Three-dimensional modeling and highly refined mesh generation of the aorta artery and its tunics

J A Cazotto¹, L A Neves², J M Machado², J C Momente², Y Shiyou⁴, M F Godoy³, G F D Zafalon², A R Pinto², C R Valêncio²

¹IGCE, Department of Statistics, Applied Mathematics and Computation (DEMAC), São Paulo State University (UNESP), Rio Claro, SP, Brazil.
²Department of Computer Science and Statistics, São Paulo State University, São José do Rio Preto, SP, Brazil.
³Interdisc. Center for the Study of Chaos and Complexity (NUTTECC), Faculty of Medicine of São José do Rio Preto (FAMERP), São José do Rio Preto, SP, Brazil.
⁴College of Electrical Engineering, Zhejiang University, Hangzhou, People Republic of China.

E-mail: jessica.cazotto@gmail.com

Abstract. This paper describes strategies and techniques to perform modeling and automatic mesh generation of the aorta artery and its tunics (adventitia, media and intima walls), using open source codes. The models were constructed in the Blender package and Python scripts were used to export the data necessary for the mesh generation in TetGen. The strategies proposed are able to provide meshes of complicated and irregular volumes, with a large number of mesh elements involved (12,000,000 tetrahedrons approximately). These meshes can be used to perform computational simulations by Finite Element Method (FEM).

1. Introduction
Computational fluid dynamics (CFD) methods based on three-dimensional vessel representations have recently shown to provide prognostically relevant hemodynamic data [1]. However, the automatic mesh generation of a given geometric domain is one of the most complex software problems which must be solved, mainly considering a high degree of refinement and obeying the solid boundaries conveniently. The assessment of clinically relevant parameters may depend on the histological features represented, which is not a trivial task. The models presented are restricted to the cases when histological features or free use conditions are necessary [2]. Models considering histological results may provide more clinically relevant information, and when built in open source codes they allow modifications without restriction. In this context, there are only a few examples of open source codes available, specially those dedicated to the solid modeling and automatic constrained tetrahedralization, which may be used to investigate three-dimensional vessel models based on FEM simulations [2].

In this context, we describe strategies to generate tetrahedral meshes of the aorta considering adventitia, media and intima walls. The strategies here proposed were based on open source code integration for vessel modeling and automatic mesh generation with a high degree of refinement.
2. Methodology

The aorta was constructed with macro and microscopic features. The macroscopic features are those which define the aorta path and ramifications. The model considers a structure with four segments: ascending aorta, aortic arch, thoracic aorta and abdominal aorta. The considered ramifications were the common carotid arteries, subclavian arteries, brachiocephalic artery, renal arteries and common iliacs. The dimensions were defined by a cardiologist. The microscopic features were dimensions of the adventitia and media tunic, with approximately 0.8 mm of thickness each, and the intima tunic with 130 µm of thickness. The proposed model was built with the Blender package (maintained by Blender Foundation) [3], an integrated system of tools and available under a dual license (BL / GNU General Public License). This package includes support to Python script language, which is based on a high level programming language model. We represented the aorta with the Blender package and used Python script to export the domain, which was used in the TetGen [4]. TetGen generates tridimensional meshes from the Delaunay algorithm [4], which is one of the most popular and robust algorithms for domain discretization.

The model was built with strategies that ensured the mesh generation with TetGen. The first strategy was the using of quadrilateral faces to discretize domains without bifurcation. The second was to consider triangular faces to represent regions with bifurcation. The triangular faces provide appropriate representations of regions with acute angles, like those occurring in bifurcation regions. For the aorta, the quadrilateral faces were used to define the ascending aorta, aortic arch, brachiocephalic artery, common carotid arteries, subclavian arteries, thoracic aorta, abdominal aorta and common iliac arteries. The triangular faces were used to represent the abdominal aorta bifurcation to the common iliac arteries and to the renal arteries, the aortic arch bifurcations to the left common carotid artery, to the left subclavian artery and to the brachiocephalic artery and the brachiocephalic bifurcations to the right common carotid artery and to the right subclavian artery. The histological particularities of the structure were represented with properties that enable highly refined meshes. These properties were face distortion minimization and density control of vertices. The first indicates that quadrilateral faces can be modified to fill regions with acute angles, since the internal angles respect values between 45 and 135 degrees as shown in figure 1. These limits avoided errors in the mesh generator stage. For a quadrilateral face not included in this interval, the region was discretized with a triangular element. A triangular element (triangular face) can be equilateral, isosceles or scalene which might be seen in figure 2a. The second property used was the density control of vertices, which defines the number of nodes in a region. The increasing number of vertices allows a better representation of curved regions, smoothing the direction transition, respecting the features defined on the first property and the orthogonality of the faces (figure 2b).

The aorta was stored into Blender files (.blend), which are a kind of container designated as datablock. An object datablock contains information about geometric operations applied into a given solid model. A mesh datablock stores information about vertices, edges, faces and holes, which resulted from the operations assigned by an object datablock. The integration of the models constructed by Blender, with the mesh generator Tetgen, was accomplished by Python scripts [5]. The script operates into objects generated by the Blender package through four iterative structures. These structures read the data contained in mesh and object datablocks, and translate the information about vertices, faces, holes and attributes accordingly. The processed information is an ASCII file (.poly), properly written in the formats required by TetGen. Since the iteration loops in [5] export only the active solid, some adaptation is necessary to provide the meshing of mixed solids. The performed adaptation executes the process for each member in the group of existing solids and generates a single ASCII file (.poly), composed by 4 parts: an indexed list of point coordinates; a list of solid faces; a list of volume holes; and, a list of attributes or boundaries (constraints).

The aorta was constructed and meshed in a computer with processor Intel® CoreTM i5, model 2410M, with 2.3 GHz, 4 GB of RAM and with a dedicated video card (AMD Radeon HD 6470M). The operating system used was the Windows 7 with 64 bits, running the Blender version 2.49b and the mesh generator TetGen version 1.4.
3. Results

The aorta was built using the concept of integration mentioned in section 2 and it is shown in Figure 3. Figure 3a shows the aorta that was constructed with the information from the ASCII file (.poly). The generated mesh of the adventitia tunic is showed in Figure 3b, which was obtained with 3,741,764 tetrahedral elements in approximately 16 minutes. The mesh of the media tunic was built in 4 parts. The part which requested most time was generated in approximately 49 minutes and has 7,930,965 tetrahedral elements. Finally, the intima tunic was generated in 79 minutes and has 11,778,860 elements. For instance, a transversal cut applied in the plane x,y, which is presented in the Figure 4, allows more detailed information about the adventitia and media walls histology meshed. Notice these blood vases comprise distinct layers of tissues with different material properties, and therefore a good domain mesh must represent these layers conveniently.

4. Conclusions

We have introduced strategies for the integration of data structures and vessel representations, based on the open source packages Blender and Tetgen. We have constructed the aorta artery that is a complex example, considering the abdominal aorta, the aortic arch and the brachiocephalic artery, as well as the bifurcations of these vessels. All these structures were generated successfully with the corresponding tunics. The adventitia tunic has 3,741,764 elements. Due to the topology and thickness of the media and the intima tunics, they have almost 8,000,000 and 12,000,000 elements, respectively. The approach here presented may already give a contribution for FEM simulations of several phenomena in blood vases, using highly refined tetrahedral meshes.
Figure 3. The aorta artery (a), illustration of a transversal cut applied in the plane $z, x$ of the aortic arch (b), and an enlargement to a more detailed view ((c) and (d)).

Figure 4. A transversal cut applied in the plane $x, y$ of the adventitia tunic (a) and a part of the media tunic (b).

Acknowledgments
This study was supported by UNESP/PROPe (Pró-Reitoria de Pesquisa) and part by FAPESP.

References
[1] Hayase H, Tokunaga K, Nakayama T, Sugiu K, Nishida A, Arimitsu S, Hishikawa T, Ono S, Ohta M and Date I 2011 Computational fluid dynamics of carotid arteries after carotid endarterectomy or carotid artery stenting based on postoperative patient-specific computed tomography angiography and ultrasound flow data *Neurosurgery* **68** 1096–101
[2] Milašinović D, Ivanović M, Tengg-Kobligk H, Böckler D and Filipović N 2008 Software tools for generating CFD simulation models of blood flow from CT images, and for postprocessing *Journal of the Serbian Society for Computational Mechanics* **2** 51–58
[3] Blender Foundation. [Online]. Available in: http://www.blender.org/. Accessed on: May 2012
[4] TetGen, User’s Manual. Available in: http://tetgen.berlios.de/files/tetgen-manual.pdf Accessed on: May 2012
[5] Pedroso D 2007 TetGen export. [Online]. Available in: http://cvs.savannah.gnu.org/viewvc/*checkout*/mechsys/mechsys/src/py_scripts/tetgen_export.py