007-8).

References

[1] L. Bavoil, S. P. Callahan, P. J. Crossno, J. Freire, C. E. Scheidegger, C. T. Silva, and H. T. Vo. Vistrails: Enabling interactive multiple-view visualizationss. *Proceedings of IEEE Visualization*, 2005.

[2] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of surfaces. *Graph. Models*, 67(5):405–451, 2005.

[3] The CUBIT Adaptive Meshing Algorithm Library, Sandia National Laboratories, http://cubit.sandia.gov/camal.html, 2007.

[4] The CUBIT Geometry and Mesh Generation Toolkit, Sandia National Laboratories, http://cubit.sandia.gov/, 2007.

Model	Surface Tool	Tetra Tool	Time	# Tetra	# Vertices	Worst Tetra	Mean	Variance
Silicium	Afront	TetGen	35s	221k	71k	0.029	0.388	0.023
		NetGen	230s	252k	78k	0.065	0.526	0.024
		CAMAL	39s	521k	121k	0.069	0.643	0.016
	Macet	TetGen	11s	68k	21k	0.015	0.400	0.027
		NetGen	53s	52k	17k	0.046	0.432	0.027
		CAMAL	9s	91k	23k	0.092	0.584	0.018
	Dual Cont	TetGen	14s	79s	24k	0.006	0.403	0.030
		CAMAL	14s	88k	23k	0.024	0.571	0.021
	M. Cubes	TetGen	175s	288k	77k	1e-05	0.321	0.037
		NetGen	88s	69k	18k	0.005	0.473	0.041
		CAMAL	18s	92k	23k	0.001	0.495	0.040
	CGAL	TetGen	68s	352k	111k	0.034	0.355	0.030
		NetGen	616s	518k	140k	0.093	0.593	0.019
		CAMAL	100s	1396k	282k	0.138	0.652	0.013
Brain	Afront	TetGen	194s	1407k	369k	0.024	0.387	0.026
		CAMAL	1633s	16728k	2896k	0.025	0.649	0.014
	Macet	TetGen	234s	979k	234k	0.015	0.406	0.026
		CAMAL	558s	5838k	1019k	0.102	0.641	0.014
	Dual Cont	TetGen	206s	827k	208k	0.013	0.397	0.027
		CAMAL	567s	6382k	1109k	0.052	0.644	0.014
	M. Cubes	TetGen	337s	1471k	351k	9e-05	0.402	0.028
		CAMAL	620s	5923k	1034k	0.059	0.634	0.016
	CGAL	TetGen	63s	439k	120k	0.025	0.362	0.030
		CAMAL	238s	3561k	638k	0.160	0.654	0.013
Engine	Afront	TetGen	1016s	5044k	1295k	1e-10	0.383	0.025
		CAMAL	1710s	22956k	4284k	0.007	0.636	0.015
	Macet	TetGen	2012s	2364k	610k	0.008	0.397	0.028
		CAMAL	655s	7091k	1337k	0.004	0.623	0.015
	M. Cubes	TetGen	68452s	7102k	1739k	1e-10	0.335	0.034
		CAMAL	735s	6830k	1293k	1e-04	0.602	0.024
	CGAL	TetGen	114s	697k	202k	0.030	0.368	0.033
		NetGen	3340s	2806k	559k	0.089	0.691	0.012
		CAMAL	199s	3047k	590k	0.097	0.636	0.014
Bonsai	Afront	TetGen	604s	3649k	871k	0.010	0.389	0.024
		CAMAL	2034s	20895k	3799k	0.014	0.644	0.014
	Macet	TetGen	1166s	1800k	433k	2e-04	0.401	0.027
		CAMAL	761s	7424k	1346k	0.059	0.634	0.015
	M. Cubes	TetGen	41925s	5667k	1303k	1e-10	0.330	0.035
		CAMAL	762s	6816k	1244k	2e-04	0.614	0.023
	CGAL	TetGen	117s	657k	197k	0.031	0.358	0.030
		CAMAL	239s	4234k	786k	0.150	0.648	0.013

Table 1: Volumetric meshes

[5] C. A. Dietrich, C. Scheidegger, J. Schreiner, J. L. D. Comba, L. P. Nedel, and C. Silva. Edge transformations for improving mesh quality of marching cubes. *IEEE Transactions on Visualization and Computer Graphics (to appear).*, 2008.

[6] H. Edelsbrunner and N. R. Shah. Incremental topological flipping works for regular triangulations. In *SCG '92: Proceedings of the eighth annual symposium on Computational geometry*, pages 43–52, New York, NY, USA, 1992. ACM.

[7] P. L. George, F. Hecht, and E. Saltel. Automatic mesh generator with specified boundary. *Comput. Methods Appl. Mech. Eng.*, 92(3):269–288, 1991.

[8] S. Gibson. Using distance maps for accurate surface representation in sampled volumes. In *VVS '98: Proceedings of the 1998 IEEE symposium on Volume visualization*, pages 23–30, New York, NY, USA, 1998. ACM.

[9] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 339–346, New York, NY, USA, 2002. ACM.

[10] L. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 57–66, New York, NY, USA, 2001. ACM.

[11] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, 1987.

[12] M. Loriot. TetMesh-GHS3D v3.1 the fast, reliable, high quality tetrahedral mesh generator and optimiser, http://www.simulog.fr/mesh/tetmesh3p1d-wp.pdf, 2006.

[13] NETGEN - automatic mesh generator, Johannes Kepler University Linz, http://www.hpfem.jku.at/netgen/, 2008.

[14] Pébay, Thompson, Shepherd, Knupp, Lisle, Magnotta, and Grosland. *New Applications of the Verdict Library for Standardized Mesh Verification Pre, Post, and End-to-End Processing*, pages 535–552. 2008.

[15] J. Schreiner and C. Scheidegger. High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1205–1212, 2006. Member-Claudio Silva.

[16] H. Si. On refinement of constrained delaunay tetrahedralizations. *Proceedings of the 15th International Meshing Roundtable*, 2006.