VI International Conference on Adaptive Modeling and Simulation
ADMOS 2013
J. P. Moitinho de Almeida, P. Díez, C. Tiago and N. Parés (Eds)

FROM SEGMENTED MEDICAL IMAGES TO SURFACE AND VOLUME MESHES, USING EXISTING TOOLS AND ALGORITHMS

CLAUDIO LOBOS[*], **RODRIGO ROJAS-MORALEDA**[†]

[*,†] Departamento de Informática
Universidad Técnica Federico Santa María
Av. España 1680, 2390123, Valparaíso, Chile
e-mail: clobos@inf.utfsm.cl[*]
rodrigo.rojas@postgrado.usm.cl[†]

Key words: Surface and Volume Meshing, Segmented Images, Finite Element Method.

Abstract. In a medical context, one of the most used techniques to produce an initial mesh (starting from segmented medical images) is the Marching Cubes (MC) introduced by Lorensen and Cline in [1]. Unfortunately, *MC* presents several issues in the meshing context. These problems can be summarized in three types: topological (presence of holes), of quality (sharp triangles) and accuracy in the representation of the target domain (the staircase effect). Even though there are several solutions to overcome topological and quality issues, the staircase effect remains as a challenging problem.

On the other hand, the Computational Geometry Algorithms Library (CGAL) [2], has implemented the *Poisson Surface Reconstruction* algorithm introduced in [3], which is capable of producing accurate and high quality triangulations based on a point set and its normal directions.

This paper shows how surface meshes can be produced using both, *MC* and CGAL. Moreover, starting from the generated quality surface mesh, this work also shows how volume meshes can be produced. Therefore, a complete workflow, starting from segmented medical images to surface and volume meshes, is introduced in this work. In particular, tetrahedral and mixed–element meshing techniques are presented to produce a simulation with the Finite Element Method.

1 INTRODUCTION

In the context of biomechanical modelling of human organs, the most common way to produce a mesh suitable for biomedical applications is to perform the following steps:

- Acquire volumetric medical images of the patient's organ using magnetic resonance imaging (MRI), ultrasound (US), computed tomography scanner (CT) or other imaging techniques.
- Achieve image segmentation in order to produce a cloud of points or an initial surface mesh defining the geometry of the modelled domain.
- Add internal nodes to produce the 3D elements that will conform the final volumetric mesh.

One of the most used methods for extracting surfaces from volumetric information arising from sources such as MRI data and CAT scans is *implicit modelling*, leading to an abundance of approaches. One of the most widespread approach is *Marching Cubes* (MC) [1], which belongs to the family of techniques that uses a structured grid for implicit modelling [4].

One apparent drawback of *MC* is the uniform sampling density which does not take the local surface curvature into account, producing problems like: *(i)* presence of holes or surface crossing (topology), *(ii)* sharp and flat triangles (quality) and *(iii)* the *MC* characteristic staircase effect. Even though many approaches have been proposed to control the mesh complexity, e.g., by adaptive octree descent with sophisticated refinements, the quality and the staircase effect remain as challenging problems.

The *Poisson Surface Reconstruction Method* (PSR) tackles these issues by estimating the surface as the iso-contour to an indicator function defined from the normal vectors to the *MC* boundary. The Computational Geometry Algorithms Library (CGAL) implements a variant of this algorithm which solves for a piecewise linear function [2].

This paper shows how to obtain high accuracy surfaces, that approximate a volume outer boundary from *MC* algorithms using a *PSR* method implemented in *CGAL*. Moreover, this paper shows how to obtain volume meshes starting from the acquired surface mesh.

2 MATERIALS AND METHODS

The overall proposed workflow is shown in Figure 1, where three main components are introduced: input data, surface meshing and volume meshing. The surface meshing is based on the *MC* method, the rendered mesh is improved in order to use CGAL to apply a *PSR* method. The output of CGAL is finally used to construct a high quality volume mesh.

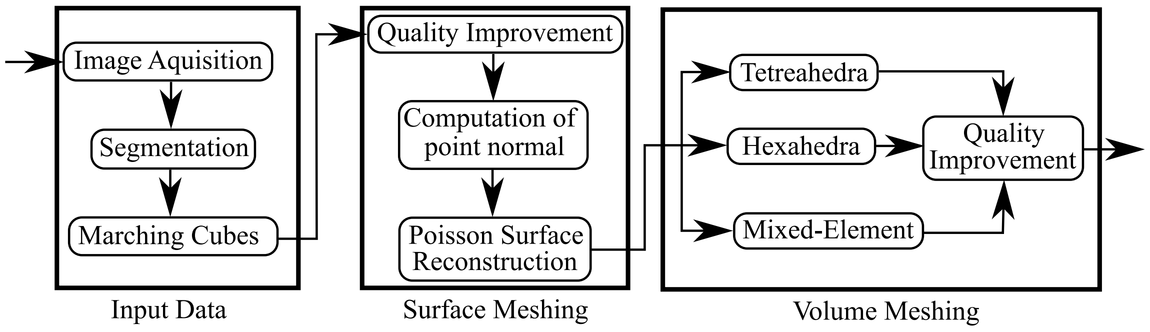

Figure 1: Main Workflow with three steps: Input Data, Surface Meshing and Volume Meshing.

2.1 Image acquisition and segmentation

Volumetric medical images from imaging acquisition technology, such as CT, MRI and PET, are represented by a stack of 2D image slices in 3D space, where the tissue type surrounding a Voxel (volumetric pixel) determines its value. The variations in tissue type give rise to varying intensity which is quantized as a scalar value known as grey level, or more often as vector or tensor values when Voxels contain multiple scalar values, e.g. in ultrasound. The segmentation problem is essentially a classification problem. A label representing the region to which an image Voxel belongs is assigned to each Voxel. The assignment is however, subject to some constraints, such as piecewise continuity and smoothness. Depending on the imaging technique, the segmentation could be difficult and a research area in itself [5].

2.2 Surface Meshing

The first step is to use *MC* to render a surface from segmented contours. The principle behind the *MC* algorithm is to subdivide space into a series of small cubes. By the values at its eight vertices -below or above an isovalue- the intersections at the cube edges are found. These intersections are used to generate triangles representing the isosurface by a lookup table of surface configurations of cubes. However, the *MC* shows important drawbacks for some surface configurations: the topology of the isosurface is not unique and the resulting surface of two adjacent neighbor cubes can produce: holes, poor quality triangles explained below and low accuracy in the representation known as the staircase effect.

With respect to holes in the triangle mesh, there are new versions of *MC* [4, 6], that address and solve this problem. Alternatively a conventional solution is the use of the *ear clipping* technique [7], adding new triangles to close the hole.

One of the most important quality metrics for surface meshes is the *minimal angle*. There are two types of poor quality triangles: sharp and flat. The first refers to a triangle that has one angle close to cero and two close to 90 degrees. A conventional solution is the edge collapsing technique, which will "join" the two nodes opposite to the "close to cero" angle. The second one has two angles close to cero and one close to 180 degrees; there is no proper or general solution for this case.

These repairing techniques allow to get a surface mesh without sharp triangles nor holes; the next section is dedicated to tackle the problem of low accuracy representation or staircase effect.

2.3 Quality meshes from implicit functions

Starting with a surface mesh free of holes and sharp triangles, problems like *staircase effect* and remaining *poor quality triangles* can be overcome by the *Poisson Surface Reconstruction* method (PSR) [2] which is available in the library CGAL. The *PSR* formulation considers [3] all points at once, without resorting to heuristic spatial partitioning or blending and so, is highly resilient to data noise. This approach allows a hierarchy of locally supported basis functions, and the solution reduces to a well conditioned sparse linear system.

CGAL takes as input a set of normal vectors (3D oriented set of points). The normal of each node is computed as the average of incident triangle normals. Given an oriented set of points. CGAL algorithm builds a 3D Delaunay triangulation from these points and refines it by *Delaunay refinement* so as to remove all badly shaped (non isotropic) triangles and to tessellate a loose bounding box of the input oriented points [2]. The *PSR* defines an indicator function $\tilde{\chi}$ as 1 at *points inside* the model, and 0 at *points outside*. A new quality surface is obtained by extracting an isosurface.

The principle behind this method is the relationship between oriented points (normal vectors) and the indicator function. The gradient of the indicator function is a vector field that is zero almost everywhere (since the indicator function is constant almost everywhere), except at points near the surface, where it is equal to the inward surface normal. Thus, the oriented point set can be considered as a sample points from the gradient of the indicator function shown in Figure 2, [3].

The indicator function thus reduces to inverting the gradient operator, i.e. finding the scalar function $\tilde{\chi}$ whose gradient best approximates a vector field $\vec{V}$ defined by the samples, i.e.

$$Min|\nabla\tilde{\chi} - \vec{V}|.$$

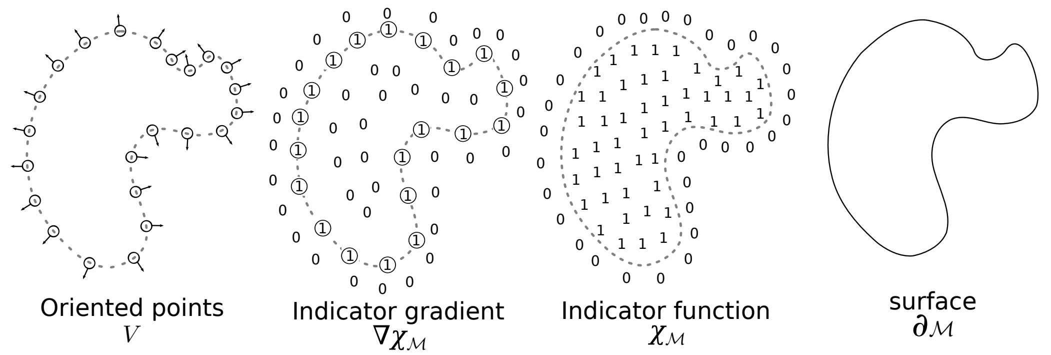

Figure 2: Four stages to reconstruct a surface: obtain a set of sample oriented points, each oriented point can be considered as a sample point of the gradient of some indicator function, the isosurface induced by the indicator function defines the reconstructed model.

Applying the divergence operator, this variational problem transforms into a standard Poisson problem: *Compute the scalar function $\tilde{\chi}$ whose Laplacian (divergence of gradient) equals the divergence of the vector field $\vec{V}$,*

$$\Delta\tilde{\chi} \equiv \nabla \cdot \nabla\vec{\chi} = \nabla \cdot \tilde{V}.$$

In order to implement and solve this approach in a discrete domain, some problems must be considered [3].

2.3.1 Define the gradient field

Because the indicator function is a piecewise constant function, explicit computation of its gradient field would result in a vector field with unbounded values at the surface boundary. To avoid this, the indicator function is convolved with a smoothing filter and considers the gradient field of the smoothed function [3].

$$\nabla(\chi_M * \tilde{F})(q_0) = \int_{\partial M} \tilde{F}(q_0)\bar{N}_{\partial M}(p)dp$$

2.3.2 Approximating the gradient field

Using the point set S to partition ∂M into distinct patches $\wp_s \subset \partial M$, we can approximate the integral over a patch $\wp_s$ by the value at point sample $s.p$, scaled by the area of the patch:

$$\nabla(\chi_M * \tilde{F})(q) \approx \sum_{s\in S} |\wp_s|\tilde{F}_{s.p}(q)s.\vec{N} \equiv \vec{V}(q)$$

2.3.3 Solving the Poisson problem

Having formed a vector field $\vec{V}$, we want to solve the function $\tilde{\chi}$ so that $\nabla\tilde{\chi} = \vec{V}$. However, $\vec{V}$ is generally not integrable (there is not guarantee that is curl-free), so an exact solution does not generally exist. To find the best least-squares approximate solution, we apply the divergence operator to form the *Poisson equation.*

$$\Delta\tilde{\chi} = \nabla \cdot \vec{V}$$

2.4 Volume Meshing

Following the workflow shown in Figure 1, there are three type of volume meshing analyzed in this work and they vary in terms of element type employed: tetrahedra, hexahedra or mixed–elements, meaning a mix of tetrahedra, hexahedra, pyramids and prisms (wedges). In order to compare the different results generated by these meshing techniques, it is first necessary to explain how quality of volume elements is measured.

2.4.1 Quality of volume meshes

The Jacobian Ratio (JR) is one of the most used quality criterion in hexahedra meshes [8, 9, 10, 11, 12] and in order to understand how it works, it is necessary to understand some concepts first.

The FEM establish a bijective mapping function F between the modeled domain referential (x_1, x_2, x_3), in which the elements of the mesh are defined, and a referential parent system (ξ_1, ξ_2, ξ_3) for each type of element. The Jacobian matrix J of mapping F considered at parent frame point ξ is then defined as:

$$J(\xi) := \frac{\partial F}{\partial \xi}(\xi)$$

If for some reason, F is no longer bijective, a Finite Element Analysis (FEA) cannot be carried out over the mesh. In order to detect the bijection property of F, two values must be computed for each node. The first is the determinant of J, $|J(\xi)|$. The second is the Jacobian Ratio JR defined as:

$$JR_i = \frac{|J(\xi)|_i}{|J(\xi)|_{max}}$$

where $|J(\xi)|_{max}$ is the maximum value of $|J|$ among all element nodes. Note that JR is normalized to present values between $(-\infty, 1]$.

Element category	invalid	very bad	questionable	good
JR value	$(-\infty, 0.001]$	$(0.001, 0.0\overline{3}]$	$(0.0\overline{3}, 0.2]$	$(0.2, 1]$

Table 1: Quality categories for Jacobian Ratio

It is now possible to say that F is not longer bijective when $JR \leq 0$ for at least one of its nodes. Moreover, the quality of an element e can be defined as $JR_e := |J|_{min}/|J|_{max}$ and therefore, if any element presents a $JR_e \leq 0$, then not only is the element invalid, but the entire mesh is considered as not suitable for FEA.

The JR is not only used to detect invalid elements, but also poor quality elements. A perfect element is the one with a $JR = 1$. An element is said to present "questionable" quality when $JR \in (0, 0.2)$ after the Verdict library of mesh quality metrics[1] and some authors [9, 11]. On the other hand, the documentation of one of the most important commercial Finite Element Solver (FES), ANSYS[2], states that an element with a JR value[3] less than 0.001 presents so poor quality that it is considered as invalid. Moreover, an element with a JR value $\in (0.001, 0.0\overline{3}]$ is suitable for FEA although the solution from the FES will be inaccurate. Regarding both quality boundary definitions, Table 1 shows the different categories for element quality.

The JR criterion is widely used in hexahedra meshes and it can be easily extended to prisms and pyramids by the computation of the $|J|$ for those element types. Unfortunately it is not a good quality measure for tetrahedra as the value of $|J|$ is the same for each tetrahedron node no matter their position, in other words, the value of JR for a tetrahedron is always 1. Moreover, the JR does not allow the detection of sliver (or flat) tetrahedra, which is one of the most common problems for this type of element and it adds imprecision to the solution found by a FES.

On the other hand, there is a quality criterion used by several authors [13, 14, 15] that considers important geometric aspects of the tetrahedron, like its volume V and its edge lengths l_i:

$$A_\gamma := \frac{\left(\frac{1}{6}\sum_{i=1}^{6} l_i^2\right)^{3/2}}{8.47867 \times V}$$

As it compromises the volume, it helps to detect sliver tetrahedra. Moreover, if V is the signed volume, A_γ can detect element inversion. Finally, the use of the edge

[1]The verdict mesh verification library (2007) Sandia National Laboratories, `http://cubit.sandia.gov/verdict.html`

[2]http://www.ansys.com

[3]Note that ANSYS defines the Jacobian Ratio as JR^{-1} regarding the notation used here.

lengths will avoid to consider extremely sharp tetrahedra as “good” elements.

In order to consider a similar scale to JR, the final quality of a tetrahedron is defined as $Q_{tetra} = A_{\gamma}^{-1}$. With this definition Q_{tetra} is negative for any inverted element, bad when it's close to 0 and perfect when its value is 1.

The JR might also fail in detecting inverted pyramids. For this reason a signed and normalized version of the aspect ratio (AR) presented in [16] is used. In this case, a negative AR shows the presence of inverted elements.

2.4.2 Tetrahedral meshing using TetGen

To generate tetrahedral meshes, one of the most used softwares is TetGen (`http://tetgen.org`), developed by Hang Si. This program has implemented several meshing algorithms that allow to produce a tetrahedral mesh. Some of its capabilities are: to preserve the input surface mesh and constraint the mesh generation process in order to create quality elements, e.g., constraining the minimal dihedral angle. It is important to note that TetGen, as well as CGAL, is capable of improving the quality of a surface mesh. However, TetGen will use all the nodes of the input mesh and will add more (Steiner points) in order to build a constrained or conforming Delaunay triangulation. In contrast to TetGen, CGAL may or may not use all input nodes in the process of achieving a quality triangulation. As it will be show in the result section, CGAL tends to produce meshes with less triangles and nodes than the quality mesh generated by TetGen (in the context of surface meshes generated with the MC).

2.4.3 Hexahedral meshing

One of the most common techniques to produce a hexahedral mesh is the Octree [17]. The Octree allows regions of different refinement level to coexist in the same mesh. By the use of several templates containing only hexahedra, transitions are performed between coarse and refined regions [9, 10, 18].

The Cubit Tool Suite developed at Sandia National Laboratories (`http://cubit.sandia.gov/`) implements most of the state of the art algorithms in hexahedral meshing. Unfortunately, even for academic purposes, it is not easy to obtain a license (as in the case of TetGen).

2.4.4 Mixed–Element mesheing

Once a hexahedral mesh is produced, there are two options to achieve surface representation: (1) remove all hexahedra intersecting the boundary (meaning an element with inside and outside nodes) and then fill the gap adding new hexahedra in a proper manner [9] or (2) project the outside nodes onto the surface.

A recent work [19] has proposed to replace boundary hexahedra by mixed–elements in order to improve the overall mesh quality. Once this replacement is done, project outside nodes onto the surface. This technique tends to produce less quality issues than using only hexahedral; although, none of these techniques can ensure mesh validity for a general case.

3 RESULTS AND DISCUSSIONS

The surface mesh generated for Breast simulation is shown in Figure 3. As mentioned in section 2.4.2, TetGen improves the quality of the input mesh; however, the final results have more triangles than the output generated by CGAL and may present flat triangles. Table 2 summarizes statistics for surface meshing in the context of the workflow shown in Figure 1. Finally, the CGAL mesh was contrained to present triangles with min angle above 30 degrees.

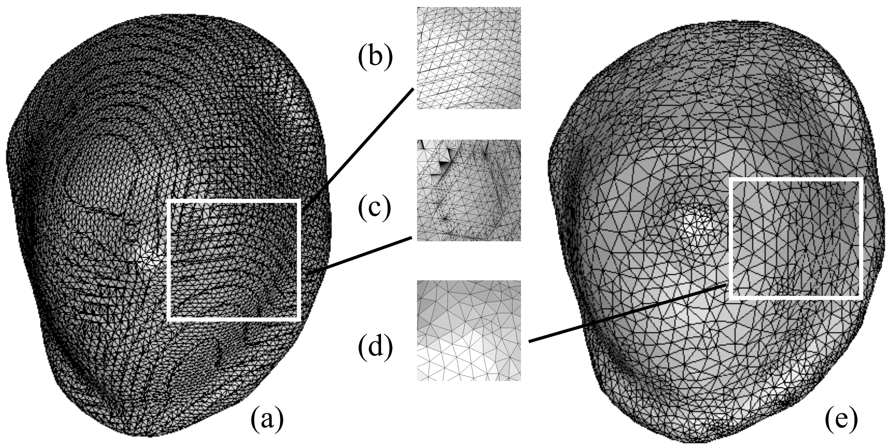

Figure 3: Breast surface meshing: (a) input data with the *MC*, (b) zoom to previous mesh, (c) zoom to the same region in a TetGen optimized mesh, (d) zoom to CGAL mesh and (e) the output mesh by CGAL

With respect to Volume meshing, the output of TetGen and the mentioned mixed–element technique are compared. Unfortunately, the Cubit output could not be compared as it was to difficult to obtain the software. Table 3 summarizes statistics for volume meshing. Finally, Figure 4 shows the results for volume meshing, where the overall mesh generated by TetGen is omitted because it is equal (in terms of surface) to the mesh generated by CGAL.

	MC	MC improved	TetGen	CGAL
nodes	12189	10104	50975	4058
triangles	24374	20204	101946	8112
min angle	0.008	0.0108	0	30
max angle	179.2	179.9	180	119.9

Table 2: Statistics for surface meshing

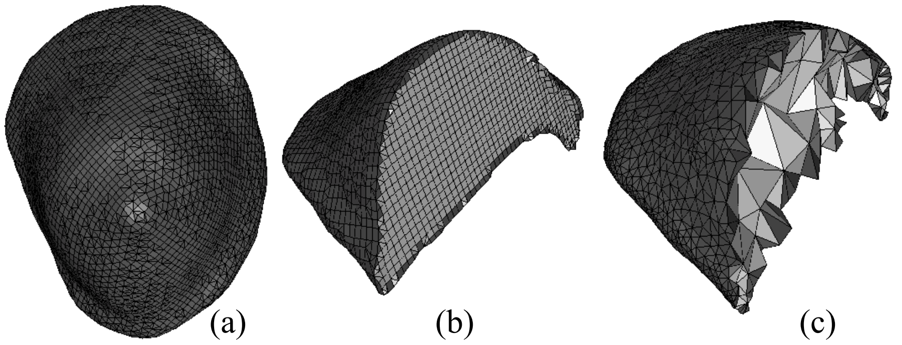

Figure 4: Breast volume meshing: (a) output by mixed–elements, (b) cut to see internal elements and (c) same cut to TetGen mesh (using the surface generated by CGAL).

4 CONCLUSIONS

In this paper, we have proposed and evaluated a workflow based of sophisticated existing tools to improve poor quality volumetric models, making the process simple enough for non–experts to use in a wide range of applications.

The major advantage of the proposed workflow is that, the ability to remove progressively poor quality elements allows to generate high quality surface meshes and subsequently an accurate volume mesh of the actual geometry, from a point–cloud acquired with conventional volumetric imaging methods.

Performance metrics over the polygonalization quality show that our workflow can successfully generate volumes free of *bad geometry problems*, with good quality triangles in many complicated surface meshes.

	Elements	Nodes	Aspect–ratio	Jacobian–ratio
TetGen	15680	4801	0.0623	–
Mixed–Elements	35748	27029	0.204	0.1894

Table 3: Statistics for volume meshing

ACKNOWLEDGES

Claudio Lobos is partially funded by: FONDECYT Iniciación 11121601, EcosConicyt C11–E01 and DGIP 24.12.33.

Rodrigo Rojas-Moraleda is partially funded by: FONDECYT 1120579, DGIP UTFSM and CONICYT Scholarships for doctoral students.

REFERENCES

[1] Lorensen, W. and Cline, H. *Marching Cubes: A high resolution 3D surface construction algorithm.* Computer Graphics (1987) **21**(4): 163–169.

[2] Alliez, P., Saboret, L. and Guennebaud, G. *Surface Reconstruction from Point Sets.* In CGAL User and Reference Manual. CGAL Editorial Board, 4.1 edition, 2012 (http://www.cgal.org).

[3] Kazhdan, M., Bolitho, M. and Hoppe, H. *Poisson Surface Reconstruction.* In Symp. on Geometry Processing (2006): 61–70.

[4] Schaefer, S. and Warren, J. *Dual Marching Cubes: Primal Contouring of Dual Grids.* In: Proceedings of Pacific Graphics (2004): 70–76.

[5] Hu, Grossberg, Mageras *Survey of Recent Volumetric Medical Image Segmentation Techniques.* In book: Biomedical Engineering, InTech (2009): 321–346.

[6] Raman, S. and Wenger, R. *Quality Isosurface Mesh Generation Using an Extended Marching Cubes Lookup Table.* Computer Graphics Forum (2008) **27**(3): 791–798.

[7] ElGindy, H., Everett, H. and Toussaint, G. *Slicing an ear using prune-and-search.* Pattern Recognition Letters (1993) **14**(9): 719–722.

[8] Knupp, P. *Achieving finite element mesh quality via optimization of the jacobian matrix norm and associated quantities. part ii – a framework for volume mesh optimization and the condition number of the jacobian matrix.* International Journal for Numerical Methods in Engineering (2000) **48**: 1165–1185.

[9] Ito, Y., Shih, A. and Soni, B. *Octree–based reasonable-quality hexahedral mesh generation using a new set of refinement templates.* International Journal for Numerical Methods in Engineering (2009) **77**(13): 1809–1833.

[10] Zhang, H. and Zhao, G. *Adaptive hexahedral mesh generation based on local domain curvature and thickness using a modified grid–based method.* Finite Element Analysis and Desing (2007) **43**(9): 691–704.

[11] Shepherd, J. and Johnson C. *Hexahedral mesh generation for biomedical models in scirun.* Engineering with Computers (2009) **25**: 97–114.

[12] Kwok, W. and Chen Z. *A simple and effective mesh quality metric for hexahedral and wedge elements.* In: Proceedings of the 9th International Meshing Roundtable (2000): 325–333.

[13] Ferrant, M., Warfield, S., Nabavi, A., Jolesz, F. and Kikinis, R. *Registration of 3d intraoperative mr images of the brain using a finite element biomechanical model.* In: Proceedings of the Third International Conference on Medical Image Computing and Computer–Assisted Intervention (2000): 19–28.

[14] Berzins, M. *Mesh quality: A function of geometry, error estimates or both?* Engineering with Computers (1999) **15**(3): 236–247.

[15] Parthasarathy, V., Graichen, C. and Hathaway, A. *A comparison of tetrahedron quality measures.* Finite Elements in Analysis and Design (1993) **15**: 255–261.

[16] Lobos, C., Bucki, M., Hitschfeld, N. and Payan, Y. *Mixed-element mesh for an intra–operative modelling of the brain tumor extraction.* In: Proceedings of the 16th International Meshing Roundtable (2007): 387–404.

[17] Shephard, M. and Georges, M. *Automatic three-dimensional mesh generation by the finite octree technique.* International Journal for Numerical Methods in Engineering (1991) **32**:709–749.

[18] Schneiders, R. *Refining quadrilateral and hexahedral element meshes.* In: Proceedings of the Fifth International Conference on Numerical Grid Generation in Computational Field Simulations (1996): 679–688.

[19] Lobos, C. *A Set of Mixed–Elements Patterns for Domain Boundary Approximation in Hexahedral Meshes.* In: Proceedings of the 20th Medicine Meets Virtual Reality (2013): 268–272