HAL
open science

A Software to Visualize, Edit, Model and Mesh Vascular Networks

Méghane Decroocq, Guillaume Lavoué, Makoto Ohta, Carole Frindel

To cite this version:

Méghane Decroocq, Guillaume Lavoué, Makoto Ohta, Carole Frindel. A Software to Visualize, Edit, Model and Mesh Vascular Networks. 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), Jul 2022, Glasgow, United Kingdom. 10.1109/EMBC48229.2022.9871365 . hal-03772160

HAL Id: hal-03772160

https://hal.science/hal-03772160v1

Submitted on 8 Sep 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Software to Visualize, Edit, Model and Mesh Vascular Networks

Méghane Decroocq[1,2,4,5,6], Guillaume Lavoué[3], Makoto Ohta[4,5] and Carole Frindel[1,5]

Abstract— **Computational fluid dynamics (CFD) is a key tool for a wide range of research areas, beyond the computer science community. In particular, CFD is used in medicine to measure blood flow from patient specific models of arteries. In this field, the creation of accurate meshes remains the most challenging step, as it is based on the segmentation of medical images, a time-consuming task which often requires manual intervention by medical doctors. In this context, user-friendly, interactive softwares are valuable. They enable to spread the new advances in numerical treatment to the medical community and enrich them with the expert knowledge (e.g anatomical knowledge) of clinicians. In this work, we present a user interface dedicated to the meshing of vascular networks from centerlines. It allows for the 3D visualization and edition of input centerlines, which constitute a simplified, easy-to-manipulate representation of vascular networks. The surface of the artery can be reconstructed from the modified centerlines by an editable parametric model and then meshed with high quality hexahedral elements. At every step of the process, the network can be confronted with medical images with enhanced visualization. The software will be released publicly.**

Clinical relevance— **This tool facilitates the manual extraction and editing of vascular networks by medical doctors. It opens the generation of hexahedral meshes for computational fluid dynamics studies to non-expert users.**

I. INTRODUCTION

Computational fluid dynamics (CFD) is increasingly used to retrieve valuable hemodynamic information, helping to understand, prevent and predict the outcome of vascular diseases [17]. As CFD widely relies on meshes, the reconstruction of patient-specific vascular networks from medical images is critical. It remains a challenging task, especially for large vascular networks with small vessels [15]. The use of simplified representations (i.e centerlines) helped overcoming this problem, leading to the creation of large open databases [21], [7]. Nevertheless, the complex pipeline involved from centerline extraction to mesh generation was not yet successfully automated and calls for manual intervention and expert anatomical knowledge. In this context, it is valuable to open this pipeline to users who do not necessarily have expert programming skills (e.g clinician, biologist), through easy-to-use, free and open softwares.

A variety of softwares are available today to generate and visualize centerlines from raw or segmented medical images on one side [8], [1], [8], [16] and to discretize a surface mesh with tetrahedral elements on the other side [10], [13]. We believe there is still a gap between the extraction of centerlines and the generation of a volume mesh, that we intend to bridge with this work. For this, we developed a user-friendly software integrating all the functions required to move from a input centerline with defects to an accurate, high quality mesh ready for CFD. Our software is composed of two main parts : (1) a dedicated graphical user interface allowing relevant manual interactions and (2) an efficient modeling and meshing engine, which was developed in a previous work [11]. In Section II we compare the proposed tool to other softwares of the literature. Section III gives an overview of the tool. Sections IV, V and VI describe in detail the different functionalities offered by our tool, parted into the data point-related, the model-related and the mesh-related functionalities. Finally, a use case of the software is proposed in the Section VII : it is used to correct and mesh centerlines from the open database BraVa [21].

II. RELATED WORK

In this section we present the available softwares dedicated to the visualization and edition of vascular networks. Two types of methods are commonly used to generate patient-specific meshes of vascular networks : the segmentation of medical images and the centerline extraction. There was a renewed interest for the centerline-based representation of vascular networks, either as an additional contribution to the segmentation or meshing, or as a self-reliant way to model vascular networks. As opposed to image segmentation, centerlines advantageously incorporates topological information and facilitates manual extraction and manipulation with its reduced number of parameters. It opens the way to numerical experiments in always larger and more complex networks. This interest is attested by the release of new softwares focusing on the visualization of vascular networks from centerlines in the recent years.

Longair et al. [16] presented the ImageJ plugin Simple Neurite Tracer. The application was originally designed for the semi-automated tracing of neurons, but can be applied to other tubular structures. The user places centerline points on the axial and sagittal slices of a 3D image. The path between points and the radius is automatically computed. The VesselVio application [8] enables the automatic extraction of centerlines from pre-binarized medical images

*This work was supported by AURA region and ANR (PreSpin)

[1]CREATIS, Université Lyon1, CNRS UMR5220, INSERM U1206, INSA-Lyon, 69621 Villeurbanne, France carole.frindel@creatis.insa-lyon.fr

[2] Univ Lyon, INSA Lyon, CNRS, UCBL, Centrale Lyon, Univ Lyon 2, LIRIS, UMR5205, F-69621 Villeurbanne, France

[3] Univ Lyon, Centrale Lyon, CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, F-69130 Ecully, France

[4] ELyTMaX IRL3757, CNRS, INSA Lyon, Centrale Lyon, Université Claude Bernard Lyon 1, Tohoku University, 980-8577, Sendai, Japan

[5] Institute of Fluid Science, Tohoku University, 2-1-1, Katahira, Aoba-ku, Sendai, Miyagi 980-8577, Japan

[6] Graduate School of Biomedical Engineering, Tohoku University, 6-6 Aramaki-aza-aoba, Aoba-ku, Sendai, Miyagi 980-8579, Japan

and the computation of various quantitative features of the vascular network. Both applications focuses on the manual or automatic extraction of centerlines and their visualization, which corresponds to a step prior to the application we propose. The edition and the generation of a CFD mesh is not the purpose of these works. Besides, Abdellah et al. proposed the VessMorphoVis Blender plugin [1], whose purpose is close to ours. This application enables to visualize, analyze and automatically repair the vascular skeletons. A basic polyline mesh as well as a more realistic polygonal mesh based on metaballs can be generated automatically from the centerlines. This useful plugin differs from our application in several ways. The interface is not intended for centerline editing on the basis of the medical images and the proposed meshing method is mainly dedicated to visualization.

None of these applications allow to produce the mesh discretization of vascular networks required for CFD. This task is commonly performed with general softwares able to discretize any surface mesh with tetrahedral elements [13]. However, discretization methods specifically designed for vascular networks are able to produce more adapted and efficient meshes, with flow oriented, high quality hexahedral elements. Few softwares are dedicated to the volume meshing of vasculature. The VMTK software [14] performs centerline extraction from a surface mesh, and uses the centerline and surface to generate a tetrahedral volume mesh with an hexahedral boundary layer. The interface PyFormex [10] is dedicated to the semi-automatic hexahedral remeshing of a surface mesh with a single vascular bifurcation. In both cases, the input required is a surface mesh, which is challenging to obtain for large vascular networks.

Our application aims at filling the gap between the visualization and meshing softwares. It includes both visualization, centerline edition, and hexahedral meshing functionalities within a single framework. Unlike existing softwares, it allows the user to manual interact and correct the algorithm outputs at every step of the modeling, on the basis of the information contained in medical images. It is the first application so far to enable the creation of hexahedral mesh from centerlines only.

III. OVERVIEW

A. Input data

The input of the software is the vascular network to edit and mesh. The objective of the present work is to deal directly with centerlines, not segmented images or meshes. The input centerlines are composed of a set of data points with three spatial coordinates (x,y,z) and a radius value (r), and the connectivity between the points -oriented in the flow direction-. Data points might have several successors, forming bifurcations. The input formats supported include the SWC format proposed by Cannon et al. [9] and VTK polyLines structure as used by the centerline extraction software VMTK (www.vmtk.org) [14]. These formats are the most commonly used in the literature and they allow to read data from the large open databases of vascular centerlines Aneurisk [3] - in VMTK format - and BraVa [21] - in SWC format - for instance. In addition, we created a new input format dedicated to the creation of idealized vascular networks with straight vessels and uniform radius. This type of vascular models are used in many studies investigating the impact of vascular geometry on blood flow [5], [6]. In a text file, each line contain a list of six values separated by a space, representing one branch. These values are respectively the id, the branch length (*mm*), the branch radius (*mm*), the number of data points, the angle formed with the previous branch (*degree*) and the id of the previous branch.

B. Main functionalities

The proposed software enables the user to visualize and edit the centerline manually with respect to the medical image. The main geometric and topological features (e.g branch angle, branch configuration) of the network can be modified effortlessly. A parametric model is used to reconstruct a realistic and continuous surface from the input centerlines, based on the method described in [11]. The parameters of the model can also be visualized and edited. From this model, a structured mesh with hexahedral, flow-oriented cells can be generated automatically [11]. This type of mesh is known to improve CFD performances compared to the commonly used unstructured tetrahedral meshes [20], [10]. At any time during the process, the centerline, the model and the mesh can be saved and exported in various formats. Four different representation modes (data, topology, model and mesh) corresponding to different steps toward the final mesh, can be displayed. These representations are detailed in the following sections and illustrated in Figure 2, as well as the functionalities mentioned in this overview.

C. Interface layout

The interface was built using the 3D programming python package vpython (www.vpython.org). It was chosen over other well-known softwares optimized to handle large meshes, such as Blender (www.blender.org) or Paraview (www.paraview.org), for its simplicity of use and user-interaction oriented functions. It is more suited for manual editing than visualization-oriented tools such as Paraview or pyvista (www.pyvista.org), while providing a more simple, user-friendly layout than Blender. As illustrated in Figure 1, the designed interface integrates the following components:

- The 3D viewer is dedicated to the display of the objects to visualize, here the vascular networks. It handles both the basic user interactions such as rotation and zoom and the original edition functions developed in this work.
- Important information, software handling guidance and algorithm output messages are displayed in the message bar.
- The edit menu activates the edition mode for one of the four vascular network representations : data, topology, model or mesh.
- The image import and management bar allows the user to import a medical image and visualize it in the 3D view through the dedicated widgets.

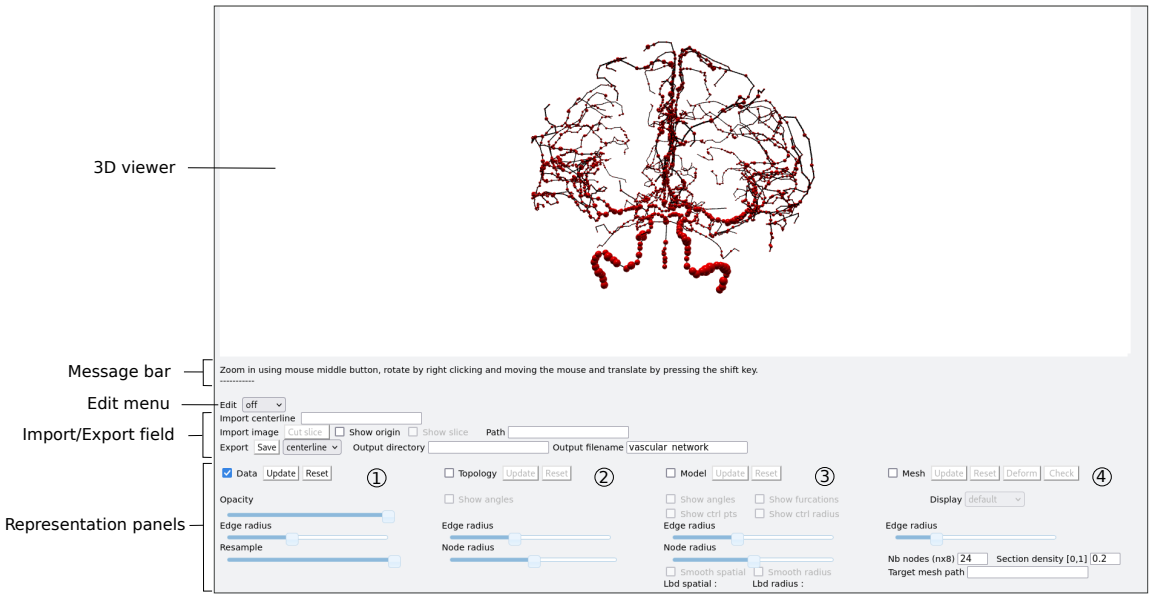

Fig. 1. The interface layout. The numbers 1 to 4 corresponds to the different representation modes of the interface (data, topology, model, mesh).

- An new input centerline can be imported trough the import field.
- The export field allows the edited networks to be saved in various formats.
- The four visualization panes can be activated independently by checking the dedicated box. They control the display of the different vascular network representation modes (data, topology, model and mesh) on the 3D viewer. The representation modes are detailed in the following sections. If activated simultaneously, the representations are superimposed on the viewer. Each pane gathers several widgets which control the functionalities associated with a given representation mode.

IV. DATA POINT EDITION

In this section, we describe the functionalities developed in order to edit the raw input centerlines, enabling to correct the centerline data with regard to the medical image and modify the topology and geometry of the network effortlessly.

A. Network representation

The input centerline data before treatment can be displayed under two representations modes ; the data mode and topology mode. In data mode, each centerline data point (x, y, z, r) is represented on the 3D view by a red sphere of center (x, y, z) and radius r. The spheres are connected with black lines, according to the topology of the network. This mode enables to visualize all the information provided by the input centerlines in a single representation. However, the underlying topology of the vascular network may be hidden by the data point spheres, especially for centerlines with high sampling. To facilitate the visualization, another representation called topology mode is proposed.

In topology mode, only the key data points (furcation, inlet and oulet) are displayed. They are represented with spheres of uniform radius. A single color is associated to each type of key point. The connecting vessels are represented by black polylines. The opacity and the size of the spheres and polylines can be adjusted using the dedicated sliders, allowing different representations to be overlapped easily. The data and topology representation modes are illustrated in the column 1 and 2 of Figure 2. In order to facilitate centerline editing, the data of the different representation modes are encoded in the back-end as directed spatial graphs [4], using the networkx python package.

B. Medical image overlap

In order to ensure that the centerline network is as close as possible to the real patient anatomy, a medical image (in NIFTI format) can be imported in the viewer. As there is no registration step, the user must ensure beforehand that the image real coordinate in *mm*, as written in the image header, matches the initial centerline data point coordinates. The visualization of large 3D images such as magnetic resonance angiography (MRA) is challenging. In order to enhance the relevant information and accelerate the rendering, we propose to display local 2D slices of the image. The center of the slice is defined by a user-set origin cursor, the normal being the current scene viewpoint angle. Ray tracing is employed to extract a slice according to a given normal from the 3D image. By default, a square slice of size $40mm^2$ is created, which we think is sufficient to visualize locally the trajectory of a given vessel. For clarity and computational time requirements, only one slice at a time can be displayed. By computing the slides sequentially along the vasculature, the input centerline can easily be compared to the vessel position on the image, enabling to manually correct different parts of the network. The image slice can also be juxtaposed with other representation modes such as the model or the mesh (Sections V and VI). A example of medical image

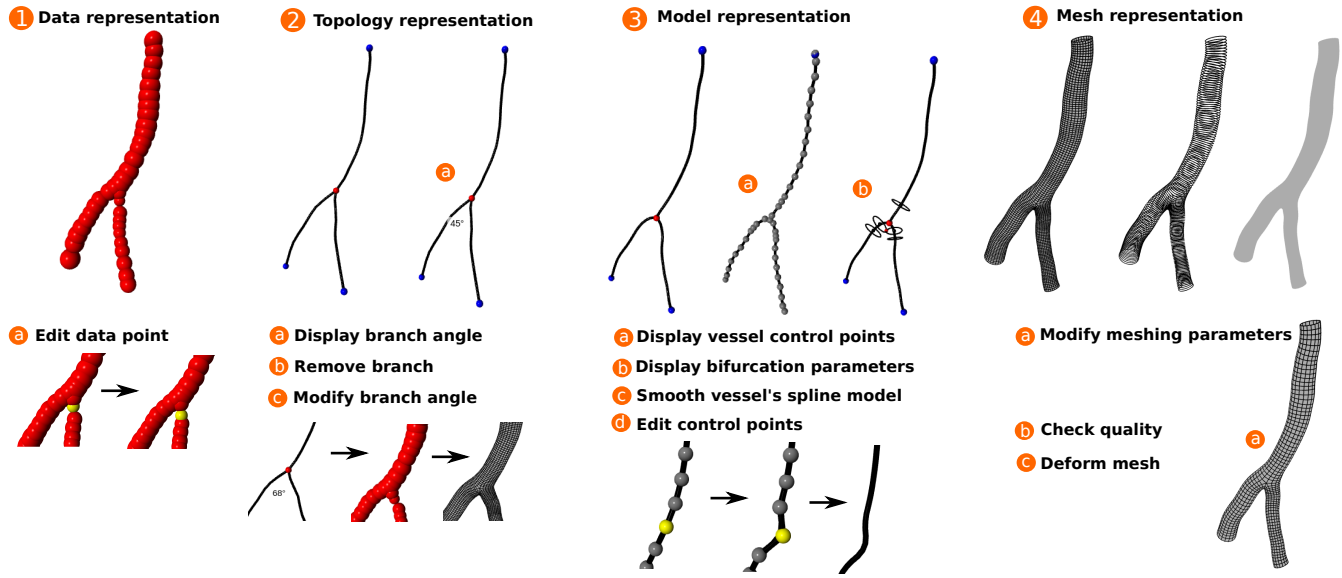

Fig. 2. Overview of functionalities of the software. The numbers 1 to 4 corresponds to the different representation modes of the interface (data, topology, model, mesh). For each mode, the related-functionalities are listed and illustrated with displays of the software. For space reason, all the functionalities could not be illustrated ; please refer to Figure 3 for more images.

superimposition is provided in Figure 3.

C. Data point editing

The centerline data points position and radius can be edited directly from the 3D view. To prevent accidental modification of the centerlines, the editing mode must be activated beforehand. Centerline editing is performed from the data representation mode. A data point can be selected by left clicking on the associated sphere. The color of the selected object is automatically modified and it can be dragged to a new position using the mouse (Figure 2 image 1.a). If the change is judged satisfying by the user, clicking the *Update* button will save the new position. The *Reset* button cancels any previous modification. The radius of the data point can also be adjusted after selection by keyboard control. Finally, a slider allows for data resampling, to facilitate the visualization of centerline with high density of data points. When moving the slider, data points are removed uniformly along the centerlines. We acknowledge that it can be hard to visualize and manipulate large networks. For this reason, we implemented a allowing the user to select the vessels of interest and crop the network accordingly, like in Figure 3. It is therefore possible to focus only on a small part of the network and easily go back to the visualization of the entire network.

D. Branch removal

The topology of the vascular network differs between individuals. For instance, the circle of Willis has various configurations, which impacts the hemodynamics [2]. To facilitate the study of the influence of the network topology, a trimming functionality is included in our software. This function allows the removal of branches or entire arterial territories. We define a branch as a vessel segment connecting two end or bifurcation points. In topology mode, the branch to remove is selected by left clicking, and suppressed by pressing the *suppr* key. To preserve the connectivity of the network, all the downstream vessels arising from the selected branch are also removed. The identification of downstream vessels is performed efficiently using a depth-first search in the graph associated with the topology representation mode, as mentioned in Section IV-A.

E. Branch angle

There are many studies investigating the impact of the angle formed by two branches on blood flow [19], [18], [5]. The creation of the meshes in those studies mostly relies on manual design using 3D modeling or CAD softwares. In this work, we propose to simplify this task by allowing the user to modify effortlessly any branch angle in the original network. The desired angle is achieved by rotating the relevant data points. The branch to rotate is first selected. Pressing the *'r'* (respectively *R*) key triggers a clockwise (respectively counterclockwise) rotation of the branch. The rotation center is the bifurcation point and the rotation axis is the current viewpoint on the scene. The default rotation step angle is 0.1. The key can be pressed repetitively or continuously until the wanted angle is achieved. The downstream branches are rotated together to preserve their original branch angles.

Additionally, the branching angles can be computed and displayed on the 3D view with the *Show angle* box. In topology mode, it is computed as the angle formed by the vectors connecting the bifurcation point to a point located one diameter downstream in each daughter branch. The angle obtained by this method may be affected by the quality of the initial centerlines. Angles can be retrieved more accurately from the model representation, in which case the computation method differs. More explanations about the

creation of the vascular model and the angle computation method associated are given in the next section. The images 2.a and 2.c of Figure 2 give an example of angle display and modification.

V. MODELING

In this section, we present the model used to reconstruct the vessel surface from the centerlines. This parametric model can also be visualized and edited from the interface.

A. Modeling method

We integrated in our tool a parametric modeling method which was developed in a previous work [11], to represent the surface of the network from its centerlines. As some features of the editor are based on this model, we will briefly describe it here. The network is divided into bifurcations and vessel segments, which are modeled separately. The vessels are represented by a tubular surface, which is defined by a spline function gathering 4 coordinates ; the three spatial coordinates of the centerline (x,y,z) and the radius r. The control points of the spline are optimized by the least square method, with a two-fold objective, as described in [12]. The first term of the cost function minimizes the distance to the input data points, the second is a smoothness penalty. The smoothness penalty is used to guarantee the smoothness to the vessel surface and to correct noise which is a common defect of centerlines extracted from medical images. The balance between data proximity and smoothness is controlled by the parameter $\lambda \in [0;+\inf[$. The value of λ is selected according to the Akaike criterion [12]. It is optimized independently for the spatial dimension ($\lambda = \lambda_s$) and the radius ($\lambda = \lambda_r$).

For bifurcations, we used the parametric model proposed by [22]. In this model, bifurcations are represented as the union of two tubular vessels sharing the same inlet. The parameters of the model are : the inlet cross section, two apical cross sections - located at the apex of the bifurcation - and two outlet cross sections. Each cross section is defined by a center, a radius and a normal vector. The normal of the inlet and outlet sections is matched to the initial/end tangents of the vessel spline models to ensure the continuity of the network. Within this model, the bifurcation angle can be estimated as the angle between the apex point and the outlet section points facing the same direction. The modeling method is only briefly described in this paper as it focuses on the interface, for details -notably concerning the reconstruction accuracy- please refer to [11].

B. Model representation

The vessel and bifurcation models obtained by the method described in the previous paragraph can be displayed via the model representation mode. By default, the bifurcation barycenter as well as the inlet and oulet points are represented by spheres of different colors, and the vessels' spline model by black curves. The control points of the splines and the cross sections composing the bifurcation models can optionally be displayed on the viewer using the dedicated checkbox. The different model representations are illustrated in the column 3 of Figure 2.

C. Control points editing

The surface of the mesh produced by our framework is defined entirely by this parametric model. For this reason, we give the possibility for the user to directly edit the model through the interface. It is an alternative to the editing of input data points presented in Section IV-C which is less time-consuming and gives a more direct control on the final surface. The control points are represented on the 3D viewer by gray spheres. Any of those spheres can be selected and dragged using the mouse, in the same way any regular data point can be moved. As the control points are modified, the spline trajectory is updated, as showed in the image 3.d of Figure 2.

D. Spline smoothing

As explained briefly in Section V-A, vessel segments are modeled by penalized splines whose spatial (resp. radius) smoothness is controlled by the parameter λ_s (respectively λ_r). Although we propose an automatic method to find the optimal value for those smoothing parameters which was proven rather robust to noise and low sampling of the centerlines [11], it can also be manually adjusted in the interface to address some cases more specifically. Unlike control points editing, smoothing affects all the control points of a spline simultaneously, modifying a vessel segment globally. In model mode, the vessel segment to smooth is selected by mouse clicking on a curve. The parameter to vary, λ_s or λ_r, is selected through the smoothing checkboxes and modified by pressing the dedicated key. The impact of the changes on the curve trajectory and control points is displayed in real time, as well as the current parameter value.

VI. HEXAHEDRAL MESHING

A. Meshing

From the model presented in section V, a cell-oriented, structured hexahedral mesh can be created by the method described in [11]. For this, a serie of cross sections is computed along the vessels and connected to form rectangular faces. The cell density can be adjusted by setting two parameters directly in the interface : the longitudinal density of cross sections and the radial density (i.e the number of nodes forming a cross section). The mesh generation has a computational time independent from the display time. It can range from a few minutes to half an hour, depending on the size of the network to mesh. On the 3D view, four modes are available to display the mesh, as showed in the column 4 of Figure 2 ; the *default* mode (surface and edges), *solid* mode (surface only), *wireframe* (edges only) *section* (cross sections only). The framework used for the creation of the interface, vpython, was selected as a compromise between the efficiency and simplicity of use. It is not optimized for the display and manipulation of large meshes as other software such as Paraview or Blender can be. This can cause the interface to become slow or irresponsive when working

with a large vascular network in mesh representation. In our experience, only meshes with less than $50k$ faces can be displayed smoothly. To circumvent this limitation, we applied the same idea as for the visualization of the 3D image : to show only what is needed. Through an edge selection menu, the mesh can be displayed one part at a time, in function of the current visualization needs.

B. Quality evaluation

The tubularity assumption can cause cross sections to intersect in high curvature areas, hindering the use of the mesh for numerical simulation purposes. In such a case, a local editing of the centerline at the intersection area is required. To facilitate the localization of the meshing failures, we implemented a function that highlights the low quality area by adding a red coloration on the surface mesh, as showed in Figure 3. We used the scaled Jacobian as a quality metric. The scaled Jacobian ranges from -1 to 1. It is generally admitted that numerical simulation will fail if the mesh has cells with a negative scaled Jacobian value. The Jacobian value is first computed for each cell of the volume mesh, then projected on the surface. The surface map is thresholded to color the invalid faces in red. As those faces can be very small or hidden, we have decided to color the entire section if the invalid cell is located in a vessel and all the faces if it is located in a bifurcation.

C. Deformation

The proposed model is based on the assumption that vessels cross sections are circular, which is limiting when dealing with pathological vessels. A way to address this limitation is to deform the cross sections to match a target surface as a post processing. For this, the user must provide a path to the target surface mesh (in stl or vtk format), and click the *Deformation* button. The nodes of the tubular surface mesh are projected to the target surface radially from the centerline. The maximum projection distance can be set to avoid unwanted large deformations if a mismatch is found between the initial and the target surface.

VII. APPLICATIONS

In this section, a use case illustrates the possible applications of the software. In this example, we will repair the centerlines of a large cerebral network provided in the open database BraVa [21] to generate an accurate mesh. The manipulations performed by the users with the software to complete this task are reported and showed in Figure 3. The centerlines of the BraVa database were extracted semi-automatically by medical doctors using the Simple Neurite Tracer plugin [16]. The use of those centerlines for CFD is hindered by limitations caused by the manual extraction (low number of data points, misplacement) and by the automatic algorithms (branching point misplacement, radius noise), as showed on Figure 3 A. If the meshing method employed in the interface was designed to be robust to centerline defects [11], it remains affected by the quality of the input data. In the A case, the misplacement of data points near the bifurcation led to the generation of invalid cells in the mesh. The low quality parts were highlighted in red by the mesh checking functionality provided in the software. It corroborates the defects identified by overlapping the medical image on different views. Once the visualization step

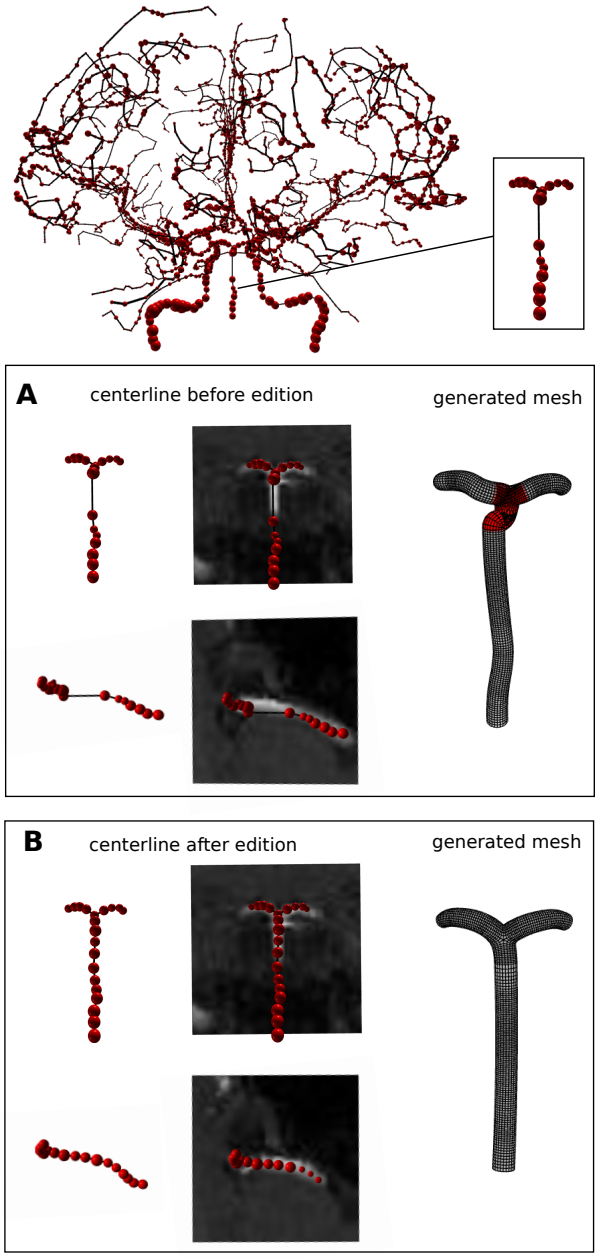

Fig. 3. Software use case for the manual correction of a vascular network of the BraVa database. The initial cerebral artery centerline as provided in the BraVa database is represented on top. In this example, we focused on the edition of the basilar artery. Panel A shows several views of the input centerline and the superimposition of the magnetic resonance angiography image. The mesh generated by the software for the centerline is showed on the right. The checking function was used to color in red the low quality cells. Panel B shows the same centerline after edition of the data points to match the medical image. The mesh produced from the corrected centerlines -on the right- has no low quality cells. The images in the figure are unchanged displays provided by the software.

(A) is achieved, the centerline data points can be corrected (B). The points were moved manually to match the medical image and their radius were adjusted consequently. Overall, the editing task only took a few minutes. The panel B of Figure 3 shows the centerline after correction and the new mesh generated. The final mesh has high quality hexahedral cells and reflects the patient anatomy with accuracy. Besides centerline edition, the software can be used for other applications such as the fast generation of a database of meshes with different branching angles or topologies from a single input network.

VIII. CONCLUSIONS

We proposed a user interface whose functionalities include the visualization, edition, model and meshing of vascular networks from centerlines. The interface was designed so that it is simple and intuitive to use. We believe that it can be useful to researchers in various fields including medicine and biomechanics. It was already tested by a panel of students with medical, biomechanical and informatic background for different types of studies. The provided feedback lead to the integration of functionalities adapted to the user needs such as the branching angle computation and display. Nevertheless, we acknowledge some limitations to this work. The interface logically inherits the limits of the automatic method employed for the modeling and meshing (e.g high curvature areas or mesh failures) [11]. However, it was precisely designed to overcome them by allowing the user to manually correct the output of the algorithm. Additionally, the mesh deformation function is limited, as it requires a target surface. Paradoxically, the easiest way to obtain this target surface is image segmentation, which is precisely the task we are trying to bypass in this framework. Moreover, big deformations can badly impact the quality and the properties of the initial tubular mesh. We plan to address this issue by implementing a new deformation method based on the medical image itself.

ACKNOWLEDGMENT

We would like to thank Kassidy Porche for her help on this work, and Pr. Hitomi Anzai, Keiichiro Shiraishi, Yutaro Kohata, Fangjia Pan, Leopold Kowalski, Chloé Adam, Sylvain Clary, Eliott Dupuis, Paul Marin and Ferdinand Nsimba for their feedbacks on the software. Founding is acknowledged through AURA region (SIMAVC project) and ANR 20-CE45-0011 (PreSpin project).

DATA AVAILIBILITY

The code of the software and some demonstration videos can be downloaded from github : https://github.com/megdec/vascularmd.

REFERENCES

[1] M. Abdellah, N. Román Guerrero, S. Lapere, J. S. Coggan, D. Keller, B. Coste, S. Dagar, J-D. Courcol, H. Markram, and F. Schürmann, Interactive visualization and analysis of morphological skeletons of brain vasculature networks with vessmorphovis, Bioinformatics, vol. 36, pp. 534–541, 2020.

[2] J. P. K. H. Alastruey, K. H. Parker, J. Peiró, S. M. Byrd, and S. J. Sherwin, Modelling the circle of willis to assess the effects of anatomical variations and occlusions on cerebral flows, Journal of biomechanics, vol. 40, pp. 1794–1805, 2007.

[3] Aneurisk-Team, AneuriskWeb project website, http://ecm2.mathcs.emory.edu/aneuriskweb, Web Site, 2012.

[4] S. R. Aylward, J. Jomier, C. Vivert, V. LeDigarcher, and E. Bullitt, Spatial graphs for intra-cranial vascular network characterization, generation, and discrimination, International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 59–66, Springer, 2005.

[5] S. Beier, J. Ormiston, M. Webster, J. Cater, S. Norris, P. Medrano-Gracia, A. Young, and B. Cowan, Impact of bifurcation angle and other anatomical characteristics on blood flow–a computational study of non-stented and stented coronary arteries, Journal of biomechanics, vol. 49, no. 9, pp. 1570–1582, 2016.

[6] J. Bredno, M. E. Olszewski, and M. Wintermark, Simulation model for contrast agent dynamics in brain perfusion scans, Magnetic Resonance in Medicine, vol. 64, no. 1, pp. 280–290, 2010.

[7] E. Bullitt, D. Zeng, G. Gerig, S. Aylward, S. Joshi, J.K. Smith, W. Lin, and M. G. Ewend, Vessel tortuosity and brain tumor malignancy: a blinded study, Academic radiology, vol. 12, no. 10, pp. 1232–1240, 2005.

[8] J. Bumgarner and R. Nelson, A novel and open-source application for vasculature dataset analysis and visualization, Preprint on webpage at https://www.researchsquare.com/article/rs-608609/v1, 2021.

[9] R. C. Cannon, D. A. Turner, G. K. Pyapali, and H. V. Wheal, An on-line archive of reconstructed hippocampal neurons, Journal of neuroscience methods, vol. 84, no. 1-2, pp. 49–54, 1998.

[10] G. De Santis, P. Mortier, M. De Beule, P. Segers, P. Verdonck, and B. Verhegghe, Patient-specific computational fluid dynamics: structured mesh generation from coronary angiography, Medical & biological engineering & computing, vol. 48, no. 4, pp. 371–380, 2010.

[11] M. Decroocq, C. Frindel, M. Ohta, and G. Lavoué, Modeling and hexahedral meshing of arterial networks from centerlines, 2022.

[12] P. H. C. Eilers and B. D. Marx, Flexible smoothing with b-splines and penalties, Statistical science, vol. 11, no. 2, pp. 89–121, 1996.

[13] C. Geuzaine and J-F Remacle, Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities, International journal for numerical methods in engineering, vol. 79, no. 11, pp. 1309–1331, 2009.

[14] R. Izzo, D. Steinman, S. Manini, and L. Antiga, The vascular modeling toolkit: a python library for the analysis of tubular structures in medical images, Journal of Open Source Software, vol. 3, no. 25, pp. 745, 2018.

[15] D. Lesage, E. D. Angelini, I. Bloch, and G. Funka-Lea, A review of 3d vessel lumen segmentation techniques: Models, features and extraction schemes, Medical image analysis, vol. 13, no. 6, pp. 819–845, 2009.

[16] M. H. Longair, D. A. Baker, and J. D. Armstrong, Simple neurite tracer: open source software for reconstruction, visualization and analysis of neuronal processes, Bioinformatics, vol. 27, no. 17, pp. 2453–2454, 2011.

[17] D. Lopes, H. Puga, J. Teixeira, and R. Lima, Blood flow simulations in patient-specific geometries of the carotid artery: a systematic review, Journal of Biomechanics, pp. 110019, 2020.

[18] S. Rashad, S. Sugiyama, K. Niizuma, K. Sato, H. Endo, S. Omodaka, Y. Matsumoto, M. Fujimura, and T. Tominaga, Impact of bifurcation angle and inflow coefficient on the rupture risk of bifurcation type basilar artery tip aneurysms, Journal of neurosurgery, vol. 128, no. 3, pp. 723–730, 2017.

[19] C. Shen, R. Gharleghi, D. D. Li, M. Stevens, S. Dokos, and S. Beier, Secondary flow in bifurcations–important effects of curvature, bifurcation angle and stents, Journal of Biomechanics, vol. 129, pp. 110755, 2021.

[20] S. Vinchurkar and P. W. Longest, Evaluation of hexahedral, prismatic and hybrid mesh styles for simulating respiratory aerosol dynamics, Computers & Fluids, vol. 37, no. 3, pp. 317–331, 2008.

[21] S.N. Wright, P. Kochunov, F. Mut, M. Bergamino, K. M. Brown, J. C. Mazziotta, A. W. Toga, J. R. Cebral, and G. A. Ascoli, Digital reconstruction and morphometric analysis of human brain arterial vasculature from magnetic resonance angiography, Neuroimage, vol. 82, pp. 170–181, 2013.

[22] H. Zakaria, A. M. Robertson, and C. W. Kerber, A parametric model for studies of flow in arterial bifurcations, Annals of biomedical Engineering, vol. 36, pp. 1515, 2008.