Journal Code:	Article ID	Dispatch: 07.04.13	CE: Aina Maree E. Siason
C N M	2 5 4 9	No. of Pages: 19	ME:

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN BIOMEDICAL ENGINEERING
Int. J. Numer. Meth. Biomed. Engng. (2013)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cnm.2549

Cardiovascular and lung mesh generation based on centerlines

E. Marchandise [1,*,†], C. Geuzaine [2] and J. F. Remacle [1]

[1] *Institute of Mechanics, Materials and Civil Engineering (iMMC), Université catholique de Louvain, Place du Levant 1, 1348 Louvain-la-Neuve, Belgium*
[2] *Department of Electrical Engineering and Computer Science, Montefiore Institute B28, Université de Liège, Grande Traverse 10, 4000 Liège, Belgium*

Q1

SUMMARY

We present a fully automatic procedure for the mesh generation of tubular geometries such as blood vessels or airways. The procedure is implemented in the open-source Gmsh software and relies on a centerline description of the input geometry. The presented method can generate different type of meshes: isotropic tetrahedral meshes, anisotropic tetrahedral meshes, and mixed hexahedral/tetrahedral meshes. Additionally, a multiple layered arterial wall can be generated with a variable thickness. All the generated meshes rely on a mesh size field and a mesh metric that is based on centerline descriptions, namely the distance to the centerlines and a local reference system based on the tangent and the normal directions to the centerlines. Different examples show that the proposed method is very efficient and robust and leads to high quality computational meshes. Copyright © 2013 John Wiley & Sons, Ltd.

Received 7 August 2012; Revised 11 February 2013; Accepted 13 February 2013

KEY WORDS: mesh generation; centerline; patient-specific; anisotropic mesh; boundary layer; quad mesh; hexahedral mesh

1. INTRODUCTION

Nowadays patient-specific simulations are widely used in the cardiovascular or respiratory field to investigate either local hemodynamics or deposition fraction from inhaled respiratory aerosols. An important but time consuming step in the modeling process is the creation of a quality computational mesh based on the segmented geometry. This segmented geometry is a tubular geometry with possibly several bifurcations and that is most of the times a~~n~~ standard triangulated language (STL) of low quality.

Q2

The pipeline leading from the segmented geometry to the computational mesh usually comprises of the following: (i) the generation of a high quality surface mesh; (ii) the creation and meshing of planar regions for boundary conditions; (iii) the extrusion of the surface mesh to create the geometry and volume mesh of the solid wall; and (iv) the volume mesh generation of the lumen. The latter step sometimes includes a boundary layer mesh in the vicinity of the lumen surface. Most known solutions to create a computational mesh still require to perform manually those different steps. If some of those steps (such as the tetrahedral meshing) are automated, other steps still require a large amount of manual processing.

In this paper, we present a fully automatic procedure for the mesh generation of tubular geometries. The procedure relies on a centerline description of the geometry that is computed using the open source Vascular Modeling Toolkit (*vmtk*, www.vmtk.org). The new automatic meshing

*Correspondence to: E. Marchandise, Institute of Mechanics, Materials and Civil Engineering (iMMC), Université catholique de Louvain, Place du Levant 1, 1348 Louvain-la-Neuve, Belgium.

†E-mail: emilie.marchandise@uclouvain.be

procedure is very efficient and robust and leads to high quality computational meshes. It is implemented inside the open source meshing software Gmsh [1], and examples on how to use it can be found on the Gmsh wiki (https://geuz.org/trac/gmsh).

The procedure is able to generate different type of meshes: isotropic tetrahedral meshes, anisotropic tetrahedral meshes, or mixed hexahedral/tetrahedral meshes. Additionally, a multiple layered arterial wall can be generated with a variable thickness. All the generated meshes rely on a mesh size field and a mesh metric that is based on centerline descriptions (distance to centerlines and local reference system based on the tangent and normal directions to the centerlines).

The anisotropic tetrahedral meshes are aligned with the dominant direction of flow (i.e., the centerlines) and the mesh size is reduced close to the boundary. Therefore, those anisotropic meshes reduce the number of elements and degrees of freedom (compared with the equivalent isotropic mesh), leading to significant computational savings for a given level of accuracy [2–4]. We also present a subtle way to control the tangent and normal mesh sizes for anisotropic volume meshes of tubular geometries.

Our mixed hexahedral/tetrahedral meshes contain hexahedra in the extruded wall and tetrahedra in the vessel lumen. Those meshes are based on the generation of a new quadrangular surface mesh of the input geometry. The presented algorithm of quadrangular mesh generation is an original approach that is a combination of a structured elliptic mesh generation approach [5] and an indirect approach [6] that uses distances in the L_∞ norm as a base for inserting new points and generates right triangles that are then recombined into quadrangles [7]. From the quadrangular surface mesh, we generate an extruded boundary layer mesh as well as an unstructured tetrahedral mesh of the lumen with pyramids as transition elements [8]. We are currently also working on an optimal placement of the mesh vertices in the lumen for the recombination of the tetrahedra into hexahedra [9–11], which will allow to also generate hexahedral dominant meshes. It follows that the presented unstructured approach for hexahedral mesh generation is general and quite fast compared with block-structured hexahedral mesh generation such as centerline-based sweeping methods [12, 13]. Indeed, such block-structured methods require that the source and the target have similar topology, which means that many templates for different tubular branching configurations (trifurcation or higher-order branching) are needed.

In the example section, we define for every type of mesh quality criteria. By running our centerline-based meshing algorithm on different arterial and lung geometries, we show that the presented method is very efficient and robust. Moreover, we show that the method is able to generate high quality meshes that are suitable for numerical methods, such as finite element or finite volume methods.

2. MATERIALS AND METHOD

In this section, we present the specific implementation of the pipeline of our automatic meshing approach on the basis of centerlines. The starting point of our algorithm is the segmented triangulated surface of arbitrary quality,[‡] that is, a set of N mesh triangles $\mathcal{T} = \{T_1, \ldots, T_N\}$ and a set of M surface mesh vertices $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$.

2.1. Generation of centerlines

Centerlines are powerful descriptors of the shape of vessels. Although the concept of what a centerline is more or less intuitive, their mathematical definition is not unique. A lot of methods have been proposed in the literature for the computation of centerlines both from angiographic images and 3D models.

The algorithm implemented in the open source vmtk [14–16] deals with the computation of centerlines starting from surface models and has the advantage that it is well-characterized mathematically and quite stable to perturbations on the surface. Centerlines are determined as the paths defined on Voronoi diagram sheets that minimize the integral of the radius of maximal inscribed

[‡] e.g., a surface in STL format.

DOI: 10.1002/cnm

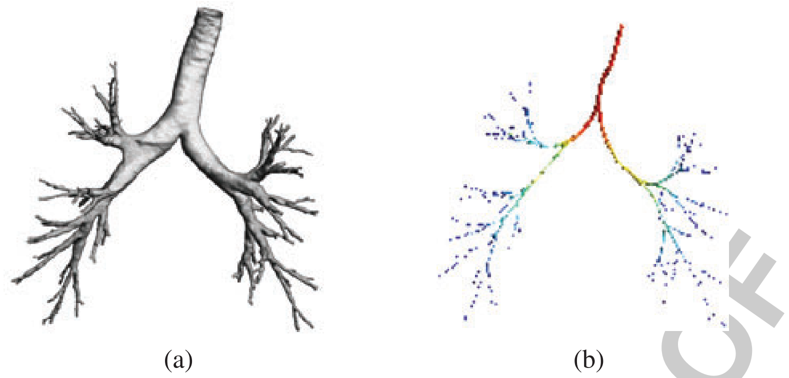


Figure 1. Geometric models of a patient-specific bronchial airway tree (a) together with the extracted centerlines (b). The colors and thickness of the centerlines represent the vessel radius (a centerline-based descriptor computed within Gmsh).

spheres along the path, which is equivalent to finding the shortest paths in the radius metric. This is performed by propagating a wave from a source point (one endpoint of the centerline) by using the inverse of the radius as the wave speed and recording the wave arrival time on all the points of the Voronoi diagram, and then backtracking the line from a target point (the other endpoint of the centerline) down along the gradient of arrival times. The propagation is described by the Eikonal equation that is computed using the fast marching method.

The script that allows to compute centerlines in vmtk is *vmtkcenterlines*.[§] For a tubular geometry, it takes as input a discrete surface (e.g., *input.stl* in STL format) and a specification of the inlet and outlet boundaries of the geometry that are used to compute the source and target points as the barycenter of those boundaries. It should be noted that prior to computing the centerlines, flow extensions can be added with vmtk for the inlets and outlets of the tubular geometry (*inputExtended.stl*), so that the computed centerlines (*centerlines.vtk*) extend out from the inlets and outlets of the input tubular geometry (input.stl).

The discrete centerlines (computed by vmtk) are a set of K consecutive line segments $\mathcal{C} = \{s_1, s_2, \ldots, s_K\}$ and L mesh vertices $\mathcal{X}_c = \{\mathbf{x}_{c_1}, \ldots, \mathbf{x}_{cL}\}$. Every line segment s_k is defined by a starting point $\mathbf{x}_{ci}$ and end point $\mathbf{x}_{cj}$ on the centerlines $s_k = \overline{\mathbf{x}_{ci}\mathbf{x}_{cj}}$. A new mesh field named *Centerline* has been implemented within Gmsh that takes as input such discrete centerlines $\mathcal{C}$. From the centerline field, two geometrical descriptors are computed and three different operators are defined.

Figure 1 shows a geometrical model of a bronchial airway tree together with the extracted centerlines. Both colors and thickness of the centerlines represent the vessel radius computed as described in the following paragraph. F1

2.2. *Centerline-based descriptors in Gmsh*

Two geometrical descriptors are computed from the centerlines $\mathcal{C}$:

- The local radius.
- The local reference system.

The local radius $r(\mathbf{x}_c)$ of the vessel is the distance from a point on the centerline $\mathbf{x}_c$ to the tubular geometry. This local radius is computed efficiently by first storing all the mesh vertices of the tubular geometry in a search tree (approximate nearest neighbor, ANN) [17, 18] and by using the search tree to compute the closest point $\mathbf{x}_p \in \mathcal{X}$. The local radius $r(\mathbf{x}_c)$ is the distance between $\mathbf{x}_p$ and $\mathbf{x}_c$.

The local reference system is a set of three axes defined for every point in the tubular volume $\mathbf{x} \in \mathbb{R}^3$. The local axes are the abscissa (unit vector tangent to the centerlines), the normal (a vector perpendicular to the centerlines), and the binormal that can be calculated as the cross product

[§] script: vmtk vmtkcenterlines - seedselector openprofiles - ifile input.stl - ofile centerlines.vtk.

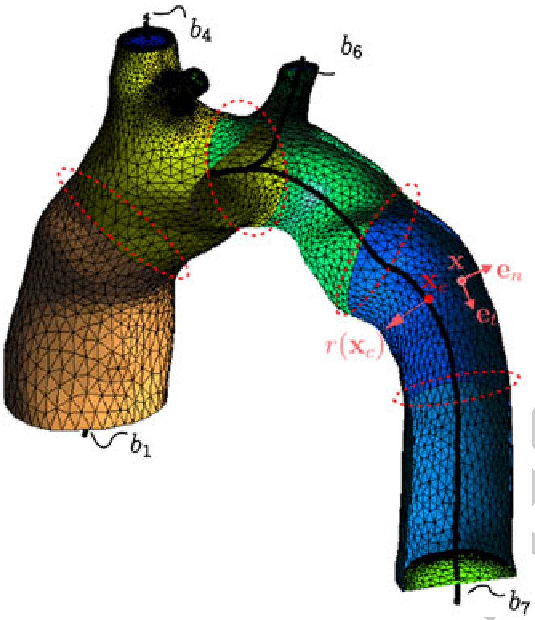


Figure 2. Centerline-based geometrical descriptors and operators used for the generation of an anisotropic mesh of a human aorta. The local radius $r(\mathbf{x}_c)$ is defined at a point on the centerlines $\mathbf{x}_c$, and the local reference system $(\mathbf{e}_t(\mathbf{x}), \mathbf{e}_n(\mathbf{x}), \mathbf{e}_{bn}(\mathbf{x}))$ is defined at a mesh point $\mathbf{x} \in \mathbb{R}^3$. The branched tree $\mathcal{B}$ defined from the centerlines is made of $W = 7$ branches. The initial surface mesh $\mathcal{T}$ has been cut by disks at four different locations (red circles) being either the tree bifurcations or locations along the long branches (such as branch b_7).

of the abscissa and the normal: $(\mathbf{e}_t(\mathbf{x}), \mathbf{e}_n(\mathbf{x}), \mathbf{e}_{bn}(\mathbf{x}))$. The local reference system is computed as follows: for a point $\mathbf{x}$ in the volume to be remeshed (i) compute, by using a fast search tree such as ANN, the two closest points on the centerlines $\mathbf{x}_{c1}$ and $\mathbf{x}_{c2}$ so that $\mathbf{e}_t = \mathbf{x}_{c2} - \mathbf{x}_{c1}$; (ii) compute the normal $\mathbf{e}_n = \mathbf{x} - \mathbf{x}_{c1}$; and (iii) compute the binormal as the cross product of the abscissa and the normal. Figure 2 shows an example of the local radius $r(\mathbf{x}_c)$ for a point on the centerline and of a local reference system for a point $\mathbf{x}$ in the volume of an aortic arch. The local reference system is uniquely defined. However, in the neighborhood of bifurcations, there will be abrupt changes in the orientations of the axes. As will be explained in the next sections, the local reference system enables us to define an anisotropic mesh metric to produce anisotropic meshes. The fact that there exist abrupt changes at bifurcations is not really an issue for the anisotropic mesh generator.

2.3. *Centerline-based operators in Gmsh*

Besides the geometrical descriptors, three centerline-based operators are defined:

- A cut operator.
- A close-volume operator.
- A vessel wall model generation.

The cut operator can cut the initial tubular geometry $\mathcal{T}$ into different mesh patches S_j of moderate geometrical aspect ratio $\eta = H/D$, that is, the ratio between the height H and the diameter D of the tube. As an example, Figure 2 shows 5 mesh patches, and Figure 4 shows 17 mesh patches. The centerline-based operator is very important for the surface remeshing of the initial triangulation. Indeed our surface (re)-meshing algorithm works as follows:

1. We compute a conformal mapping $\mathbf{u}(\mathbf{x})$ that maps a surface patch S_j into a 2D surface S'_j embedded in $\mathbb{R}^2$ and that preserves (in a least square sense) the angles between the iso-u and iso-v lines:

$$\mathbf{x} \in \mathsf{S}_j \subset \mathbb{R}^3 \mapsto \mathbf{u}(\mathbf{x}) \in \mathsf{S}'_j \subset \mathbb{R}^2. \tag{1}$$

DOI: 10.1002/cnm

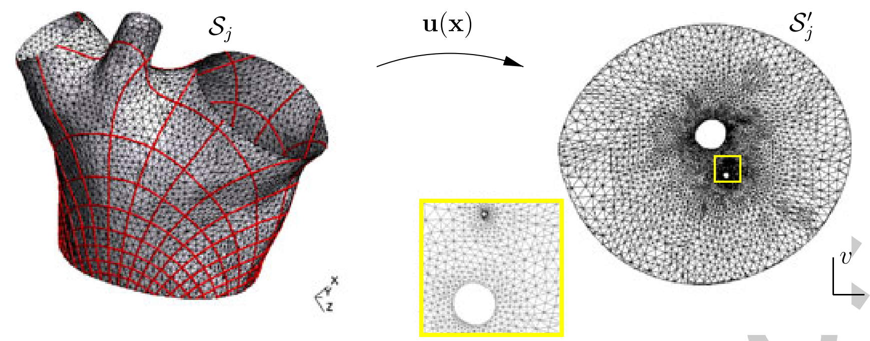


Figure 3. A conformal mapping $\mathbf{u}(\mathbf{x}) = \{u(\mathbf{x}), v(\mathbf{x})\}$ that transforms continuously a 3D surface $\mathsf{S}_j \in \mathbb{R}^3$ (a tri-bifurcation) into a 2D surface S'_j embedded in $\mathbb{R}^2$. The red lines on the 3D surface are the iso-u and iso-v lines.

The conformal mapping $\mathbf{u}(\mathbf{x})$ is the solution of a $2M \times 2M$ linear system [19–22] where M is the number of surface mesh vertices of the initial triangulation. Figure 3 shows the computed conformal mapping for a triangulated surface of a tri-bifurcation. As can be seen in Figure 3, the triangles that are close to the three outlets of the tri-bifurcation becomes very small in the parametric space. We have shown in [22] that the higher the aspect ratio of the geometry, the closer the mesh vertices nearby the outlets in the parametric space. Moreover, we showed that a necessary condition for avoiding indistinguishable mesh vertices in the parametric space is to ensure that the geometrical aspect ratio remains smaller than $\eta = 4$. This is why our cut algorithm is of importance as it guarantees that the patches S_j are of moderate aspect ratio ($\eta < 4$).

2. The initial surface patch S_j can then be remeshed in the parametric space by using any 2D mesh generation procedure with a given variable isotropic mesh size field or an anisotropic mesh metric.
3. The new mesh is then mapped back to the original surface.

The cut operator relies on a branched tree structure of the centerlines. A branched tree is a set of W branches $\mathcal{B} = \{b_1, \ldots, b_W\}$ that know their children branches (see the branched tree structure of the lung made of $W = 305$ branches in Figure 1(b) or the branched tree structure of the aorta made of $W = 7$ branches in Figure 2). The variables of a branch b_j are its length, the minimum and maximal local radius $r(\mathbf{x}_{cj})$ of all mesh vertices of the branch $\mathcal{X}_c^{b_j} = \{\mathbf{x}_{c1}, \ldots, \mathbf{x}_{cJ}\}$, and the children branches $\mathcal{B}^{\text{child}} \subset \mathcal{B}$.

By using the branched tree, the cut algorithm is defined as follows: we loop over all the branches of the tree and cut the mesh by a disk at all the tree bifurcations (junction between the parent and the children branches) and at different lengths of the branch such that the maximal aspect ratio of a branch never exceeds $\eta = 4$. The cutting disk is defined by the position of the cut (vertex $\mathbf{x}_c^{\text{cut}}$), the radius of the vessel at the cut $r(\mathbf{x}_c^{\text{cut}})$ and the unit vector along the direction of the branch at the cut $\mathbf{e}_t(\mathbf{x}_c^{\text{cut}})$. The radius of the cutting disk R is chosen a little bit larger than the vessel radius $R = 1.2r(\mathbf{x}_c^{\text{cut}})$ to guarantee an exact cut of the mesh. The triangles of the mesh $\mathcal{T} = \{T_1, \ldots, T_N\}$ that are cut by the disk are divided into three subtriangles. Figure 4 shows an example of an initial STL mesh $\mathcal{T}$ that has been cut by 16 disks by using the centerline-based cut operator. The resulting mesh contains 17 different surface patches S_j than can be remeshed using the remeshing techniques based on parametrizations. Figure 4(b) shows the conformal parametrization of the white mesh patch.

The second operator is the close-volume operator. This operator creates planar faces at the inlet and outlet of the tubular geometry. The planar faces are defined by the mean plane of the mesh vertices $\mathbf{x} \in \mathcal{X}$ located at the boundaries of the input geometry. Those faces can then be subsequently meshed by the planar mesh generators.

DOI: 10.1002/cnm

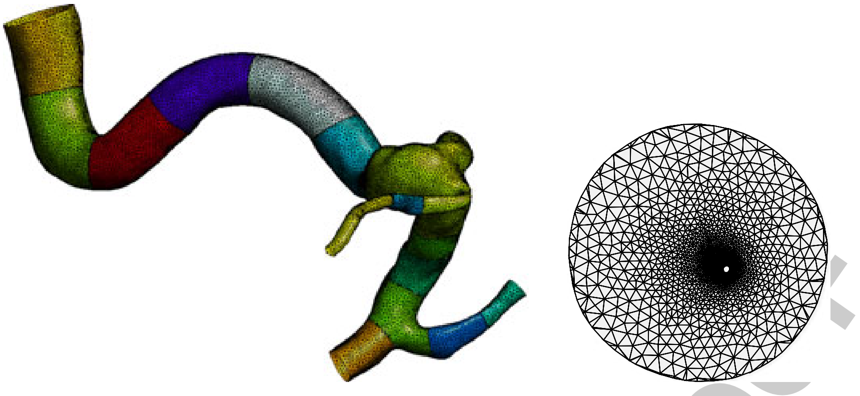


Figure 4. The cut operator on the basis of the centerlines has cut the initial tubular geometry, a cerebral aneurysm, into 17 different mesh surfaces S_j (a). For the remeshing of those patches, a conformal parametrization $\mathbf{u}(\mathbf{x})$ is computed for every mesh patch. The parametrization is shown for the white mesh patch (b).

The last defined operator is a vessel wall model generation. In many cases, the entire wall surface is not available from image segmentation and needs to be reconstructed from the lumen wall by using, for example, a quite realistic radius-dependent wall-thickness δ^W such as a percentage of the local radius of the tubular geometry. The geometry of the wall surface can then be obtained by extruding the lumen surface mesh in the outward direction with a radius-dependent wall thickness. The extrusion is performed in Gmsh by using an advancing layer method [23–25] with a given number of layers. The method starts from a surface mesh on which a boundary layer must be grown. From each surface node $\mathbf{x} \in \mathcal{X}$ a direction is picked for placing the nodes of the boundary layer mesh. The direction is computed using an estimate to the surface normal at the node by using Gouraud shading[26], and the extrusion thickness δ is computed as a percent α of the vessel radius:

$$\delta^W(\mathbf{x}) = \alpha r\left(\mathbf{x}_c\left(\mathbf{x}\right)\right), \tag{2}$$

where $\mathbf{x}_c \in \mathcal{X}_c$ is the mesh vertex on the centerlines that is closest to $\mathbf{x}$. If the surface mesh is a quadrangular mesh, the nodes are connected to form layers of hexahedra, and in the case it is a triangular mesh, the nodes are connected to form layers of prisms that are subsequently subdivided into tetrahedra. An example of vessel wall generation is shown in Figure 5. F5

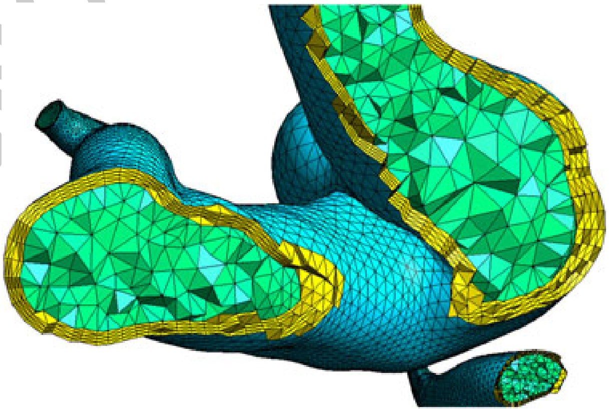


Figure 5. Vessel wall model generation for the geometry of cerebral aneurysm: cut-away view of the tetrahedral volume mesh of both the lumen (in green) and the vessel wall (in yellow). The vessel wall is built using a radius-dependent wall thickness $\delta^W(\mathbf{x})$ with $\alpha = 0.2$ and four layers.

DOI: 10.1002/cnm

2.4. Defining mesh metrics for anisotropic meshes

It has been shown in Section 2.2 how to compute a local reference system on the basis of the centerlines $(\mathbf{e}_t, \mathbf{e}_n, \mathbf{e}_{bn})$. For simple tubular geometries, this local reference system is such that the normal vector $\mathbf{e}_n$ is close to the surface normal $\mathbf{n}$ and that the tangent vector $\mathbf{e}_t$ is close to the eigenvector $\mathbf{t_1}$ corresponding to the smallest eigenvalue of the curvature tensor $\mathbb{W}$ computed at the surface. However, at geometric singularities (singular points of the centerlines such as at the bifurcations), this is not always the case. This problem is addressed by taking two different local reference systems: close to centerlines the centerline-based local reference system is chosen $(\mathbf{e}_t, \mathbf{e}_n, \mathbf{e}_{bn})$, and in the vicinity of the surface (in a CFD boundary layer of thickness δ), a local reference system is computed from the eigenvectors of the curvature tensor $(\mathbf{t_1}, \mathbf{t_2}, \mathbf{n})$. Here, the curvature tensor is computed per face on the discrete tubular surface by using the method proposed by S. Rusinkiewicz [27]. The method of Rusinkiewicz computes first the vertex curvatures by taking a weighted average of the adjacent faces' normal vectors. The curvatures per face are then defined in terms of the directional derivative of the surface points and their normals. The minimal normal curvature κ_1 and the maximal normal curvature κ_2 (also called principal curvatures) are obtained as eigenvalues of the curvature tensor $\mathbb{W}$. The associated eigenvectors $\mathbf{t}_1$ and $\mathbf{t}_2$ are the two principal tangent directions that are always perpendicular to each other. The normal vector $\mathbf{n}$ is the cross-product of $\mathbf{t}_1$ and $\mathbf{t_2}$.

To create a 3D anisotropic computational mesh from an input geometry, we need to define three mesh size fields h_i in the axis directions of the chosen local reference system. The following mesh metric $\mathcal{M}(\mathbf{x})$ can then computed:

$$\mathcal{M}(\mathbf{x}) = \mathbf{R}^T \begin{pmatrix} h_1^{-2} & 0 & 0 \\ 0 & h_2^{-2} & 0 \\ 0 & 0 & h_3^{-2} \end{pmatrix} \mathbf{R}, \tag{3}$$

where $\mathbf{R}$ is a matrix whose columns are the vectors of the reference system. This mesh metric can then be used by the different 1D, 2D, and 3D anisotropic meshing algorithms.

There are several criteria than can be applied to assign the different mesh sizes at each point $\mathbf{x}$. Let us first look at the mesh metric in the vicinity of the surface.

- If one is willing to build a CFD boundary layer mesh inside the lumen inside the lumen, the normal mesh size can be computed as [28]:

  $$h_n(\mathbf{x}) = h_n^0 + d(\mathbf{x}) \ln \beta, \tag{4}$$

  where $d(\mathbf{x})$ is the distance to the tubular geometry $\mathcal{T}$ that is again computed efficiently by using an ANN [17, 18], $\beta > 1$ is the normal growth of the boundary layer, that is, the ratio between two successive element sizes in the normal direction and h_n^0 is the normal size at the wall. This normal size can be for example a function of the vessel radius at the wall: $h_n^0 = \alpha r(\mathbf{x}_c)$.
- Some control on the geometrical accuracy of the mesh should be ensured. For that, mesh sizes h_{t1} and h_{t2} should depend on the curvature of the surface in directions $\mathbf{t_1}$ and $\mathbf{t_2}$. Typically, we choose for the mesh points on the wall:

  $$h_{t1}^0 = \frac{2\pi\rho_{t1}^0}{N_p} \quad \text{and} \quad h_{t2}^0 = \frac{2\pi\rho_{t2}^0}{N_p} \tag{5}$$

  with N_p a control parameter that is the number of mesh points per circumference and $\rho_{t1} = 1/\kappa_1$ and $\rho_{t2} = 1/\kappa_2$ the radii of curvature of the surface in both directions $\mathbf{t_1}$ and $\mathbf{t_2}$. When we leave the surface, the tangent mesh size is allowed to grow as follows:

  $$h_{t1}(\mathbf{x}) = h_{t1}^0 + d(\mathbf{x}) \ln \beta. \tag{6}$$

However, we will show that a more subtle control (more subtle than Equation (5)) on the tangent mesh size fields has to be performed to avoid large variations of mesh sizes. Let us consider a 2D geometrical domain Ω of boundary Γ with radius of curvature ρ^0 (see Figure 6). A boundary layer mesh of thickness δ is constructed with a prescribed 'normal to the wall' mesh size h_n

DOI: 10.1002/cnm

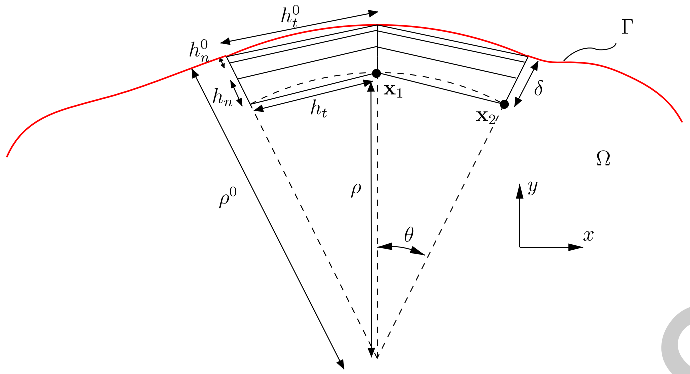


Figure 6. Defining the mesh sizes h_t and h_n on a 2D geometry of boundary Γ with the radius of curvature ρ_0. The boundary layer thickness δ is sufficiently thin so that $\rho \approx \rho_0$ in the boundary layer.

(Equation (4)). The boundary layer thickness δ is sufficiently thin to $\rho \approx \rho_0$. We raise here the following question: is it possible to choose freely the 'tangent to the wall' mesh size h_t^0? In most mesh generation procedures, a smoothing smoothing step is applied to the metric field to avoid large variation of mesh sizes. A smoothing ratio $\beta \geqslant 1$ is defined as the maximum ratio of two adjacent edge lengths. Let us demonstrate in 2D that mesh size $h_t(\mathbf{x})$ is indeed constrained by β, h_n, and ρ_0 and should therefore not be given using (5). In what follows, we will drop the superscript 0 for denoting the tangent h_t^0 and the normal h_n^0 mesh size to the wall.

At point $\mathbf{x}_1$, the 2D metric field can be written as

$$\mathcal{M}(\mathbf{x}_1) = \begin{pmatrix} h_t^{-2} & 0 \\ 0 & h_n^{-2} \end{pmatrix}.$$

At point $\mathbf{x}_2$, the metric is rotated at an angle θ:

$$\begin{aligned} \mathcal{M}(\mathbf{x}_2) &= \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} h_t^{-2} & 0 \\ 0 & h_n^{-2} \end{pmatrix} \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \\ &= \frac{1}{h_n^2 h_t^2} \begin{pmatrix} h_n^2\cos^2\theta + h_t^2\sin^2\theta & \left(h_n^2 - h_t^2\right)\cos\theta\sin\theta \\ \left(h_n^2 - h_t^2\right)\cos\theta\sin\theta & h_t^2\cos^2\theta + h_n^2\sin^2\theta \end{pmatrix}. \end{aligned} \tag{7}$$

Assuming $\theta = h_t/\rho_0 \ll 1$, we have

$$\cos^2\theta \simeq 1 - \frac{h_t^2}{\rho_0^2} \quad \text{and} \quad \sin^2\theta \simeq \frac{h_t^2}{\rho_0^2}.$$

and the mesh size h_x in the x direction at point $\mathbf{x}_2$ is computed as

$$\frac{1}{h_x^2} = \frac{1}{h_n^2 h_t^2}\left(h_n^2\cos^2\theta + h_t^2\sin^2\theta\right) \simeq \frac{1}{h_t^2} + \frac{1}{\rho_0^2}\left(\frac{h_t^2}{h_n^2} - 1\right).$$

It is now possible to correct mesh h_t in direction x at point $\mathbf{x}_1$ by using the relation $h_t = \beta h_x$, that is,

$$\frac{\beta^2}{h_t^2} = \frac{1}{h_t^2} + \frac{1}{\rho_0^2}\left(\frac{h_t^2}{h_n^2} - 1\right). \tag{8}$$

Providing that $\frac{1}{h_t^2}$ is positive, (8) has the unique solution

$$\frac{1}{h_t^2} = \frac{1}{2\rho_0^2(\beta^2 - 1)}\left(\sqrt{1 + \frac{4\rho_0^2(\beta^2 - 1)}{h_n^2}} - 1\right). \tag{9}$$

Copyright © 2013 John Wiley & Sons, Ltd.

DOI: 10.1002/cnm

Mesh size h_t is then an upper bound of the mesh size in the tangent direction: for example, if ρ is very large, then the tangent mesh size can be chosen arbitrarily. If $\beta = 1$, then Equation (8) has the solution $h_t = h_n$, which means that no anisotropy is possible in a uniform mesh.

Let us now look at definition of the mesh sizes outside the boundary layer, where we would like to have mesh elements aligned with the centerlines. A mesh size field h_t in the direction $\mathbf{e}_t$ is defined that grows linearly from h^0_{t1} at the surface boundary to h_c at the center of the tubular geometry. The mesh size field in the two other directions $(\mathbf{e}_n, \mathbf{e}_{bn})$ are given by Equation (4).

By using the anisotropic mesh metric (3), we can compute the length of a path given by a mesh edge $\mathbf{e} = \overline{\mathbf{x}_i \mathbf{x}_j}$ by using the straight line parametrization $\mathbf{x}(t) = \mathbf{x}_i + t\, \overline{\mathbf{x}_i - \mathbf{x}_j}$, $t \in [0, 1]$:

$$l_{\mathcal{M}} = \int_0^1 \|\gamma(t)\| \mathrm{d}t = \int_0^1 \sqrt{\mathbf{e}^T \mathcal{M}(\mathbf{x}(t)) \mathbf{e}}\, \mathrm{d}t. \tag{10}$$

For the surface remeshing of the tubular geometry, we however do not remesh in the 3D space but remesh in the parametric space thanks to the conformal parametrization (Figure 3). This enables us to use any available 2D meshers and in particular the 2D anisotropic mesh generator bidimensional anisotropic mesh generator [29] developed by F. Hecht and integrated within Gmsh. The length of the edge on the mesh patch S_j can then be computed from the length in the parametric space $\mathbf{e}'$ as follows: $\mathbf{e} = \mathbf{x}_{,\mathbf{u}} \mathbf{e}'$ so that the length of the edge with respect to the mesh metric can be computed as follows:

$$l_{\mathcal{M}_\mathbf{x}} = \int_0^1 \sqrt{\mathbf{e}'^T \underbrace{\left(\mathbf{x}^T_{,\mathbf{u}} \mathcal{M}(\mathbf{x}) \mathbf{x}_{,\mathbf{u}}\right)}_{\mathcal{M}(\mathbf{x})'} \mathbf{e}'}\, \mathrm{d}t, \tag{11}$$

where $\mathcal{M}(\mathbf{x})'$ is the mesh metric that needs to be given to the anisotropic 2D mesh generators. For our piecewise linear conformal mapping $\mathbf{u}(\mathbf{x})$, the derivatives $\mathbf{x}_{,\mathbf{u}}$ can be computed quite easily (see [22] for more details).

2.5. *Quadrangular and hexahedral mesh generation*

The first step for our hexahedral mesh generation algorithm is the generation of a quadrangular mesh of the initial geometry. The presented algorithm of quadrangular mesh generation is an original approach that combines a structured approach and an indirect unstructured approach. The mesh patches that are created by our centerline-based cut operator can be classified into two types of patches: annular patches and bifurcation patches. The annular mesh patches are then remeshed using a structured approach, whereas the bifurcation patches (tri-bifurcations or higher order bifurcations) are meshed with an indirect approach.

The structured meshes are built using an elliptic mesh generation method [5]. The elliptic grid generator relies on the conformal mapping $\mathbf{u}(\mathbf{x})$ (Equation (1)) of the physical domain onto the parametric domain and on a second mapping $\mathbf{u}(\boldsymbol{\xi})$ that maps a periodic rectangular computational domain onto the parametric domain (see Figure 7). F7

The different steps involved in the structured quadrangular mesh generator of an annular patch are the following:

1. Create N mesh vertices on the edges of the model, that is, the inlet ring e_1 and the outlet ring e_2 (see the the black dots in the right Figure 7). The number of mesh vertices on a ring is a user-defined parameter.
2. Create a regular grid in the parametric domain $\mathbf{u}(\mathbf{x})$ (see the green dots in the parametric domain). To create a regular grid, we choose an arbitrarily vertex v_1 on e_1 and find v_0, the closest vertex to v_1 on e_2 in the parametric domain. Then, compute the number M of mesh vertices for edge e_0 as follows: $M = \left|e^{\mathbf{x}}_0\right| / (2\pi r/N)$, where $\left|e^{\mathbf{x}}_0\right|$ is the length of edge e_0 in the physical domain.[¶] Create in the parametric domain M uniformly distributed mesh vertices on edge e_0 and repeat this operation for the other edges joining the vertices of edges e_1 and e_2.

[¶] This length is computed by numerical integration.

DOI: 10.1002/cnm

Color Online, B&W in Print

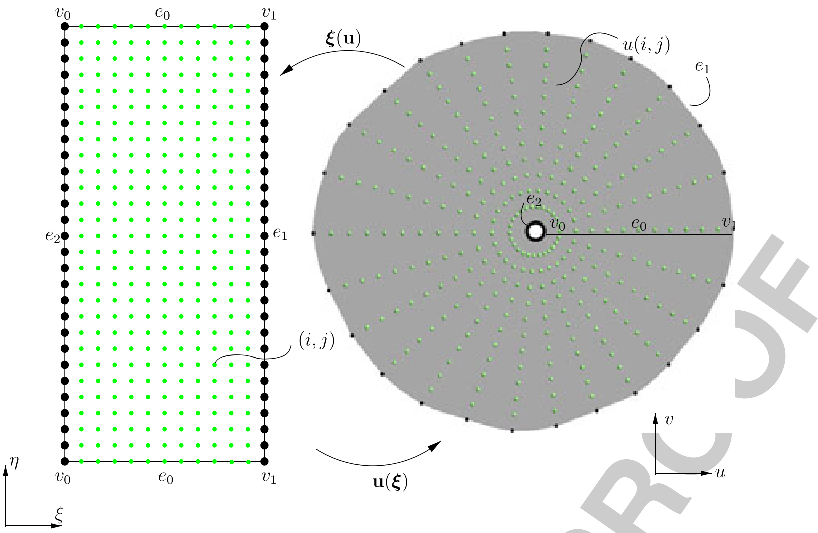


Figure 7. Structured mesh generation for annular patches for which the mapping $\mathbf{u}(\mathbf{x})$ has already been computed. The elliptic mesh generation method relies on the computation of a second mapping $\mathbf{u}(\boldsymbol{\xi})$ that maps a regular computational domain to the parametric domain.

F8

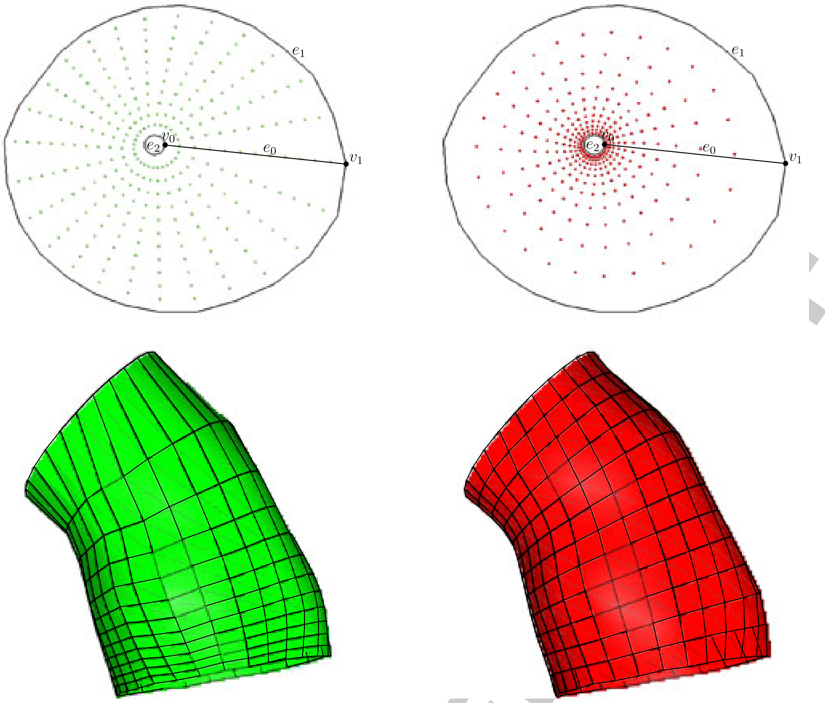
3. Since the mapping $\mathbf{u}(\mathbf{x})$ does not preserve the lengths, the new vertices which are uniformly distributed in the parametric space, are not uniformly distributed in the physical domain (see bottom left Figure 8). The idea is then to move the new created mesh vertices in the parametric domain (u, v) in such a way that those mesh vertices are uniformly distributed in the physical domain (see right Figure 8). Such locations of mesh vertices can be found by computing a second mapping $\boldsymbol{\xi}(\mathbf{u})$ that maps the parametric domain onto a periodic rectangular domain of size $M \times N$. The mapping can be computed by solving the two following Laplace (elliptic) PDEs:

$$\frac{\partial^2 \xi}{\partial u^2} + \frac{\partial^2 \xi}{\partial v^2} = 0, \qquad \frac{\partial^2 \eta}{\partial u^2} + \frac{\partial^2 \eta}{\partial v^2} = 0 \tag{12}$$

with appropriate Dirichlet and Neumann boundary conditions.
4. The problem (12) is solved using finite differences in the space of (ξ, η), and the unknowns are the positions (u, v) in the parametric space. The resulting nonlinear system is solved iteratively by point-Jacobi iterations. For the first iteration ($k = 0$), the positions $\left(u^0_{i,j}, v^0_{i,j}\right)$ are the uniformly distributed parametric points. For each point-Jacobi iteration, the new positions can be found as follows (see [30] for more details):

$$\begin{aligned}
\alpha &= \frac{1}{4}\left(\left(u_{i,j+1} - u_{i,j-1}\right)^2 + \left(v_{i,j+1} - v_{i,j-1}\right)^2\right)\\
\gamma &= \frac{1}{4}\left(\left(u_{i+1,j} - u_{i-1,j}\right)^2 + \left(v_{i+1,j} - v_{i-1,j}\right)^2\right)\\
\beta &= \frac{1}{16}\left(\left(u_{i+1,j} - u_{i-1,j}\right)\left(u_{i,j+1} - u_{i,j-1}\right) + \left(v_{i+1,j} - v_{i-1,j}\right)\left(v_{i,j+1} - v_{i,j-1}\right)\right)\\
u^k_{i,j} &= \frac{1}{2(\alpha+\gamma)}\left(\alpha(u_{i+1,j} + u_{i-1,j}) + \gamma(u_{i,j+1} + u_{i,j-1})\right.\\
&\qquad \left. -2\beta(u_{i+1,j+1} - u_{i-1,j+1} - u_{i+1,j-1} + u_{i-1,j-1})\right)\\
v^k_{i,j} &= \frac{1}{2(\alpha+\gamma)}\left(\alpha(v_{i+1,j} + v_{i-1,j}) + \gamma(v_{i,j+1} + v_{i,j-1}\right.\\
&\qquad \left. -2\beta(v_{i+1,j+1} - v_{i-1,j+1} - v_{i+1,j-1} + v_{i-1,j-1}\right).
\end{aligned} \tag{13}$$

DOI: 10.1002/cnm



(a) (b)

Color Online, B&W in Print

Figure 8. Elliptic grid generation of a tubular mesh patch (the mesh patch corresponds to the blue mesh patch of the aorta presented in Figure 2). (a) Initial structured grid in parametric domain (top) and in physical domain (bottom); and (b) final mesh points and quad mesh obtained with the elliptic smoother in both parametric and physical domain.

5. At convergence, the parametric points $\left(u^k_{i,j}, v^k_{i,j}\right)$ that are mapped (by using the mapping $\mathbf{x}(\mathbf{u})$) to the physical domain are uniformly distributed and can be used to create regular quadrangles. Figure 8(b) shows after 100 iterations the final points (red points) in the parametric space as well as the resulting mapped mesh in the physical space.

The indirect way of producing quad meshes comprises three steps: (i) the triangulation is tailored with the aim of producing right triangles in the domain by using the infinity norm to compute distances in the meshing process [6]; (ii) the triangles are recombined into quads by using the well-known Blossom algorithm of graph theory that computes the minimum cost perfect matching in a graph in polynomial time [7]; and (iii) local and global mesh cleanup operations are performed, such as Guy Bunin's one-defect remeshing operation [31] to reduce irregular nodes in the mesh.

Once the quadrangular surface mesh has been generated, we generate an extruded boundary layer mesh as well as an unstructured tetrahedral mesh of the lumen with pyramids as transition elements [8]. We are currently also working on an optimal placement of the mesh vertices in the lumen for the recombination of the tetrahedra into hexahedra [9–11], which will allow to also generate hexahedral dominant meshes.

3. EXAMPLES

In this section, we have run our new automatic meshing algorithm on different medical tubular geometries including cerebral aneurysm, carotid arteries, and airways. Different types of computational meshes are generated: isotropic tetrahedral meshes, anisotropic tetrahedral meshes, or mixed meshes. Some of the meshes include a vessel wall.

Timings as well as mesh qualities are given for the different meshes. The computations are performed on a MacBook Pro 2.66 GHz 4 GB RAM Intel Core i7. Q3

DOI: 10.1002/cnm

Table I. Mean and minimum quality ($\bar{\gamma}_T$, $\bar{\gamma}_\tau$, and $\gamma_\tau^{\min}$), number of mesh elements # and timings (in s) for the generation of isotropic tetrahedral meshes starting from tubular geometries.

	STL		Surface mesh		Volume mesh			Time (s)	Time (s)	Time (s)
Geometry	#	$\bar{\gamma}_T$	#	$\bar{\gamma}_T$	#	$\gamma_\tau^{\min}$	$\bar{\gamma}_\tau$	$\mathcal{C}$ field	2D mesh	3D mesh
Aorta	4 10^3	0.73	12 10^3	0.97	58 10^3	0.21	0.69	0.08	0.78	2.26
Aneurysm	38 10^3	0.94	27 10^3	0.97	104 10^3	0.19	0.65	3.30	3.60	4.20
Airways	493 10^3	0.87	168 10^3	0.93	587 10^3	0.06	0.68	410.10	25.90	35.11

STL, standard triangulated language.

3.1. *Isotropic tetrahedral computational meshes*

Let us first look at isotropic tetrahedral computational meshes. We define two quality criteria: one quality γ_T for the triangles T of the surface mesh [32] and one quality γ_τ for the tetrahedra τ of the volume [32, 33].

$$\gamma_T = \alpha \frac{\rho_{\text{in}}}{\rho_{\text{out}}} \tag{14}$$

$$\gamma_\tau = \frac{2\sqrt{6}\,\rho}{h}. \tag{15}$$

Here α is a constant, ρ_{in} is the radius of the inner circle/sphere of the mesh element, ρ_{out} is the radius of the circumscribed circle/sphere of the element and h is the length of the longest edge of the tetrahedron. With those definitions, the regular triangle/tetrahedron has $\gamma = 1$ and the degenerated (zero surface/volume) mesh element has $\gamma = 0$.

T1 Table I shows the mean triangle and tetrahedra quality ($\bar{\gamma}_T$, $\bar{\gamma}_\tau$), the minimum tetrahedral quality ($\gamma_\tau^{\min}$), the number of mesh elements # and the timings (in s) for the generation of different isotropic tetrahedral meshes starting from tubular geometries (STL triangulations). The tubular geometries are the aorta shown in Figure 2, the cerebral aneurysm shown in Figure 4 and the airways in Figure 1. The meshes are obtained using the 2D frontal Delaunay[‖] algorithm [34] and the 3D Delaunay algorithm. The timing for the computation of the centerline field (i.e., the timing for the computation of the centerline-based descriptors and operators described in Section 2) is given. For isotropic tetrahedral meshes, the only computed descriptor is the local radius, and the two computed operators are the cut operator and the close-volume operator. The timings are also given for the 2D and 3D mesh generation (including mesh optimization).

F9 Figure 9 shows the generated isotropic tetrahedral mesh for the lung on the basis of the centerline field. As can be seen, the mesh size is a function of the vessel radius, reducing therefore considerably the total number of mesh elements compared with a uniform tetrahedral mesh. ~~(Figure 10)~~ F10 Q4

3.2. *Mixed hexahedral/tetrahedral/pyramidal computational meshes*

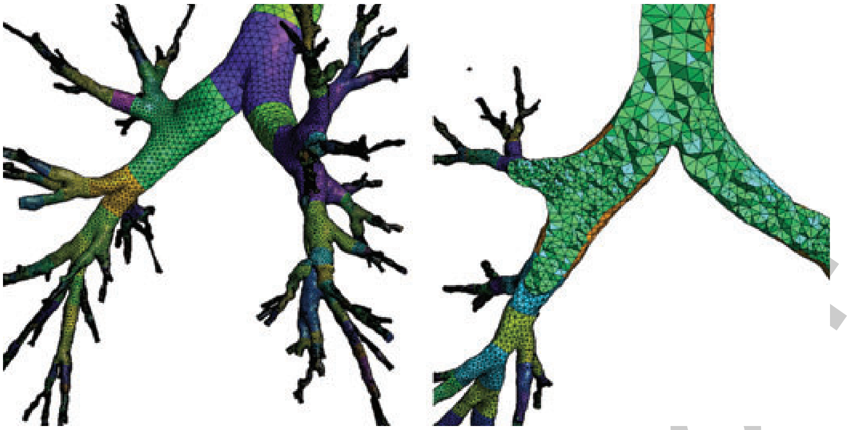
We first define a quality measure for quadrilateral elements. Consider a quadrilateral element q and its four internal angles α_k, $k = 1, 2, 3, 4$. We define the quality η_q as follows:

$$\eta_q = \max\left(1 - \frac{2}{\pi}\max_k\left(\left|\frac{\pi}{2} - \alpha_k\right|\right), 0\right). \tag{16}$$

This quality measure is $\eta = 1$ if the element is a perfect quadrilateral and is $\eta = 0$ if one of those angles is either $\leqslant 0$ or $\geqslant \pi$. For the hexahedral mesh elements, we define the equi-skew angle mesh quality ζ_H as a normalized measure of skewness ranging from $\zeta_H = 1$ (best) to $\zeta_H = 0$ (worst)

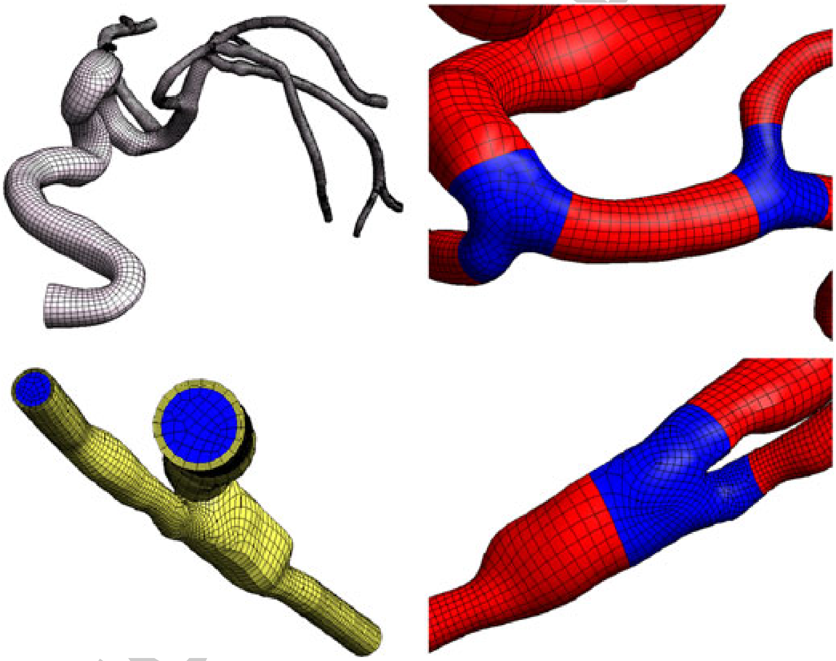
[‖] It was indeed shown in [21] that optimal isotropic tetrahedral meshes (i.e., with the highest mesh quality) can be obtained by combining a conformal parametrization with a frontal mesher.

DOI: 10.1002/cnm


(a) (b)

Color Online, B&W in Print

Figure 9. Isotropic tetrahedral mesh of the airways created using the centerline operators. The colors correspond to the different mesh patches that have been created by the cut operator.



Color Online, B&W in Print

Figure 10. Quadrangular surface mesh of the cerebral arterial tree and hexahedral mesh of the arterial wall (in yellow) of the carotid bifurcation. The blue mesh patches (tri-bifurcations or higher-order bifurcations) on the right figures correspond to the quad meshes obtained using the indirect approach, whereas the red patches correspond to the direct quad approach.

that depends on the angle formed between the faces's edges of each cell in the mesh ($\zeta_H = 1$ corresponds to a perfectly equiangular hexahedra) [12]:

$$\zeta_H = 1 - \frac{2}{\pi} \max\left[\left(\theta_{\max} - \frac{\pi}{2}\right), \left(\frac{\pi}{2} - \theta_{\min}\right)\right], \qquad (17)$$

where $\theta_{\max}$ and $\theta_{\min}$ are the largest and smallest angle in the hexahedra.

Table II shows the minimum and quality ($\eta_q^{\min}$, $\bar{\eta}_q$, $\zeta_H^{\min}$, and $\bar{\zeta}_H$), the number of mesh elements # and the timings (in s) for the generation of the mixed hexahedral/tetrahedral meshes starting from different tubular geometries. The parameters for the extruded hexahedral mesh are four layers of total thickness $\delta_W = 0.2r(\mathbf{x}_c)$. For the volume elements, we only show the statistics for the T2

DOI: 10.1002/cnm

Table II. Minimum and mean quality ($\eta_q^{\min}$, $\bar{\eta}_q$, $\bar{\zeta}_H^{\min}$, and ζ_H), number of mesh elements # and timings (in s) for the generation of mixed hexahedral/tetrahedral meshes starting from tubular geometries.

	Quad surface mesh			Hex volume mesh			Time (s)	Time (s) ă	Time (s) 3D
Geometry	#	$\bar{\eta}_q^{\min}$	$\bar{\eta}_q$	#	$\zeta_H^{\min}$	$\bar{\zeta}_H$	2D mesh	3D hex mesh	mixed mesh
Aorta	$3\ 10^3$	0.21	0.85	$11\ 10^3$	0.28	0.83	7.9	0.6	2.3
Carotid	$5\ 10^3$	0.22	0.91	$16\ 10^3$	0.33	0.86	4.3	0.3	2.5
Cerebral	$27\ 10^3$	0.19	0.89	$99\ 10^3$	0.28	0.85	29.2	2.5	20.0

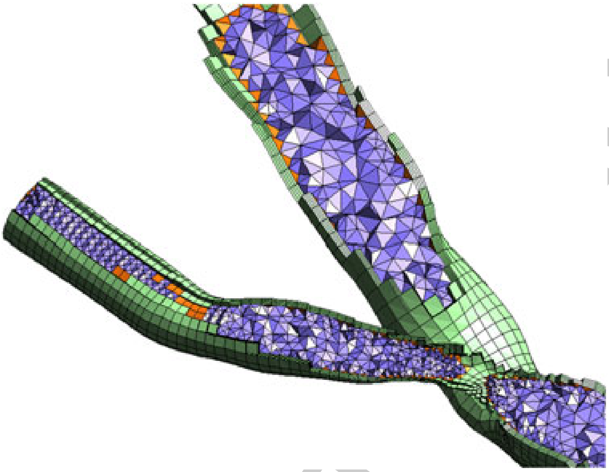


Figure 11. Mixed mesh of the carotid geometry. Hexahedra are in green, tetrahedra are in blue, and pyramids are in orange. The mesh is composed of 58,046 tetrahedra, 16,500 hexahedra, and 4,236 pyramids.

hexahedra of the mixed meshes. The timings presented in Table II are quite fast compared with the timings for the generation of block-structured hexahedral meshes (timings of several hours to generate about 5000 tetrahedra in [12]).

Mixed three dimensional meshes can be generated that are composed of a mixture of tetrahedra, hexahedra, and pyramids. Figure 11 shows an image of a hybrid 3D mesh of the carotid.

3.3. Anisotropic tetrahedral computational meshes

The anisotropic meshes are built using first a 2D and then a 3D anisotropic mesh procedure. From the initial triangulation, we first generate a new isotropic triangular surface mesh by using a conformal mapping of the surface mesh (see Equation (1)). Then, given the mapped surface mesh and the discrete metric tensor (11), the anisotropic surface mesh is generated in the parametric space by the bidimensional anisotropic mesh generator [29] integrated within Gmsh. Then, from this anisotropic surface mesh, an initial volume Delaunay tetrahedral mesh is created using TetGen [35] without adding any new points on the surface mesh during the tetrahedralization. The initial tetrahedral mesh is then given as input to mmg3d [36, 37], a 3D Delaunay-based anisotropic mesh adaptation library. This library, also integrated in Gmsh, produces quasi-uniform meshes with respect to a metric tensor field by using local mesh modifications and a Delaunay kernel to adapt the initial mesh.

Given the metric tensor $\mathcal{M}$ (Equation (3)), we define the following criteria [37] that measures how well an anisotropic tetrahedron matches the metric specification, both in terms of size (edge lengths) and of shape (aspect ratio):

$$Q_\tau = \frac{\left(\sum_{i=0}^{6} \mathbf{e}_i^T \bar{\mathcal{M}} \mathbf{e}_i\right)^{3/2}}{V_\tau} \sqrt{det(\bar{\mathcal{M}})}. \tag{18}$$

Table III. Mean and minimum quality ($\bar{Q}_\tau, Q_\tau^{\min}$), efficiency index τ, number of mesh elements # and timings (s) for the generation of anisotropic tetrahedral meshes starting from the tubular geometries.

	Volume mesh					Time (s)	
Geometry	#	$\bar{Q}_\tau$	$Q_\tau^{\min}$	$1 < Q_\tau < 3(\%)$	τ	2D mesh	3D mesh
Aorta	323 10^3	1.6	11	0.98	0.836	7.2	96
Aneurysm	477 10^3	1.5	17	0.96	0.821	17	145
Bypass	342 10^3	1.5	12	0.97	0.829	11	67

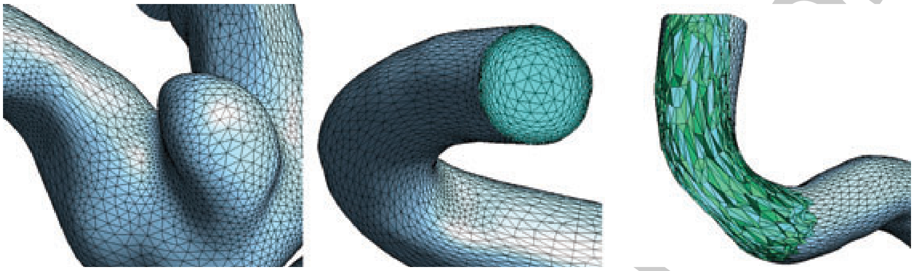


Figure 12. Anisotropic tetrahedral mesh of the aneurysm.

Color Online, B&W in Print

Here, V_τ is the volume of the tetrahedron, $\mathbf{e}_i$ are the unit vectors along the edges of the tetrahedron, and $\bar{\mathcal{M}}$ is the average metric of the tetrahedron computed from the metric at the four vertices of the tetrahedron:

$$\bar{\mathcal{M}} = \frac{1}{4}\left(\sum_{i=1}^{4} \mathcal{M}_i^{-1}\right)^{-1}.$$

With this definition $Q_\tau \in [1, +\infty]$, with a high quality value indicating a degenerated tetrahedron. In numerical simulations, optimal meshes should have more than 90% of the tetrahedra with a quality measure better than 3, that is, $1 < Q_\tau < 3$ [37, 38].

In addition, we define an efficiency index τ that provides a single scalar value to evaluate how well the tetrahedral mesh complies with the metric requirements:

$$\tau = \exp\left(\frac{1}{n_e}\sum_{1<i<n_e} l_{\mathcal{M}}(\mathbf{e}_i)\right), \quad \begin{cases} l_{\mathcal{M}}(\mathbf{e}_i) = l_{\mathcal{M}}(\mathbf{e}_i) - 1 & \text{if } l_{\mathcal{M}}(\mathbf{e}_i) < 1 \\ l_{\mathcal{M}}(\mathbf{e}_i) = l_{\mathcal{M}}^{-1}(\mathbf{e}_i) - 1 & \text{otherwise,} \end{cases} \tag{19}$$

where n_e denotes the total number of mesh edges, and the edge length $l_{\mathcal{M}}(\mathbf{e}_i)$ is given by Equation (10). Optimal meshes have an efficiency index τ close to the optimal value of 1. In numerical simulations, a value of $\tau > 0.80$ is considered as an acceptable lower bound [37, 38].

Table III shows the quality of different anisotropic tetrahedral meshes together with meshing timings. We present the mean quality index $\bar{Q}_\tau$, the percent of elements that have a good quality measure $1 < Q_\tau < 3$, as well as the efficiency index τ. The geometries are STL triangulations of three different artery trees: an abdominal aorta and a cerebral aneurysm described in Table I as well as an arterial bypass.[**] For the different examples, we have chosen a boundary layer thickness $\delta = 0.3r(\mathbf{x}_c)$, a mesh size normal to the wall $h_n^0 = \delta/20$, a boundary layer growth ratio $\beta = 1.2$ in the boundary layer, and $\beta = 2.8$ outside. T3

[**] This geometry has been downloaded from the Simbios web site https://simtk.org/frs/download.php?file_id=183.

DOI: 10.1002/cnm

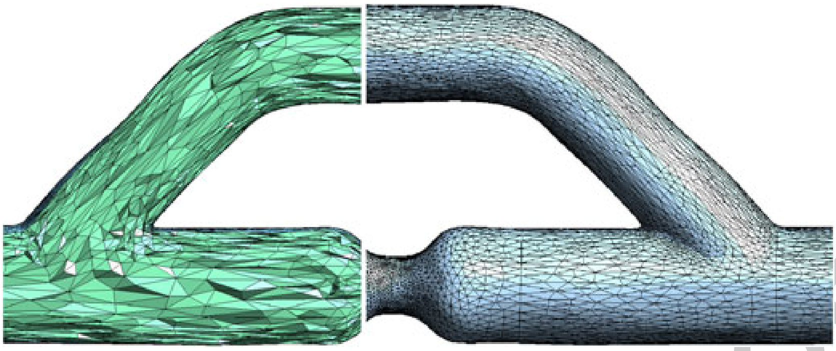


Figure 13. Magnified views of the anisotropic tetrahedral mesh of the bypass.

F12 F13

Figure 12 shows the surface and a cut of the volume mesh of the anisotropic mesh of the geometry of the aneurysm presented in the left Figure 4 and Figure 13 shows the surface and a cut of the volume mesh of the anisotropic mesh of the geometry of the bypass.

4. CONCLUSION

We have presented a new automatic meshing algorithm for the generation of computational meshes from a segmented tubular geometry. The proposed methodology is based on different centerline-based descriptors and operators.

Different types of computational meshes can be generated with this method: isotropic tetrahedral meshes, anisotropic tetrahedral meshes, or mixed hexahedral/tetrahedral meshes as well as boundary layer meshes for the lumen wall. The mesh size field is a function of the centerline-based descriptor.

Besides the original centerline-based meshing algorithms, two important contributions are included in this paper:

- A subtle control on the tangent and the normal mesh sizes for the generation of anisotropic CFD boundary layer meshes.
- A flexible and fast approach for hexahedral mesh generation. The proposed approach relies on the generation of a quadrangular surface mesh that combines a structured elliptic grid generator together with an indirect quadrangular meshing approach. Structured hexahedral meshes are then created for the vessel wall and connected to the tetrahedra of the lumen with pyramids.

We are currently working on several hexahedral meshing algorithms that will enable us to generate also dominant hexahedral meshes in a close future by using the presented method for hexahedral mesh generation.

The presented automatic meshing algorithm is implemented in the open-source mesh generator Gmsh [32] and examples can be found on the Gmsh wiki.[††]

ACKNOWLEDGEMENTS

This work has been partially supported by the Belgian Walloon Region under WIST grants ONELAB 1017086 and DOMHEX 1017074.

The authors like to thank Luca Antiga for his support with vmtk, for his constructive comments and the geometrical model of the cerebral arteries. The authors also like to thank the people from FluidDA NV for providing the geometrical model of human airways.

[††] Gmsh's wiki: `https://geuz.org/trac/gmsh` (username: gmsh, password: gmsh).

DOI: 10.1002/cnm

REFERENCES

1. Geuzaine C, Remacle J-F. Gmsh: a finite element mesh generator with built-in pre- and post-processing facilities, 1996. (Available from: http://www.geuz.org/gmsh/). Q5
2. Botti L, Piccinelli M, Ene-Iordache B, Remuzzi A, Antiga L. An adaptive mesh refinement solver for large-scale simulation of biological flows. *Communications in Numerical Methods in Engineering* 2009. Q6
3. Sahni O, Jansen K, Shephard M. Automated adaptive cardiovascular flow simulations. *Engineering with Computers* 2009; **25**(1):25–36.
4. Sahni O, Mueller J, Jansen K, Shephard M, Taylor C. Efficient anisotropic adaptive discretization of cardiovascular system. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**:5634–5655.
5. Thompson JF, Soni BK, Weatherill NP (eds). *Handbook of Grid Generation*. CRC Press, 1998.
6. Remacle J-F, Henrotte F, Carrier-Baudouin T, Bechet E, Marchandise E, Geuzaine C, Mouton T. A frontal Delaunay quad mesh generator using the l_∞ norm. *International Journal for Numerical Methods in Engineering* 2012. accepted.
7. Remacle J-F, Lambrechts J, Seny B, Marchandise E, Johnen A, Geuzaine C. Blossom-quad: a non-uniform quadrilateral mesh generator using a minimum cost perfect matching algorithm. *International Journal for Numerical Methods in Engineering* 2012; **89**:1102–1119.
8. Owen S, Cannan S, Saigal S. Pyramid elements for maintaining tetrahedra to hexahedra conformability. In *Trends in Unstructured Mesh Generation*, Vol. 220, AMD (ed.). ASME, 1997; 123–129. Q7
9. Meshkat S, Talmor D. Generating a mixed mesh of hexahedra, pentahedra and tetrahedra from an underlying tetrahedral mesh. *International Journal for Numerical Methods in Engineering* 2000; **49**(17-30). Q8
10. Yamakawa S, Shimida K. Fully-automated hex-dominant mesh generation with directionality control via packing rectangular solid cells. *International Journal for Numerical Methods in Engineering* 2003; **57**:2099–2129.
11. Yamakawa S, Shimida K. Increasing the number and volume of hexahedral and prism elements in a hex-dominant mesh by topological transformations. *12th International Meshing Roundtable*, 2003; 403–413. Q9
12. De Santis G, De Beule M, Segers P, Verdonck P, Verhegghe B. Patient-specific computational haemodynamics: generation of structured and conformal hexahedral meshes from triangulated surfaces of vascular bifurcations. *Computer Methods in Biomechanics and Biomedical Engineering* 2011; **14**(9):797–802.
13. Zhang Y, Bazilevs Y, Goswami S, Bajaj CL, Hughes TJR. Patient-specific vascular nurbs modeling for isogeometric analysis of blood flow. *Computer Methods in Applied Mechanics and Engineering* 2007; **196**(2943-2959).
14. Vmtk. (Available from: www.vmtk.org).
15. Antiga L, Ene-iordache B, Remuzzi A. Centerline computation and geometric analysis of branching tubular surfaces with application to blood vessel modeling. In *WSCG*, 2003. Q10
16. Antiga L, Ene-Iordache B, Remuzzi A. Computational geometry for patient-specific reconstruction and meshing of blood vessels from MR and CT angiography. *IEEE Transactions on Medical Imaging* 2003; **22**(5).
17. ~~Mount D, Arya N, Netanyahu R, Silverman AY.~~ An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM* 1998; Q11
18. Mount DM, Arya S. ANN library. (Available ~~from: http://www.cs.umd.edu/mount/ANN/~~).
19. Marchandise E, Compère G, Willemet M, Bricteux G, Geuzaine C, Remacle J-F. Quality meshing based on STL triangulations for biomedical simulations. *International Journal for Numerical Methods in Biomedical Engineering* 2010; **83**:876–889.
20. Marchandise E, Crosetto P, Geuzaine C, Remacle J-F, Sauvage E. Quality open source mesh generation for cardiovascular flow simulations. In *Modeling of Physiological Flows*, Vol. 5, MS&A Modeling, Simulation and Applications, chapter of ~~modelling~~ physiological flows. Springer-Verlag: Berlin Heidelberg, 2012; 395–414.
21. Marchandise E, Remacle J-F, Geuzaine C. Optimal parametrizations for surface remeshing. *Engineering with Computers* 2013. accepted.
22. Remacle J-F, Geuzaine C, Compère G, Marchandise E. High quality surface meshing using harmonic maps. *International Journal for Numerical Methods in Engineering* 2010; **83**:403–425.
23. Connell S, Braaten ME. Semi-structured mesh generation for three-dimensional Navier-Stokes calculations. *AIAA Journal* 1995; **33**(6):1017–1024.
24. Garimella RV, Shephard MS. Boundary layer mesh generation for viscous flow simulations. *International Journal for Numerical Methods in Engineering* 2000; **49**(1):193–218.
25. Kallinderis Y, Khawaja A, McMorris H. Hybrid prismatic/tetrahedral grid generation for viscous flows around complex geometries. *AIAA Journal* 1996; **34**(291-298).
26. Gouraud H. Continuous shading of curved surfaces. *IEEE Transactions on Computers* 1971; **20**(6):623–629.
27. Rusinkiewicz S. Estimating curvatures and their derivatives on triangle meshes. In *2nd International Symposium on 3D Data Processing*, September 2004; 486–493. Q12
28. Alauzet F. Size gradation control of anisotropic meshes. *Finite Elements in Analysis and Design* 2010; **46**(1):181–202.
29. Hecht F. Bamg: bidimensional anisotropic mesh generator, 2006. (Available from: http://www.freefem.org/ff++).
30. Remacle JF. An introduction to mesh generation part IV : elliptic meshing. (Available from: http://perso.uclouvain.be/vincent.legat/teaching/documents/meca2170-jfr-cours4.pdf).
31. Bunin G. Non-local topological clean-up. In *15th International Meshing Roundtable*. Springer-Verlag, September 2006; 3–20. Q13

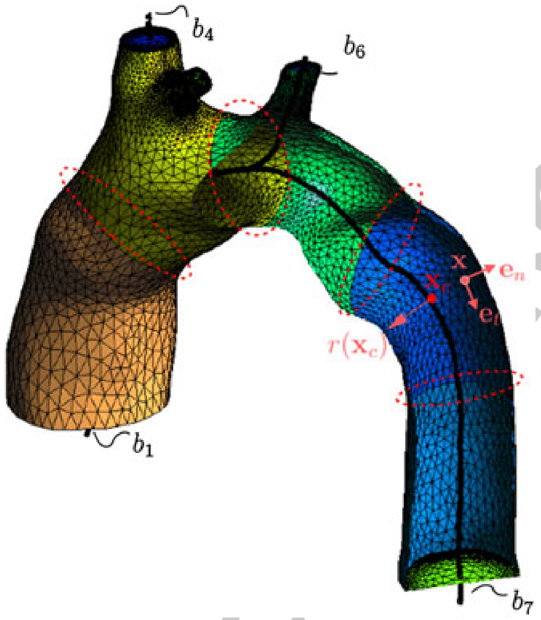
32. Geuzaine C, Remacle J-F. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering* 2009; **79**(11):1309–1331.
33. George P-L, Frey P. *Mesh Generation*. Hermes, 2000.
34. Rebay S. Efficient unstructured mesh generation by means of Delaunay triangulation and bowyer-watson algorithm. *Journal of Computational Physics* 1993; **106**:25–138.
35. Si H. TetGen. a quality tetrahedral mesh generator and three-dimensional delaunay triangulator, 2007. (Available from: http://tetgen.berlios.de).
36. Dobrzynski C. Mmg3d 4.0: anisotropic tetrahedral remesher/moving mesh generation, 2012. (Available ~~from: http://www.math.u-bordeaux1.fr/cdobrzyn/logiciels/mmg3d.php).~~
37. Dobrzynski C, Frey P. Anisotropic delaunay mesh adaptation for unsteady simulations. In *17th international Meshing Roundtable*. Springer: Heidelberg, 2008; 177–194.
38. Alauzet F, Xiabgrong L, Seegyoung Seol E, Shephard M. Parallel anisotropic 3D mesh adaptation by mesh modification. *Engineering with Computers* 2006; **21**:247–258.

DOI: 10.1002/cnm

Research Article

Cardiovascular and lung mesh generation based on centerlines

E. Marchandise, C. Geuzaine, and J. F. Remacle



This paper presents a new automatic meshing algorithm for the generation of computational meshes from a segmented tubular geometry. The proposed methodology is based on different centerline-based descriptors and operators. Different types of computational meshes can be generated with this method: isotropic tetrahedral meshes, anisotropic tetrahedral meshes, or mixed hexahedral/tetrahedral meshes as well as boundary layer meshes for the lumen wall. The mesh size field is a function of the centerline-based descriptor.

Author Query Form

Journal: International Journal for Numerical Methods in Biomedical Engineering

Article: cnm_2549

Dear Author,

During the copyediting of your paper, the following queries arose. Please respond to these by annotating your proofs with the necessary changes/additions.

- If you intend to annotate your proof electronically, please refer to the E-annotation guidelines.
- If you intend to annotate your proof by means of hard-copy mark-up, please refer to the proof mark-up symbols guidelines. If manually writing corrections on your proof and returning it by fax, do not write too close to the edge of the paper. Please remember that illegible mark-ups may delay publication.

Whether you opt for hard-copy or electronic annotation of your proofs, we recommend that you provide additional clarification of answers to queries by entering your answers on the query sheet, in addition to the text mark-up.

Query No.	Query	Remark
Q1	AUTHOR: Please check presentation of affiliations 1 and 2 if presented correctly.	✓
Q2	AUTHOR: Standard triangulation language. Is this the correct definition for STL? Please change if incorrect.	✓
Q3	AUTHOR: Please give manufacturer information for this product: company name, town, state (if USA), and country.	✓
Q4	AUTHOR: Figure 10 was not cited in the text. An attempt has been made to insert the figure into a relevant point in the text - please check that this is OK. If not, please provide clear guidance on where it should be cited in the text.	✓
Q5	AUTHOR: Please provide “access on” date for References 1, 14, 18, 29, 30, 35, and 36.	✓
Q6	AUTHOR: If References 2, 6, and 21 are now published online please provide DOI information. If published in print please provide volume number, issue number and page range.	✓
Q7	AUTHOR: Please provide city location of publisher for References 8, 31, and 33.	✓
Q8	AUTHOR: Please provide page range for References 9, 13, 15-17, and 25.	✓
Q9	AUTHOR: Please provide location where conference was held for Reference 11.	✓

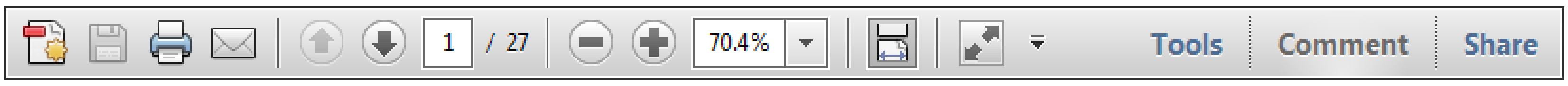
Q10	AUTHOR: Please provide name of publisher and city location for References 15.	✓
Q11	AUTHOR: Please check if author names are presented correctly for Reference 17.	✓
Q12	AUTHOR: Please provide location where symposium was held for Reference 27.	✓
Q13	AUTHOR: Please check if Reference 31 is presented correctly.	✓

WILEY-BLACKWELL

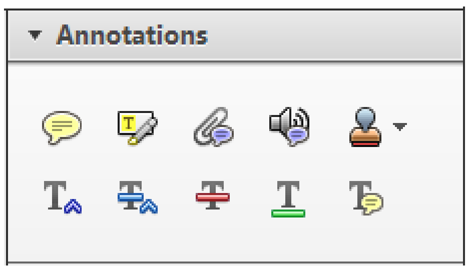
USING e-ANNOTATION TOOLS FOR ELECTRONIC PROOF CORRECTION

Required software to e-Annotate PDFs: Adobe Acrobat Professional or Adobe Reader (version 7.0 or above). (Note that this document uses screenshots from Adobe Reader X)
The latest version of Acrobat Reader can be downloaded for free at: http://get.adobe.com/uk/reader/

Once you have Acrobat Reader open on your computer, click on the Comment tab at the right of the toolbar:



This will open up a panel down the right side of the document. The majority of tools you will use for annotating your proof will be in the Annotations section, pictured opposite. We've picked out some of these tools below:

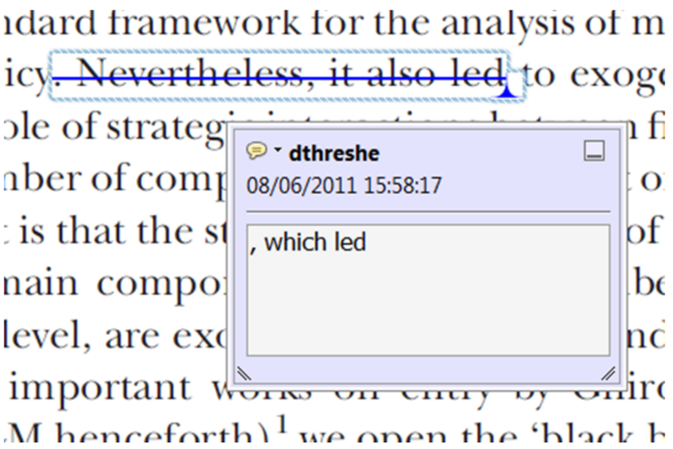


1. Replace (Ins) Tool – for replacing text.

Strikes a line through text and opens up a text box where replacement text can be entered.

How to use it

- Highlight a word or sentence.
- Click on the Replace (Ins) icon in the Annotations section.
- Type the replacement text into the blue box that appears.



ıdard framework for the analysis of m
icy ~~. Nevertheless, it also led~~ to exoge
ple of strateg
ıber of comp
: is that the st
nain compo
level, are exc
important
M henceforth)[1] we open the 'black b

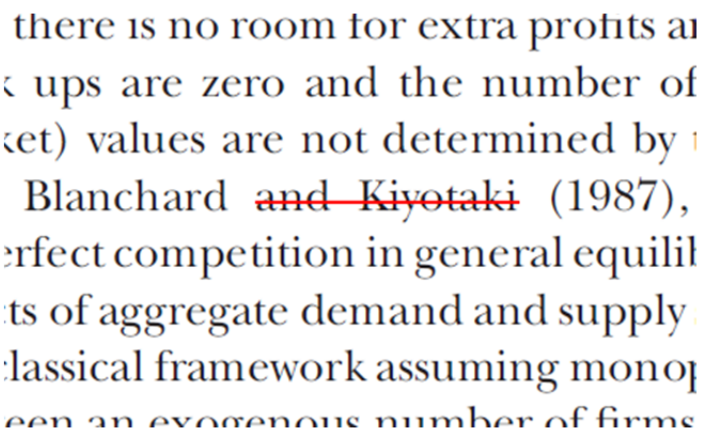
dthreshe
08/06/2011 15:58:17
, which led

2. Strikethrough (Del) Tool – for deleting text.

Strikes a red line through text that is to be deleted.

How to use it

- Highlight a word or sentence.
- Click on the Strikethrough (Del) icon in the Annotations section.



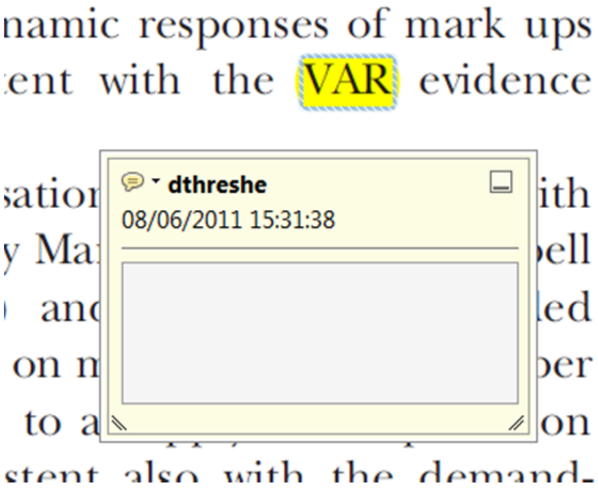
there is no room for extra profits a
ups are zero and the number of
ket) values are not determined by
Blanchard ~~and Kiyotaki~~ (1987),
erfect competition in general equili
ts of aggregate demand and supply
lassical framework assuming mono
een an exogenous number of firms

3. Add note to text Tool – for highlighting a section to be changed to bold or italic.

Highlights text in yellow and opens up a text box where comments can be entered.

How to use it

- Highlight the relevant section of text.
- Click on the Add note to text icon in the Annotations section.
- Type instruction on what should be changed regarding the text into the yellow box that appears.



namic responses of mark ups
ent with the VAR evidence
sation
y Ma
and
on n
to a
stent also with the demand-

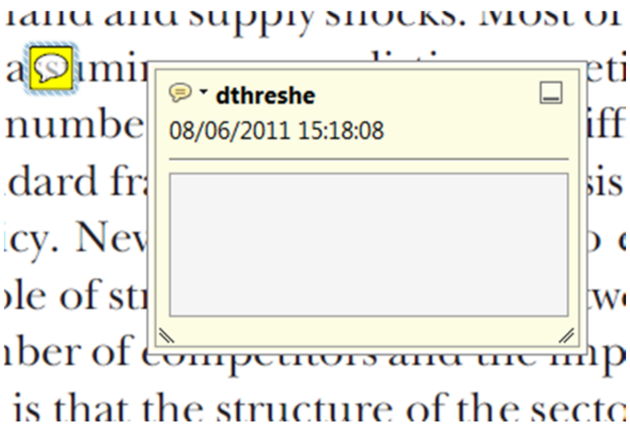
dthreshe
08/06/2011 15:31:38

4. Add sticky note Tool – for making notes at specific points in the text.

Marks a point in the proof where a comment needs to be highlighted.

How to use it

- Click on the Add sticky note icon in the Annotations section.
- Click at the point in the proof where the comment should be inserted.
- Type the comment into the yellow box that appears.



land and supply shocks. Most of
a mi
numbe
dard fr
cy. Nev
ple of st
ıber of competitors and the imp
is that the structure of the secto

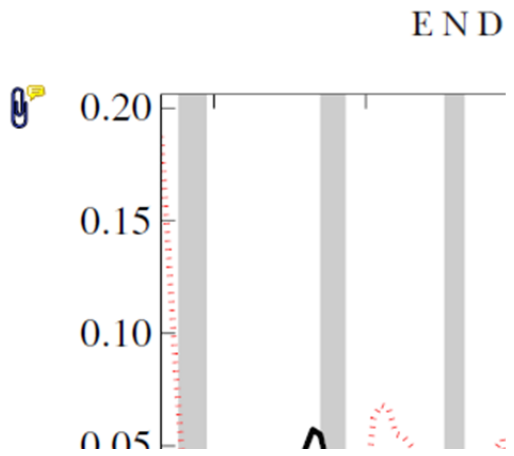
dthreshe
08/06/2011 15:18:08

5. Attach File Tool – for inserting large amounts of text or replacement figures.

Inserts an icon linking to the attached file in the appropriate pace in the text.

How to use it

- Click on the Attach File icon in the Annotations section.
- Click on the proof to where you'd like the attached file to be linked.
- Select the file to be attached from your computer or network.
- Select the colour and type of icon that will appear in the proof. Click OK.

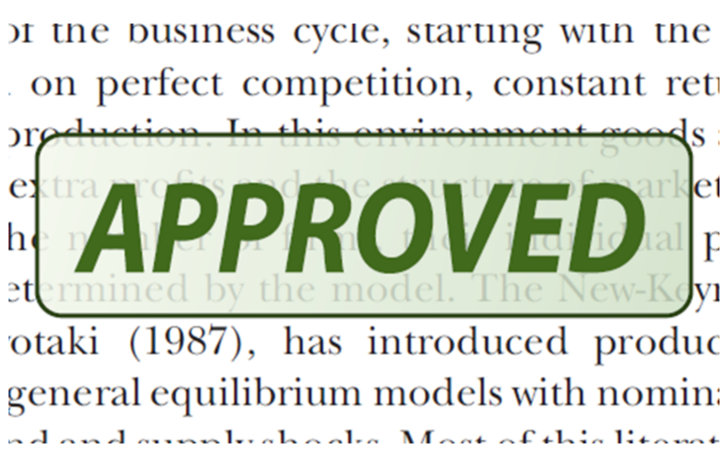


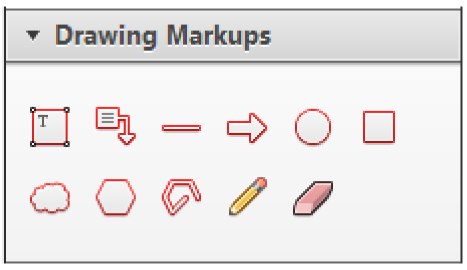
6. Add stamp Tool – for approving a proof if no corrections are required.

Inserts a selected stamp onto an appropriate place in the proof.

How to use it

- Click on the Add stamp icon in the Annotations section.
- Select the stamp you want to use. (The Approved stamp is usually available directly in the menu that appears).
- Click on the proof where you'd like the stamp to appear. (Where a proof is to be approved as it is, this would normally be on the first page).



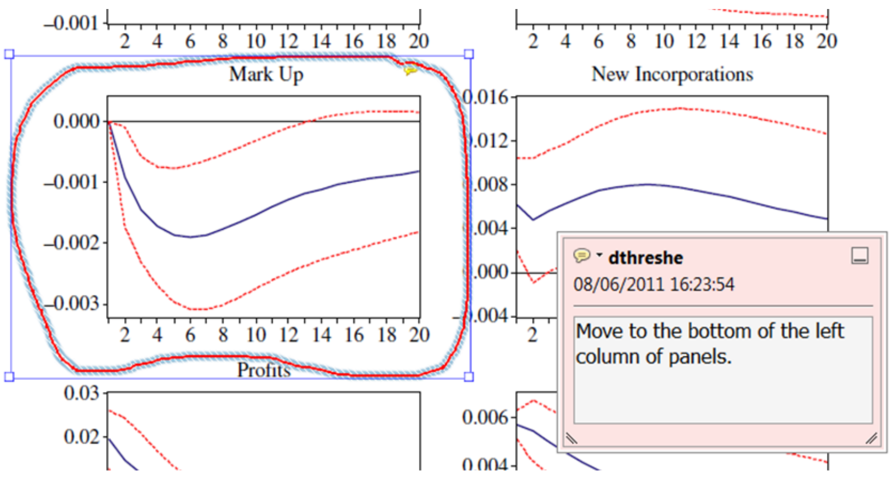


7. Drawing Markups Tools – for drawing shapes, lines and freeform annotations on proofs and commenting on these marks.

Allows shapes, lines and freeform annotations to be drawn on proofs and for comment to be made on these marks..

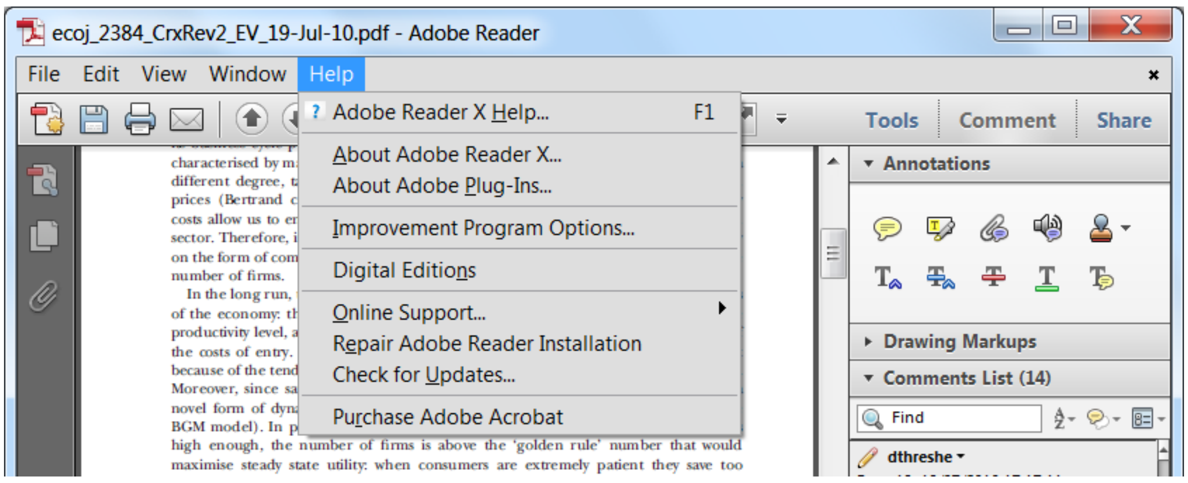
How to use it

- Click on one of the shapes in the Drawing Markups section.
- Click on the proof at the relevant point and draw the selected shape with the cursor.
- To add a comment to the drawn shape, move the cursor over the shape until an arrowhead appears.
- Double click on the shape and type any text in the red box that appears.



dthreshe
08/06/2011 16:23:54
Move to the bottom of the left column of panels.

For further information on how to annotate proofs, click on the Help menu to reveal a list of further options:



ecoj_2384_CrxRev2_EV_19-Jul-10.pdf - Adobe Reader