The 9th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2024
3–7 June 2024, Lisboa, Portugal

SKELETON-BASED BLOCK-STRUCTURED MESH GENERATION OF VASCULAR DOMAINS

Domagoj Bošnjak[1], Thomas-Peter Fries[2]

Institute of Structural Analysis
Graz University of Technology
Lessingstrasse 25, Graz, 8010, Austria
e-mail: [1] bosnjak@tugraz.at
e-mail: [2] fries@tugraz.at

Key words: Structured hexahedral meshing, block-structures, vascular modeling

Summary. Mesh generation of patient-specific blood vessels represents a key step in finite-element-based numerical simulations. In cardiovascular applications, they can serve as a non-intrusive analysis tool to support physicians in understanding and decision-making. The focus herein lies on patient-specific blood vessels, both healthy as well as non-healthy ones, which may have significantly altered morphologies. The input of the algorithm is the surface mesh of the original domain, which is then represented implicitly, by a signed distance function. From the same surface mesh, we generate a topological skeleton. It is used as as intermediate step to facilitate the generation of a block-structure. Thereafter, transfinite maps are applied to complete the mesh generation. The approach offers large flexibility in terms of spatial refinement as well as mesh order, and satisfactory mesh quality. Suitability for anisotropic refinement enables the generation of boundary layers without increasing the number of mesh elements, making them imminently suitable for fluid simulations. Another useful feature of the approach is the easy incorporation of semantic information to individual mesh elements, enabling straightforward assignment of local properties. The meshing algorithm is verified on examples from an open data repository.

1 Introduction

Blood vessels present a demanding class of domains with respect to mesh generation, which is a crucial part of numerical simulations based on the finite element method (FEM). Whether one considers blood flow simulations in the context of computational fluid dynamics (CFD) [22, 11], or fluid-structure interaction (FSI) [5, 19], an adequate mesh of the computational domain is a necessity. Usually, this is a cumbersome task requiring multiple levels of interaction from the user. Even then, obtaining a mesh that is not only usable, but also of high quality, is not straightforward or guaranteed. Unstructured meshing is still predominant when meshing complex domains, offering a good rate of success in producing valid meshes, while requiring only the domain's surface. Additional effort is usually necessary to obtain the final simulation-ready mesh, involving, e.g., parameter tuning or the post-production of boundary layers. The resulting unstructured meshes are commonly fairly large, in terms of the number of nodes and elements, easily going into the millions [17].

An alternative is discussed herein, namely a structured meshing approach [23]. At the price of having to narrow down to a specific class of domains, we obtain more control over the quality as well as the size of the mesh, which plays a crucial role in certain applications. Examples include sensitivity analysis and uncertainty quantification, where one varies certain parameters of the domain and thus requires multiple simulation results, e.g., in the order of tens or hundreds of thousands. There, a large mesh presents a direct bottleneck.

In order to combat this issue, we follow a *block-structured* meshing approach, meaning that we firstly seek to generate a block-structure [3, 1], which is a representation of the domain in the form of coarse blocks. In this work, blocks are valid hexahedral finite elements. Note that the only requirement on the block-structure is that it is topologically equivalent to the domain of interest, not necessarily representative of, e.g., the curvature. In vascular domains, the topology may also be easily represented via a centerline, also called a *skeleton* [4]. Hence, a skeleton is a natural basis for the generation of the block-structure, as well as a useful tool for such domains in general [12]. This has been documented in other works [13, 11, 7, 9]. To proceed with the mesh generation, we obtain an *implicit surface description* from the input data, which most often comes in the form of a surface mesh. To complete the mesh generation, *transfinite maps* [20] are applied to obtain nodal positions, and a connectivity matrix is generated thereafter. With respect to spatial refinement, the mesh can be as coarse as the block-structure. Additionally, the approach supports the generation of higher-order meshes, relevant for p-FEM.

Finally, we consider our algorithm's capability of performing *semantic meshing*, i.e, labeling elements based on a given criterion, often done in the context of segmentation [16], which is a key step in obtaining patient-specific surface meshes. Namely, it is often desirable to equip different properties to different parts of a given domain, e.g., when one wants to consider multiple types of materials, or separate a fluid domain from a solid domain. Hence, it is useful to be able to tag certain areas in the domain, so as to automatically differentiate between them on the mesh level. In our approach, it is enough to tag the skeleton, which is an easy process, since the skeletons are relatively coarse. Thereafter, the tags are carried over to the block-structure, and subsequently to the mesh. Moreover, the block-structure is designed to have a boundary layer, meaning that it is relatively straightforward to assign different layer properties to the domain as well. Thus, semantic meshing comes at a fairly low price, in the proposed approach.

The rest of the paper consists of two main sections. The first one covers the data acquisition and preprocessing as well as the mesh generation itself, and the following section contains the examples and the accompanying discussion.

2 Methods

2.1 Data acquisition

The input data for all of the examples in the paper were obtained from the Vascular Model Repository [21], an open-source database of cardiovascular domains. We made use of the surface meshes as well as the skeletons, where they were available. Preprocessing of the skeletons was necessary, e.g., to remove some skeleton nodes which are outside of the domain or to merge separated branches and junctions. The surface meshes, which are used as the input for our algorithm, are of very good quality. Minimal processing was performed in Meshlab [8] to perform smoothing around junctions, via Loop subdivision surfaces [15]. For certain operations, it is not crucial to have the smooth surface of the domain, which may contain many elements. Therefore, for the purpose of speeding the algorithm up, we also made use of coarser surface representations obtained from the original surfaces, using the quadric edge collapse decimation [10] implemented in Meshlab. Fig. 1 shows the examples as they are in the Vascular Model Repository, and Tab. 1 the relevant input data information.

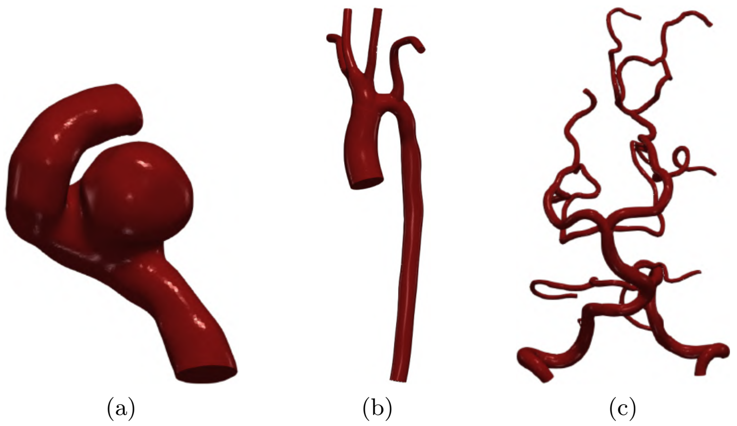


Figure 1: Patient-specific models obtained from the Vascular Model Repository: (a) cerebral aneurysm, (b) aortic coarctation, and (c) a healthy cerebral artery.

Case	# of skel. nodes	# of skel. segments	# of surf. nodes	# of surf. elements
Aneurysm	11	10	1 892	3 780
Coarctation	58	57	81 250	162 496
Cerebral	576	575	49 367	98 730

Table 1: Input data size: number of skeleton nodes and segments, and the number of elements and nodes in the corresponding surface meshes.

In general, skeleton extraction from the input surface can be performed with various tools and methodologies, most notably the *vascular modeling toolkit* [2].

2.2 Mesh generation

In order to be able to precisely map points to the surface, an implicit description is generated from the input triangle surface mesh. We use a typical signed distance function, which measures the distance of a given point to the input surface mesh, but with a negative sign in case the point is outside of the domain.

The mesh generation procedure consists of two steps, assuming a skeleton and an implicit surface description are present. The first part involves the generation of a block-structure based on the skeleton. Various skeleton configurations that occur in the domains of interest need to be considered, in order to design targeted block-structure schemes. To this end, we consider a fundamental differentiation of skeleton points, based on the respective number of neighbours. In case of 1 or 2 neighbours, a point is referred to as a non-junction point, otherwise it is a junction point. For junction points with 3 neighbours, which is the most common case in the aorta, we follow the scheme from [6], and generate a reference line $u = u_1 \times u_2 + u_2 \times u_3 + u_3 \times u_1$, where u_i

are defined as direction vectors from the neighbours of the given point. This is then used a basis to generate connected cross-sections facing the neighbours, shown in Fig. 2. Moreover, the planes where the cross-sections are placed can be rotated around the said reference line. A sensible approach is the plane where the cross-section (on average) remains closest to the skeleton point, in order to better represent junctions with significantly different neighbour radiuses.

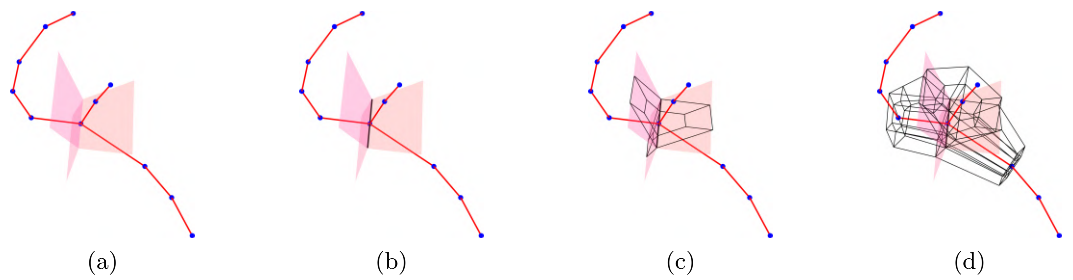


Figure 2: Generation of cross-sections around junction points: (a) planes based on the three pairs of neighbouring points, (b) the reference line which is the intersection of the three planes, (c) the three (half) cross-sections facing the neighbours, and (d) the final junction block-structure scheme.

For non-junction points, we determine the planes for cross-section placement based on their neighbours. This is much more straightforward, as a point with two neighbours is assigned exactly the plane between the two segments. Additionally, a bias may be introduced in case of a large difference between segment lengths or local radiuses. Endpoints, i.e., points with only one neighbour are assigned a plane with a normal vector of precisely the only associated segment. Their respective reference lines are interpolated from the fixed reference lines of the junctions in the skeleton, to minimize twisting. Fig 3 showcases the procedure.

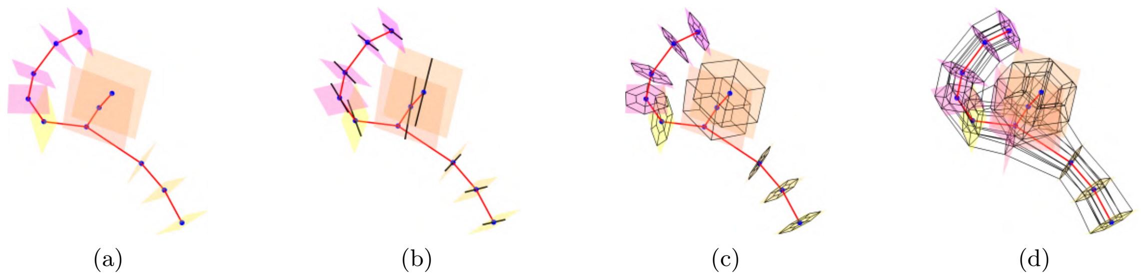


Figure 3: Generation of cross-sections around non-junction points: (a) planes based on the neighbours of each point, (b) reference lines interpolated from the junction reference line, (c) the cross-sections, and (d) the final block-structure obtained by connecting the cross-sections into blocks.

Junctions, alongside abrupt radius changes, represent the most difficult skeleton configurations to tackle. This is true in the general context of skeleton-based approaches [7, 13, 11]. Assuming the junction in question is more complex, i.e., has more than three neighbours, the same approach essentially works in case the junction is nearly planar. Fig.4 shows different cases of junctions, namely a 5-furcation, neighbouring junctions, and a junction with significantly differing neighbour radiuses. All of the configurations stem from examples found in the examples section.

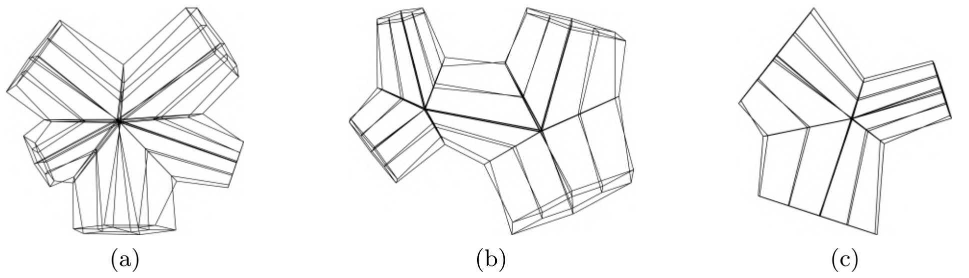


Figure 4: Various complex configurations of the blocks: (a) a (nearly planar) 5-furcation, (b) two connected junctions, and (c) a junction whose neighbours have vastly different radiuses.

In the final part of the mesh generation procedure, we make use of the implicit surface description and the block-structure via transfinite maps. Depending on the desired spatial refinement level and mesh order, the required number of points is placed on each edge of the block-structure, and projected to the surface in case the original edge was on the surface of the domain. Then, we apply transfinite maps for quads to generate the inner points on the faces of the block-structure. Again, for each face that is on the surface of the block-structure, its points are thereafter projected to the surface of the domain. Finally, transfinite maps for hexahedra are applied to generate the points in each block interior. After the generation of a connectivity matrix, the meshing process is completed. For more details on this step we refer the reader to [20, 7, 6].

As an example of a local property that can be induced on the block-structure level, we consider anisotropic refinement, which is useful for the treatment of boundary layers in fluid simulations. Rather than adding new layers of thin elements to the mesh, a non-uniform node distribution is induced in the block-structure, and straightforwardly propagated to the final node positioning through the transfinite maps for quads and hexas. Fig. 5 illustrates the difference in the cross-section caused by modified refinement. Note that both meshes have the same amount of elements.

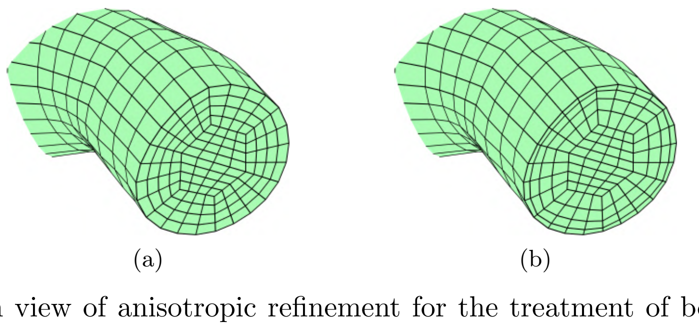


Figure 5: Zoomed-in view of anisotropic refinement for the treatment of boundary layers: (a) an isotropically refined mesh, and (b) an anisotropically refined mesh.

2.3 Semantic meshing

Being able to differentiate between different parts of the domain is useful for assigning spatially dependent attributes, such as varying material properties. In a block-structured approach, it is clearly known which elements stem from which blocks. Moreover, since the block-structure is generated from the skeleton, it is possible to associate individual blocks to skeleton points and segments. Therefore, if one labeled the skeleton, which is a rather coarse object in this particular approach (11 points for the cerebral aneurysm, see Tab. 1), this labeling could be transferred to individual mesh elements, greatly alleviating the difficulty of the process. As a concrete example, we consider simply separating a cerebral aneurysm from the rest of the cerebral artery, as shown in Fig. 6.

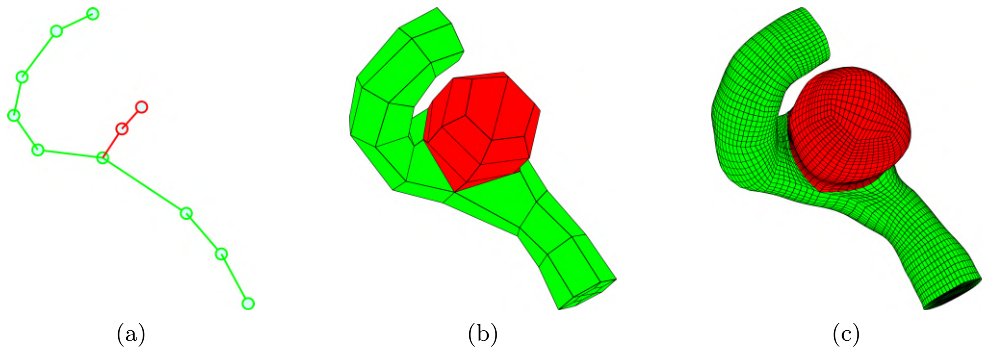


Figure 6: Semantic mesh labeling: (a) color-coded skeleton labeling, with the red part representing the aneurysm, (b) the analogously color-coded block-structure, passed on from the skeleton, and (c) the color-coded mesh.

3 Examples

In this section we highlight several mesh examples generated by our algorithm. All of the presented domains are in fact patient-specific. The first example showcases a cerebral aneurysm. Namely, the cerebral arteries are vital in the process of supplying blood to the spine as well as the brain. An untreated aneurysm may rupture, which may lead to serious health problems. Fig. 7 shows the skeleton, block-structure and the mesh examples for the aneurysm case. The next example is a coarctation of the aorta, showcased analogously to the previous case, in Fig. 8. Patients with this condition exhibit a narrowing of the aorta, most often just after the sub-vessels at the top. The condition can cause various issues, in particular if the narrowing is severe. The final example is an extensive model of a healthy cerebral artery, again shown in the same format as the previous examples, in Fig. 9

In Fig. 10, we show the histograms of scaled Jacobian values for each of the mesh examples. The scaled Jacobian is a widely used mesh quality metric[14, 18], where an ideal element has the value of 1, and negative values imply invalid elements.

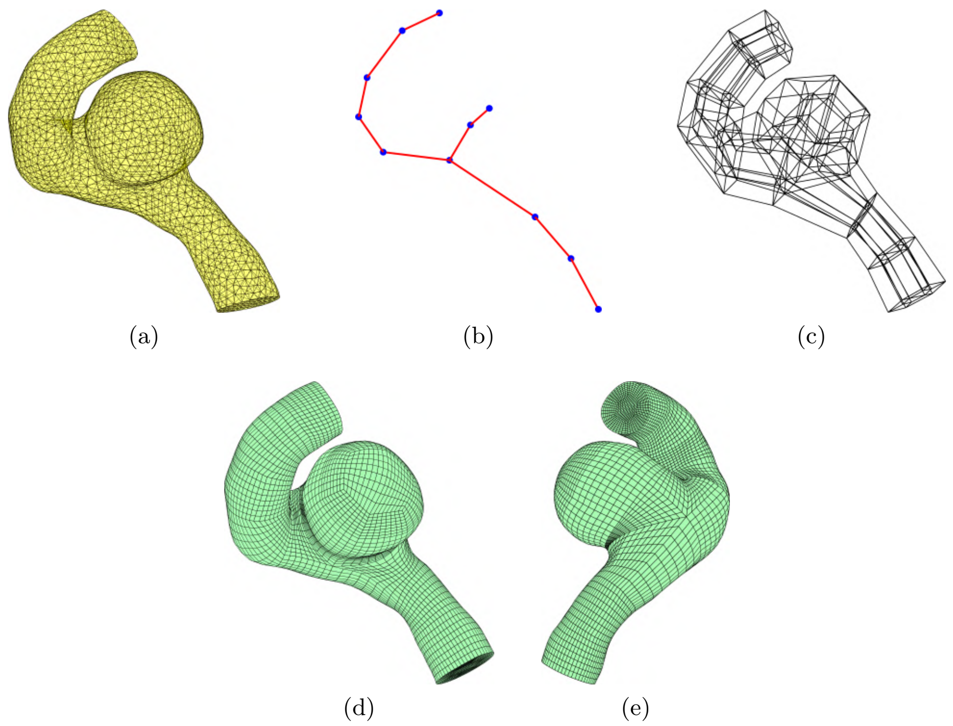


Figure 7: First mesh example showing a cerebral aneurysm: (a) the input triangle surface mesh, (b) the skeleton, (c) the block-structure, and (d)-(e) different views of the structured hexahedral mesh.

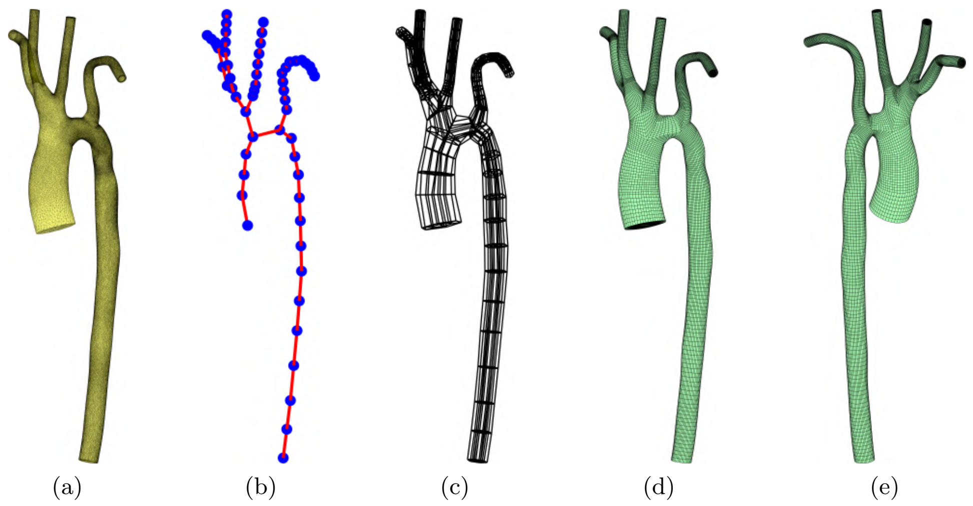
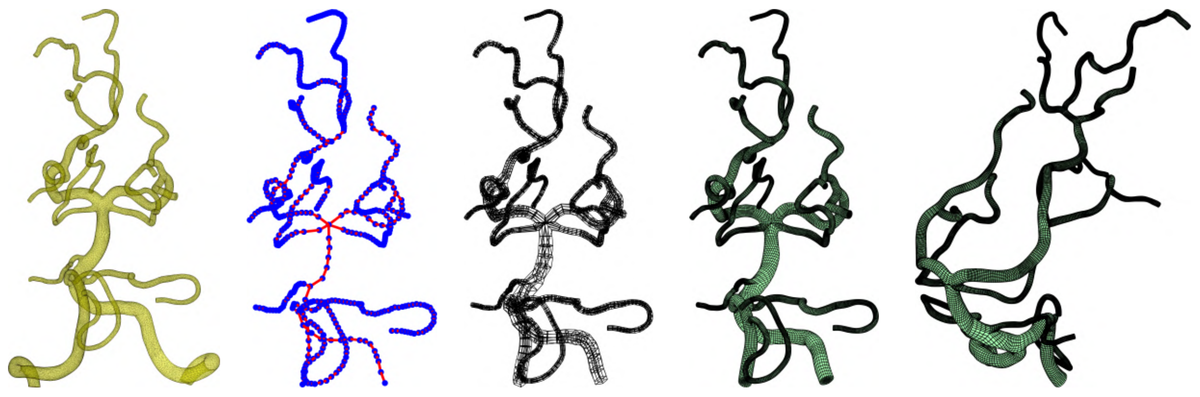


Figure 8: Second mesh example illustrating an aortic coarctation: (a) the input triangle surface mesh, (b) the skeleton, (c) the block-structure, and (d)-(e) different views of the structured hexahedral mesh.



(a) (b) (c) (d) (e)

Figure 9: Third mesh example containing a highly complex configuration of a healthy cerebral artery: (a) the input triangle surface mesh, (b) the skeleton, (c) the block-structure, and (d)-(e) different views of the structured hexahedral mesh.

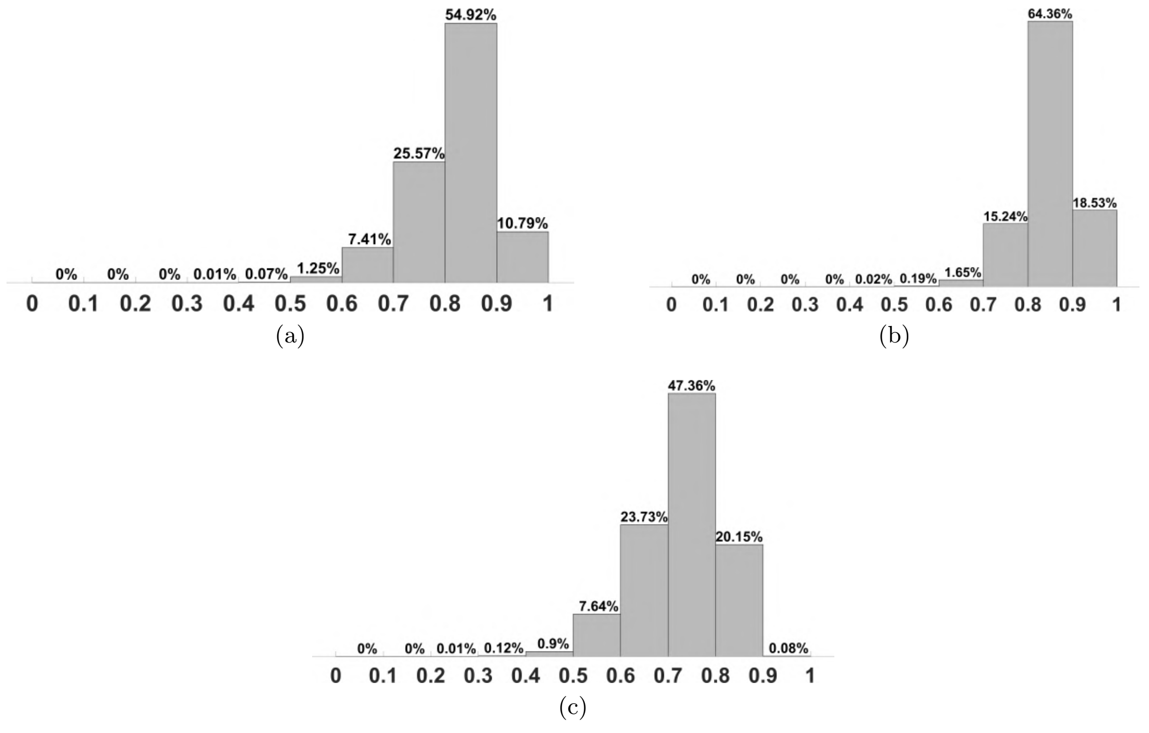


Figure 10: Mesh quality evaluation using histograms of element-wise value of the scaled Jacobian for the three cases, where 1 indicates an ideal element.

4 Conclusion

Block-structured meshing offers unique advantages of coarse mesh generation, local editing, and straightforward element tagging. In case any of those are properties of major significance, as is often the case in cardiovascular applications, we argue that a block-structured approach is worth the additional effort, compared to opting for a traditional unstructured method. The domains in question, i.e., blood vessels, can in most cases be adequately described by their skeleton, exhibiting a certain underlying structure even in the non-healthy cases. This makes them suitable for structured meshing approaches such as this one. Overall, high mesh quality is obtained, even though no particular post-processing or optimization was performed on the resulting hexahedral meshes, which would be a natural next step.

Data availability

The meshes shown in this paper, as well as a more extensive and consistently growing set of meshes will be made available post-publication.

Acknowledgement

The authors gratefully acknowledge Graz University of Technology, Austria for the financial support of the LEAD-project: Mechanics, Modeling and Simulation of Aortic Dissection.

REFERENCES

[1] Z. Ali, P. Dhanasekaran, P. Tucker, R. Watson, and S. Shahpar. Optimal multi-block mesh generation for cfd. *Intern. J. of Comp. Fluid Dynamics*, pages 195–213, 2017.

[2] L. Antiga, M. Piccinelli, L. Botti, B. Ene-Iordache, A. Remuzzi, and D. Steinman. An image-based modeling framework for patient-specific computational hemodynamics. *Med. Biol. Eng. Comput.*, 46(11):1097–1112, 2008.

[3] C. Armstrong, H. Fogg, C. Tierney, and T. Robinson. Common themes in multi-block structured quad/hex mesh generation. *Procedia Eng.*, 124:70–82, 2015.

[4] O. Au, C. Tai, H. Chu, D. Cohen-Or, and T. Lee. Skeleton extraction by mesh contraction. *ACM Trans. Graph.*, 27(3):1–10, 2008.

[5] Y. Bazilevs, J. Gohean, T. Hughes, R. Moser, and Y. Zhang. Patient-specific isogeometric fluid–structure interaction analysis of thoracic aortic blood flow due to implantation of the jarvik 2000 left ventricular assist device. *Comp. Methods Appl. Mech. Engrg.*, 198(45):3534–3550, 2009.

[6] D. Bošnjak and T. Fries. Block-structured mesh generation from implicit geometries for cardiovascular applications. *PAMM*, 23(2), Sept. 2023.

[7] D. Bošnjak, A. Pepe, R. Schussnig, D. Schmalstieg, and T. Fries. Higher-order block-structured hex meshing of tubular structures. *Eng. with Comp.*, 40:931–951, 05 2023.

[8] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. MeshLab: an Open-Source Mesh Processing Tool. In *Eurographics Italian Chapter Conference.* The Eurographics Association, 2008.

[9] M. Decroocq, C. Frindel, P. Rougé, M. Ohta, and G. Lavoué. Modeling and hexahedral meshing of cerebral arterial networks from centerlines. *Med. Image Anal.*, 89:102912, Oct. 2023.

[10] M. Garland and P. Heckbert. Surface simplification using quadric error metrics. *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 1997, 07 1997.

[11] M. Ghaffari, K. Tangen, A. Alaraj, X. Du, F. Charbel, and A. Linninger. Large-scale subject-specific cerebral arterial tree modeling using automated parametric mesh generation for blood flow simulation. *Comput. Biol. Med.*, 91, 2017.

[12] D. Hahn. Coronary artery centerline extraction in 3d slicer using vmtk based tools. *Department of Medical Informatics, University of Heidelberg*, page 105, 2010.

[13] M. Livesu, A. Muntoni, E. Puppo, and R. Scateni. Skeleton-driven adaptive hexahedral meshing of tubular shapes. *Comput. Graph. Forum*, 35(7):237–246, 2016.

[14] S. Lo. Finite element mesh generation and adaptive meshing. *Prog. struct. eng. mater.*, 4(4):381–399, 2002.

[15] C. T. Loop. Smooth subdivision surfaces based on triangles. 1987.

[16] J. Lv, X. Chen, J. Huang, and H. Bao. Semi-supervised mesh segmentation and labeling. *Computer Graphics Forum*, 31(7):2241–2248, 2012.

[17] S. Owen. A survey of unstructured mesh generation technology. *Proc. International Meshing Roundtable*, 3, 05 2000.

[18] N. Pietroni, M. Campen, A. Sheffer, G. Cherchi, D. Bommes, X. Gao, R. Scateni, F. Ledoux, J. Remacle, and M. Livesu. Hex-mesh generation and processing: A survey. *ACM Trans. Graph.*, 42(2), 2022.

[19] R. Schussnig, M. Rolf-Pissarczyk, G. Holzapfel, and T. Fries. Fluid-structure interaction simulations of aortic dissection. *PAMM*, 20, 2021.

[20] P. Šolín, K. Segeth, and I. Doležel. *Higher-order finite element methods*. CRC Press, Boca Raton, FL, 2003.

[21] N. Wilson, A. Ortiz, and A. Johnson. The vascular model repository: A public resource of medical imaging data and blood flow simulation results. *J. Med. Devices*, 7:0409231–409231, 12 2013.

[22] Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, and T. Hughes. Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. *Comp. Methods Appl. Mech. Engrg.*, 196:2943–2959, 2007.

[23] O. Zienkiewicz and R. Taylor. *The Finite Element Method: The Basis*, volume 1. Butterworth-Heinemann, Oxford, 2000.