EG UK Computer Graphics & Visual Computing (2019)
G. K. L. Tam and J. C. Roberts (Editors)

Generating High Fidelity Surface Meshes of Neocortical Neurons using Skin Modifiers

Marwan Abdellah[†], Cyrille Favreau, Juan Hernando, Samuel Lapere and Felix Schürmann[‡]

Blue Brain Project, Ecole Polytechnique Fédérale de Lausanne (EPFL)
Geneva, Switzerland

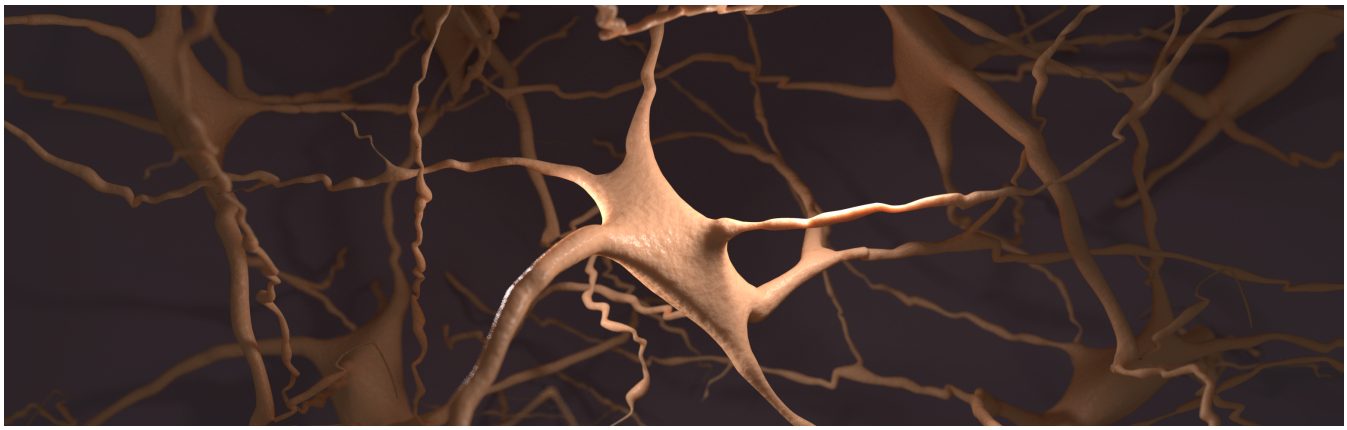

Figure 1: *A network of neocortical neuronal meshes reconstructed with the proposed algorithm using the skin modifier in Blender.*

Abstract

We present the results of exploring the capabilities of skinning modifiers to generate high fidelity polygonal surface meshes of neurons from their morphological skeletons that are segmented from optical microscopy slides. Our algorithm is implemented in Blender as an add-on relying on its standard Python API. The implementation is also integrated into an open source domain specific framework, NeuroMorphoVis, that is used to visualize and analyze neuronal morphologies available from the neuroscientific community. Our technique is applied to create meshes for a set of neurons with 55 different morphologies reconstructed from the neocortex of a 14-days-old rat. The generated meshes are used to visualize full compartmental simulations of neocortical activity for analysis purposes and also to create high quality scientific illustrations of in silico neuronal circuits for media production with physically-based path tracers.

1. Introduction

Recent years have witnessed a growing interest in exploiting the power of *visual computing* to accelerate the pace of neuroscience research, aiming at advancing our understanding of the mammalian—and potentially the human—brain. A fundamental approach in neuroscience is studying the morphology of individual neurons, the principal information processing units in the brain, where the morphology of a neuron influences network topology and dynamics [Abd17]. Visualizing neurons is not trivial; they are characterized by thin, long and interleaving arborizations spanning different regions of the brain. Therefore, they have a very large spatial extent with low space occupancy, which makes it virtually impossible to visually recognize and identify a single neuron in a complex network scene [EBA*12]. Moreover, they have unalike or non-identical shapes, where every neuron has its unique branching. Nevertheless, they can be classified into multiple morphological categories based on the idiosyncratic shapes of their dendritic and axonal trees [KRS*19]. Existing domain-specific applications use various methods to visualize neurons using discon-

† *marwan.abdellah@epfl.ch*
‡ *felix.schuermann@epfl.ch*

DOI: 10.2312/cgvc.20191257

nected cylinders [GSS07], tapered cylinders with spheres at the branching points [GG90], polylines [AHE*18] and connected surface meshes [HSP12; GBM*17]. Using polygonal meshes might be relatively expensive, but it is the most convenient approach to visualize membrane potentials, Calcium concentrations and also to create high quality multimedia content.

We present a method for generating high fidelity polygonal surface mesh models that reflect the cell membranes of neurons from their morphological skeletons using the *Skin Modifier* in Blender [Fou16; Hes07]. To verify the consistency of the algorithm, our implementation is applied to a diverse set of neurons that reflect 55 different morphological types found in the somatosensory cortex (part of the neocortex) of juvenile rats [MMR*15].

1.1. Background

There are two principal sources of neuronal morphologies. They are either segmented from brain tissue samples in wet lab experiments or grown synthetically in computer simulations. Biological morphologies are reconstructed and digitized from optical microscopy stacks with manual, semiautomated or even automated techniques using machine learning methods [GG90; Mei10; PA13]. On average, these biologically-reconstructed skeletons contain several thousands of morphological samples [LHH*12]. Synthetic morphologies are generated using some rules and stochastic sampling methods in a process called *neuronal morphogenesis* [KTvH*09; FFB*10]. These synthetic, yet biologically-inspired, skeletons are normally oversampled and contain many more samples than the biological ones. The end result from both sources is the same: a raw file containing a *hierarchical morphology skeleton* described as a set of connected samples stored in a common format. Figure 2 shows a schematic view of a simple neuronal morphology with three arbors truncated at branching order of two.

The morphology has two distinct structures: a cell body (or soma) and arbors (or neurites). Traditionally, the soma is simply represented by a centroid, average radius and two-dimensional projective profile. Advanced reconstructions have a more realistic soma shape with multiple contours at different depth of fields which could be used to create accurate three-dimensional profile relying on Poisson surface reconstruction [BTS*14]. Nevertheless, the majority of publicly available morphologies lacks this information. The arbors are classified into axons, basal and apical dendrites, where each is composed of a set of samples (or points) and each sample defines a three-dimensional Cartesian point, radius, arbor type and some connectivity information which links this sample to a parent or child one. Each two consecutive samples form a segment and a set of segments between two branching points constructs a section.

There are two approaches followed to reconstruct a closed surface mesh of the entire neuron from this description. The first one uses an initial sphere that is sampled at certain points on its surface to identify a polygon that can be extruded to form the arbor [LHH*12]. The other approach divides the morphology into multiple components, creates a corresponding mesh to each individual component and finally welds all the meshes together to create the final mesh [BMB*13; GBM*17]. These approaches are summarized in the following part.

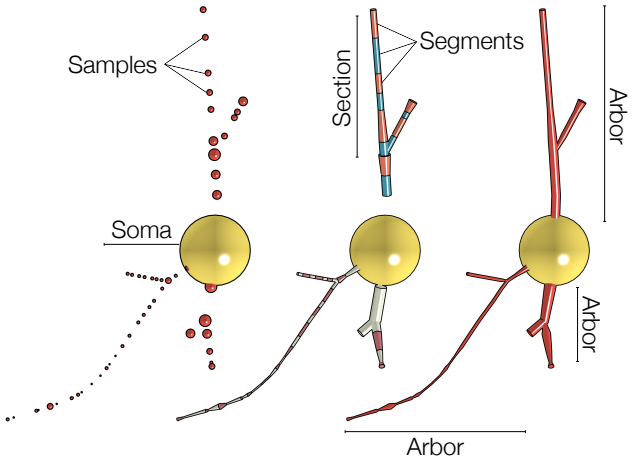

Figure 2: *A schematic view of a neuronal morphology skeleton. The arbors emanate from a cell body (soma), where each arbor is composed of a set of raw samples. Two connected sampled define a segment, and a set of segments between two branching points defines a section. Multiple connected sections define an arbor.*

1.2. Related Work

Creating polygonal surfaces from skeletons is a significant research topic in computer graphics. It has been applied in various domains, for example, to tubular structures such as trees or blood vessels [XMF13]. However, its application to neuronal skeletons has only been addressed recently in few research studies to either create high fidelity surface meshes for visualization and analysis or to generate watertight and tetrahedral volume meshes for simulating sub-cellular events. Earlier approaches have used simplified models, such as tube-based rendering, to visualize the neurons. These models might have low memory footprint, but their limitations cannot be tolerated. They do not provide accurate shapes of the cell bodies and they cannot be used for accurate visualization of compartmental simulations that require a per-vertex mapping scheme.

Lasserre et al. have presented a method based on extruding a control mesh that can be used to generate a detailed mesh with a smoothing operation [LHH*12]. Their approach was limited in several aspects: the code was not available for reproducibility, the algorithm relies on extruding a coarse quadrilateral mesh around the skeleton which makes it extremely slow, the code was implemented in a proprietary software product (Maya) using a specific scripting language called MEL [WK05], and finally the quality of the reconstructed meshes was questionable due to the presence of sharp edges along the branches.

Brito et al. have presented an interactive tool, called Neuronize, capable of building adaptive and multi-resolution neuronal meshes with realistic somata that are reconstructed on a physically-plausible basis using Hooke's law and mass spring models [BMB*13]. Based on Neuronize, Garcia et al. have presented another hardware-accelerated tool, called NeuroTessMesh, for generating neuronal meshes and refining them on-the-fly using a Finite Element Method and tessellation shaders, allowing to reconstruct

digital models of large scale neuronal circuits in reasonable amount of time [GBM*17]. Neuronize and NeuroTessMesh were able to overcome several limitations of other traditional tools (such as NeuroConstruct [GSS07] and Neurolucida [GG90]). In contrast to the symbolic somata representations using spheres or disks, they could generate realistic three-dimensional somata profiles. They can also generate a single connected and closed surface around the branching points that would result in a continuous surface over the entire mesh. Nevertheless, they were limited in two aspects: the quality of the branching was low even for highly tessellated meshes and the close samples were causing zigzag-like artifacts and sharp edges, which would limit their usability to render high quality scientific visualizations.

Abdellah et al. have developed a method to create neuron meshes that can be used for creating volumetric models of large scale digital brain circuits to be used in optical microscopy simulations [AHA*17]. The meshes were *piecewise watertight*, i.e. for a set of connected sections along a given branch, but the resulting mesh for a single arbor was not watertight. Nevertheless, this method uses soft body objects and the physics engine in Blender to simulate the growth of the cell bodies into highly realistic meshes.

There are also other approaches that have focused on generating watertight polygonal meshes that can be used to build tetrahedral volume meshes for offline reaction-diffusion simulations such as those presented by McDougal et al. [MHL13], Morschel et al. [MBQ17] and Koene et al. [KTvH*09]. The resulting meshes are watertight and two-manifold, but they cannot be used for interactive visualization and analysis for two reasons: they have extremely high tessellation and they ignore the realistic and organic appearance of neuronal morphologies.

In this current work, we investigate using the skin modifier in Blender to reconstruct high fidelity neuronal meshes with organic looking branching and extending previous implementations [AHA*17; AHE*18] to create highly plausible somata shapes, trying to fill a gap that is still largely unfulfilled.

2. Neuronal Meshing with Skin Modifiers

Skin modifiers generate *skinned* surfaces from input skeletons (hierarchical structures with either cyclic or acyclic graphs) that are composed of a set of connected vertices and their edges. They have significant applications in computer graphics in the area of surface based deformations for animation. They create low-tessellated or coarse meshes that are used later to reconstruct smooth organic shapes with arbitrary topology. The resulting meshes have quadrilateral faces and probably some triangular ones around bifurcating branches and intersections. The realism of the resulting surface is improved if a per-vertex radius is used instead of a unique radius for all the vertices along the skeleton [Vil14]. Figure 3 shows an example of building a surface mesh of a small part of an acyclic morphology based on the skin modifier. In the following sections, we discuss our approach that leverages the skin modifier to generate a realistic and high fidelity neuronal surface mesh with organic looking arbors from a given morphological skeleton.

2.1. Realistic Somata Generation

An exact surface mesh of the soma can be only reconstructed if its three-dimensional profile is traced during the segmentation procedure [LBB*15]. This profile approximates an input point cloud where a mesh can be obtained using Poisson surface reconstruction [BTS*14; KBH06]. Unfortunately, this three-dimensional profile is only obtained in advanced and recent traces. In some morphologies, somata are represented with a centroid and average radius only. In other cases, the radius of the soma is not even available, i.e. set to zero. Given these conditions, and aiming to have a generic solution that works for all the morphologies, the most plausible approach to create realistic somata is based on mass spring models and Hooke's law to deform a soft body object with a series of pulling forces to simulate the growth of arbors from the soma [BMB*13; AHA*17].

Brito et al. have presented their custom and domain specific software, Neuronize, to reconstruct somata, however it is written in C++ [BMB*13]. Abdellah et al. have adapted the same approach to generate somata using soft body objects and the physics engine in Blender [AHA*17]. Their implementation was open sourced and integrated later in NeuroMorphoVis [AHE*18], but the somata were created as individual objects without being connected to the arbors. Moreover, there were some artifacts around the morphological samples that connect the arbors to the soma body. Since we use the skin modifiers in Blender, it was convenient to rely on their Soma Reconstruction Toolbox and extend it to meet our objectives. We therefore extended their implementation and added two other features to improve the visual quality of the reconstructed soma mesh. We carefully processed all the samples that are located within the extent of the soma to ensure the absence of intersecting branches with the soma or duplicated samples that can cause irregularities after the bridging of the arbors to the soma. We used 200 simulation steps instead of 100 only. We also added the option to bridge the meshes of each arbor to the corresponding extrusion faces on the soma mesh to yield a single connected surface mesh of the entire neuron. Figure 4 shows the progressive reconstruction of the soma of a pyramidal neuron evolving from a pre-processed sphere to a highly realistic soma mesh. A video showing the entire sequence is available in the supplementary material.

2.2. Arbors Meshing

At this stage, the skin modifier is used to create a high fidelity mesh for each arbor in the morphology and then connect it to the corresponding face on the soma reconstructed in the previous stage. This step consists of the following operations:

1. Preprocessing the arbor to eliminate any severe artifacts that would cause any deformations in the final mesh.
2. Constructing a skeleton mesh base from the morphological samples of the arbor and their connectivity information.
3. Applying the skin modifier on the generated skeleton to obtain a coarse mesh with quadrilateral faces.
4. Applying the surface subdivision operator on the coarse mesh to obtain a smooth one.
5. Welding each arbor mesh to its corresponding face on the soma mesh using bridging.

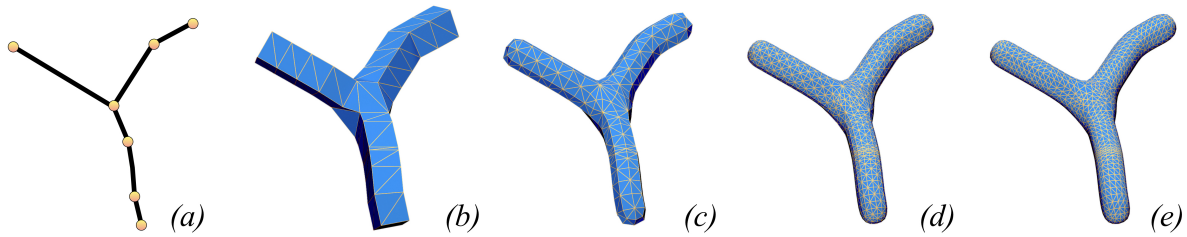

Figure 3: *Converting a part of a neuronal morphology into high quality mesh using the skin modifier in Blender. (a) The skeleton is created from a group of morphological samples and their connectivity information. (b) The skin modifier is applied to reconstruct an initial coarse mesh around the skeleton. A subdivision operator based on the Catmull-clark algorithm [HKD93] is applied to obtain a smooth mesh using subdivision levels of one (c) and two in (d). (e) The final mesh is obtained after converting all the quadrilateral faces into triangular polygons and adjusting normals.*

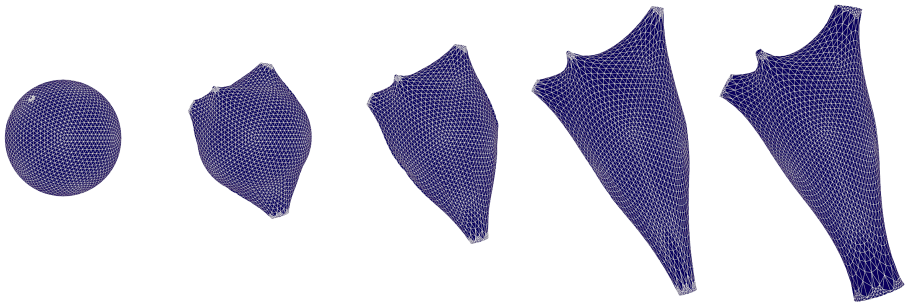

Figure 4: *Progressive reconstruction of the soma of a neocortical pyramidal neuron showing how the pulling forces deform a soft body sphere to reach a plausible profile that approximates the realistic one. The simulation uses the physics engine of Blender based on Hooke's law and mass spring models and runs for 200 steps to maximize the realism of the reconstructed profile.*

Preprocessing the Arbor Neuronal arbors might contain several artifacts due to the manual tracing procedure such as disconnected sections, abrupt changes in their diameters, self intersections and also intersections between the arbors and the soma. Such artifacts will certainly impact the quality of the reconstructed mesh. Therefore, each arbor is analyzed using a series of unit tests to detect and repair these artifacts before the skinning operation. We have also noticed that some morphologies, in particular the synthetic ones, are oversampled. This oversampling will potentially cause irregularities along the surface of the mesh, impact its visual appearance and accidentally increase its number of polygons. Consequently, we resample each section along the arbor to eliminate duplicated and overlapping samples.

Indexing Samples Each sample in the original morphology tree has a global index that is used to retrieve its data such as position, radius, type and parent. The extrusion and skinning procedures are applied per arbor, therefore there should be an exact correspondence between the samples of each arbor and the vertices of the skeleton generated from the extrusion operation, with which we can adjust the radii of the skinned object later before converting it to a control mesh. In addition to the *global index* parameter that is defined per sample, we also define another index called *local index* and set its value before extruding the root vertex to build the skeleton mesh of the arbor.

Extruding Arbor Skeleton Unlike the polygon extrusion technique presented by Lasserre et al. [LHH*12], which is extremely slow for dense morphologies, our approach extrudes the skeleton of each arbor on a *per-vertex basis* using the internal mesh editing API in Blender (or *bmesh*) [JLW10; Con17]. This skeleton (represented by a set of ordinary vertices and edges) is then used to efficiently generate a coarse mesh with organic shapes and arbitrary topology. The *bmesh* module is also more efficient than the default Blender Python API (*bpy*) allowing to add primitives, extend and build geometry without updating the scene.

Initially, an empty bmesh structure, that does not have any vertices, is created. Then, starting from the first sample on the root section, all the samples along the arbor tree are traversed with depth-first traversal. For each sample, a new vertex is created, linked to the bmesh object and connected to another vertex that corresponds to the parent sample. Owing to the indexing step, extruding the branching segments is straightforward; the index of a newly created vertex is automatically set to the local index of the corresponding sample along the arbor. After the traversal of the entire arbor, the

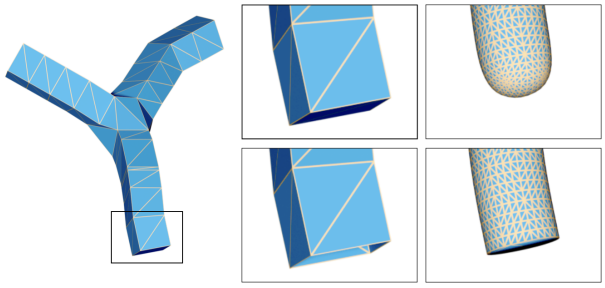

Figure 5: *A closeup showing the effect of smoothing the coarse mesh without (top) and with (bottom) removing the face that is used to bridge the arbor mesh with the soma.*

bmesh object is converted to a visible mesh that is linked to the scene where we can apply all the modifiers on. This resulting mesh does not have any polygons, but rather a set of vertices that are connected together with edges. This process is quite efficient; it takes only few milliseconds even for complex arbors.

Creating a Coarse Mesh After the generation of the skeleton mesh, a *Skin Modifier* is created and assigned to it. The initial radii of all the samples are set by default to one, and thus it is essential to update them to match the values specified in the original morphology. The skinned object is traversed again to adjust the radii at every node. Afterwards, the skin modifier is applied to create a control quadrilateral mesh that matches the skeleton of the arbor.

Mesh Smoothing The coarse mesh resulting from the skin modifier could be smoothed using the Catmull-Clark subdivision algorithm to create a high quality mesh [HKD93]. However, if this modifier is applied directly on the mesh, the quadrilateral face that is supposed to be welded to the soma will be subdivided into multiple non-planar faces, which would make it impossible to bridge the final mesh of the arbor to that of the soma. To overcome this issue, the connection face is selected and initially removed, then the subdivision modifier is applied. Afterwards, we use the *face-filling* operator to add a new face (with circular cross section) where it can be easily bridged to the corresponding one on the soma. This procedure is illustrated in Figure 5.

2.3. Connecting Soma to Arbors

To reconstruct a single mesh object that could represent a continuous surface of the membrane of the entire neuron, the individual meshes of the soma and the arbors must be welded together. We implement this step using the *Bridge Edge Loops* utility that is part of the *Edge Tools* in Blender. This bridging operation connects two closed edge loops (or faces) of the same mesh; it cannot be applied directly on multiple meshes. Therefore, the meshes of the arbors must be joined or merged together with the soma mesh into a single object before applying the bridging operation.

The mesh of each arbor is selected, set to active and converted to an editable object where we can select individual vertices and faces. The soma connection with the arbor occurs at the first sample of its root section. Therefore, the nearest face to this sample is selected on the arbor mesh. We also apply the same operation on the soma mesh and select the nearest face to the root sample of the arbor. After the selection of the faces, the two meshes are joined together and the bridging operator is applied to connect the two selected faces. This operation is performed later on the rest of the meshes of the arbors till the reconstruction of a mesh object that represents the final mesh of the entire neuron surface.

3. Results & Applications

We implemented our algorithm in the latest stable version of Blender (2.79) relying on its Python API. The code is also integrated in a domain specific framework called NeuroMorphoVis [AHE*18] that is used widely in *in silico* neuroscience research to validate, analyze and visualize individual neuronal morphologies upon their reconstruction from optical microscopy stacks. The code is freely available on GitHub.

Our algorithm is applied to generate surface meshes of a selection of neurons from a digital brain circuit that is reconstructed from the somatosensory cortex of a two-weeks-old rat to simulate a series of *in vivo* and *in vitro* experiments in a supercomputer [MMR*15; RCA*15]. The circuit contains ∼31,000 neurons representing 55 different morphological types spanning the different layers of a functional unit of the neocortex, the *neocortical column*. A collage of the resulting meshes for the 55 morphologies is shown in spectral colors in Figure 6. A summary of the abbreviations of all the morphological types (or m-types) and their classification is available in [MMR*15].

3.1. Performance

To assess the performance of our implementation, we benchmarked the entire pipeline for all the selected morphologies and reported detailed performance measures for every stage with respect to the number of samples per morphology in Figure 6. The performance was measured on a desktop with an Intel Core i7-6700 CPU running at 3.40 GHz.

The arbor reconstruction stage is composed of four principal steps: extruding the base skeleton mesh, applying the skin modifier to generate a control mesh, smoothing this control mesh and computing the vertex normals to apply smooth shading to the final mesh of the arbor. The execution time of the extrusion stages is directly proportional to the number of samples in the morphology. This stage takes, on average, a few tens of milliseconds, and less than a second for a layer V pyramidal cell that contains ∼8500 samples. In general, the performance of the skin modifier should be proportional as well to the number of samples, however, the *geometry of the branches* is another important factor that impacts the performance. Applying the skin modifier to the base skeleton mesh takes on average less than one second, but we notice that it takes more time for the L5_TTPC_1 and L6_IPC types compared to a L5_STPC morphology that has ∼2000 samples more due to the aforementioned reason. The Catmull-Clark subdivision operator takes approximately one second with a subdivision level of two,

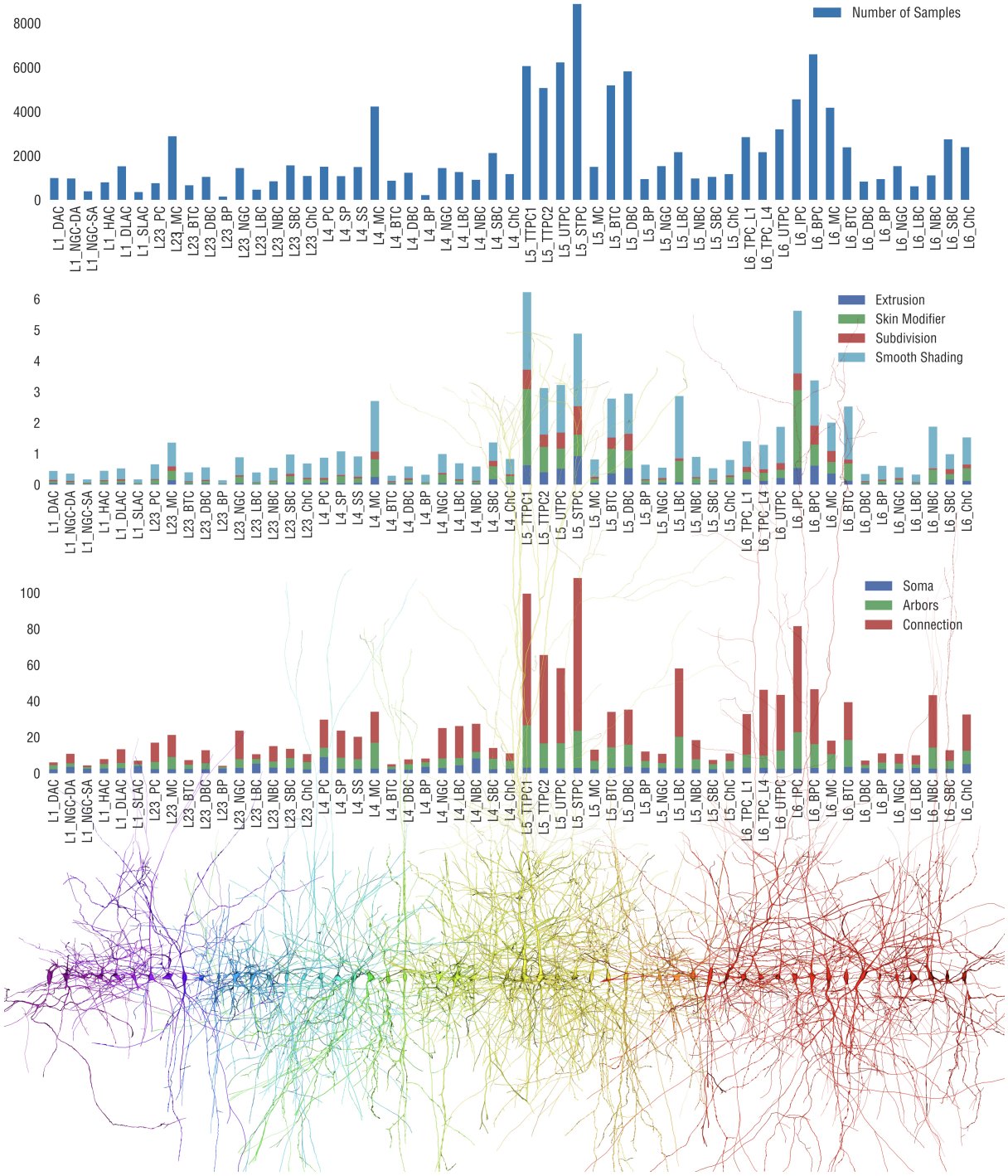

Figure 6: *Average performance profiles of reconstructing high quality surface meshes for exemplar neurons representing the 55 morphology types identified in the neocortical column [MMR*15]. The top graph shows the number of samples in the morphology. The middle shows detailed performance profiles for all the arbor reconstruction steps (in seconds). The bottom one shows the total time it takes to reconstruct the entire neuron mesh (in seconds).*

and finally, applying the smooth shading operator takes a considerable portion of the total arbor reconstruction time. This last step can be ignored if the normals are computed in the application that is used to visualize the meshes during the loading stage, so we make it as an optional parameter in our implementation.

The soma reconstruction takes between two and eight seconds. This time is proportional to the number of arbors in the morphology if the branches do not intersect around the soma, otherwise some processing is required to resolve the intersection and create the correct faces where the pulling occurs. However, bridging the soma mesh to the arbors is the most time consuming step in the whole pipeline. The bridging operator is relatively fast, but it cannot be applied in the object mode, therefore we must toggle the mesh from the object mode to the edit mode, select the connecting faces, apply the bridging operation and then toggle back to the object mode. This operation is relatively expensive and can take several tens of seconds for a complex morphology since it merges the contents of multiple mesh structures into a single mesh object.

Using the meshes to visualize simulations or to focus on intracellular components requires transparency, but in some cases bump mapping is applied to get certain visual effects, for example electron microscopy shading, for scientific illustrations. In these cases, we can ignore the bridging operation and obtain the same visual quality even for very zoomed closeups and high resolution images. Figure 7 shows a comparison between two meshes for the same neuron when the soma is bridged to the arbors and when the bridging is skipped.

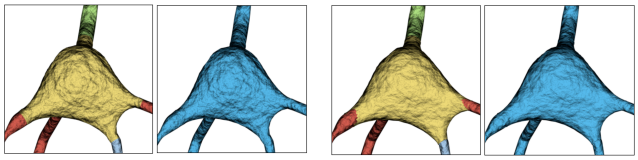

Figure 7: *Generating two meshes for the same neuron with (left) and without (right) bridging the arbors with the soma. The visual quality difference is minimized if a bump map shader with the same color is used to render all the components of the neuron.*

3.2. Comparison with Previous Approaches

Creating realistic and high fidelity surface meshes of neurons from their morphologies can be achieved if two major aspects are addressed: reconstructing accurate three-dimensional somata profiles which can approximate their actual shapes as seen on stained microscopy slides, in addition to meshing arbors with natural looking or organic branching. Reconstructing highly realistic somata profiles based on mass spring models and the Finite Element method has been provided by few studies such as [BMB*13; AHA*17; AHE*18; GBM*17], but having a smooth branching was missing. On the other hand, Lasserre et al. [LHH*12] have used quadrilateral extrusion to create a control mesh and smooth it with surface subdivision to synthesize realistic-looking arbors, but the somata were represented by sphere-like simplified approximations. Our approach combines both aspects to fulfill the requirements and fill the gap. Figure 8 compares the results for creating a pyramidal neuron mesh using NeuroTessMesh [GBM*17], the piecewise-watertight meshing algorithm by Abdellah et al. [AHE*18] and our implementation.

3.3. Visualizing Compartmental Simulations

One of the essential applications in simulation-based neuroscience is to visualize simulated electrical activity to analyze the models used to develop and test our hypotheses. This requires high quality surface meshes with which we can apply transparency algorithms to visualize the propagation of the electrical activity from one compartment to another. In Figure 9, the meshes created with our algorithm are mainly used to create high quality visualizations of full compartmental simulations of individual neurons and digitally reconstructed neocortical circuits [MMR*15]. To visualize these simulations, we used Brayns, an interactive ray tracing engine for rendering large scale neuronal networks where the simulation mapping is already implemented [FBN*15].

3.4. High Quality Neuroscientific Multimedia Generation

Another direct applications to our method is to create surface meshes with outstanding visual quality for creating artistic and multimedia content for scientific media production from a raw morphology file without any manual processing. Figures 1 and 10 show some examples of creating high quality images with single and network of neurons rendered with path tracing in Cycles [Val13].

4. Conclusion

We presented the results of our investigation on using the skin modifier in Blender to reconstruct high fidelity and natural-looking polygonal mesh models of neocortical neurons from their morphological descriptions. Our implementation has been integrated into NeuroMorphoVis, an open source neuroscience specific visualization framework, allowing computational neuroscience researchers to directly create mesh models for their simulation-based studies. Compared to those produced with recent methods, the quality of the meshes reconstructed with our approach is better. We compared the visual quality of our meshes to two recent methods to highlight the significance of the presented work. We then demonstrated two principal applications that benefit from those meshes: visualizing full compartmental electrophysiological simulations of digitally reconstructed neocortical circuits and creating compelling artistic renderings for neuroscientific illustrations. The algorithm can be used *as is* to reconstruct meshes for other kinds of data for both acyclic and cyclic directed graphs, for example astrocytes or networks of blood vessels.

Data

Our implementation can be tested by downloading the Blender package from its GitHub repository and selecting the *Skinning* method from the *Meshing Toolbox*. We provide a list of neocortical morphologies in the supplementary material. Other cell types are publicly available in SWC format from the NeuroMorpho.org digital library [ADH07].

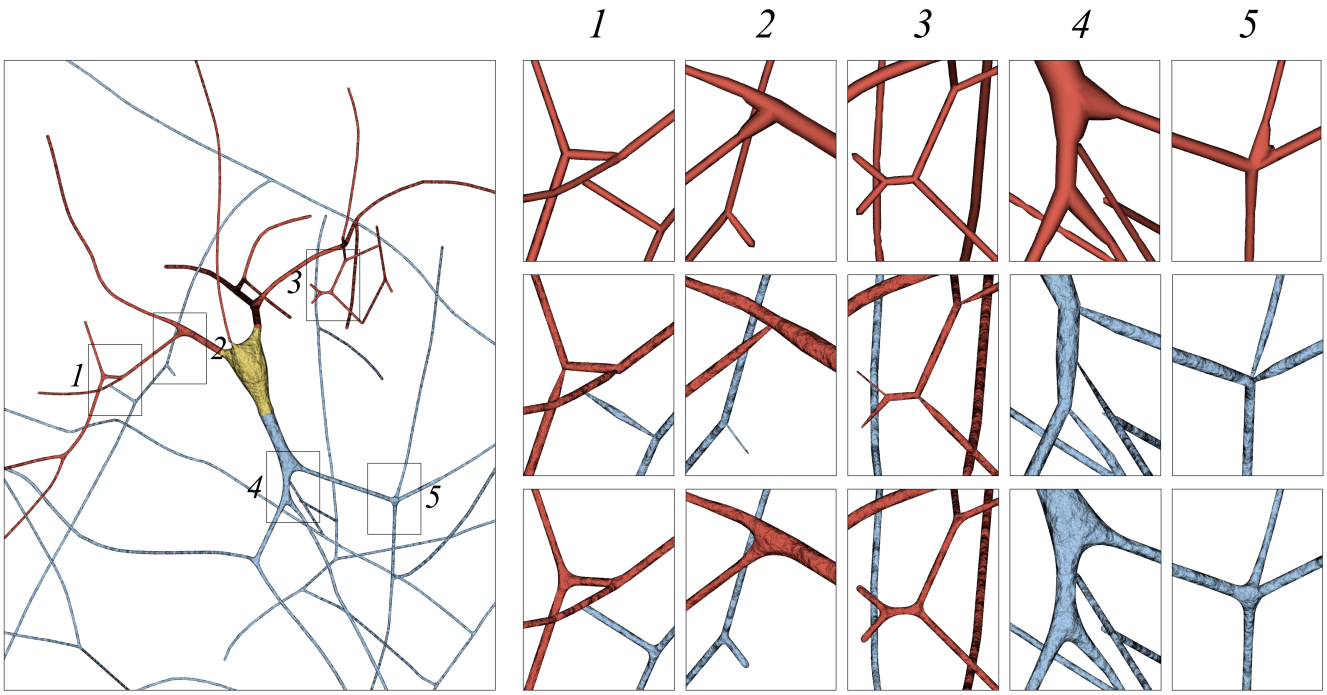

Figure 8: *An annotated surface mesh of a neocortical pyramidal neuron reconstructed with our technique. The closeups compare the results obtained by NeuroTessMesh [GBM*17] (top) and the piecewise-watertight meshing presented by by Abdellah et al. [AHA*17] (middle) and our approach (bottom), where the branching looks natural and realistic for bifurcations and even complex trifurcations.*

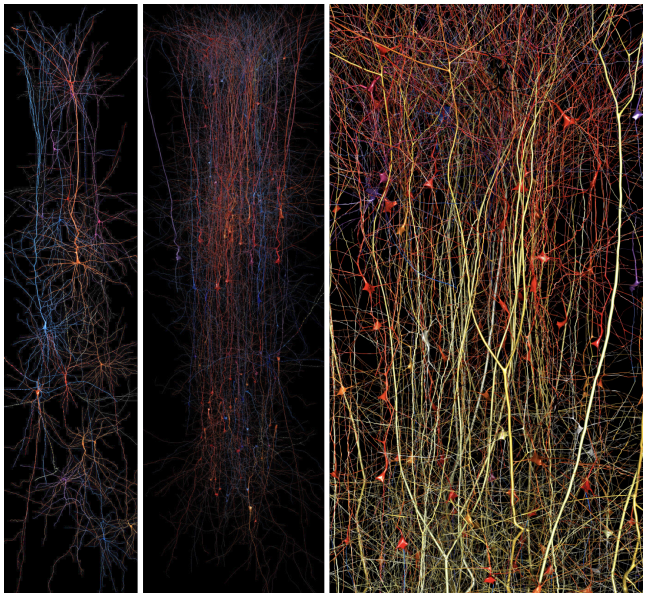

Figure 9: *Visualization of full compartmental simulations of multiple neocortical circuits with different scales using meshes created with our algorithm. The simulation mapping and visualization is done with Brayns [FBN*15].*

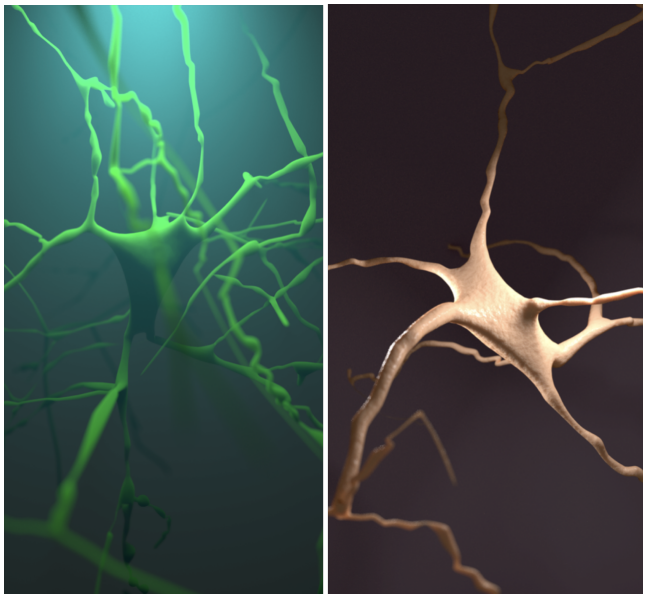

Figure 10: *High quality rendering of neurons created with our algorithm in two different lighting setups. The rendering is done with Cycles.*

References

[Abd17] ABDELLAH, MARWAN. "In Silico Brain Imaging: Physically-plausible Methods for Visualizing Neocortical Microcircuitry". PhD thesis. EPFL, 2017 1.

[ADH07] ASCOLI, GIORGIO A, DONOHUE, DUNCAN E, and HALAVI, MARYAM. "NeuroMorpho. Org: a central resource for neuronal morphologies". *Journal of Neuroscience* 27.35 (2007), 9247–9251 7.

[AHA*17] ABDELLAH, MARWAN, HERNANDO, JUAN, ANTILLE, NICOLAS, et al. "Reconstruction and visualization of large-scale volumetric models of neocortical circuits for physically-plausible in silico optical studies". *BMC bioinformatics* 18.10 (2017), 402 3, 7, 8.

[AHE*18] ABDELLAH, MARWAN, HERNANDO, JUAN, EILEMANN, STEFAN, et al. "NeuroMorphoVis: a collaborative framework for analysis and visualization of neuronal morphology skeletons reconstructed from microscopy stacks". *Bioinformatics* 34.13 (2018), i574–i582 2, 3, 5, 7.

[BMB*13] BRITO, JUAN P, MATA, SUSANA, BAYONA, SOFIA, et al. "Neuronize: a tool for building realistic neuronal cell morphologies". *Frontiers in neuroanatomy* 7 (2013). DOI: 10.3389/fnana.2013.00015 2, 3, 7.

[BTS*14] BERGER, MATTHEW, TAGLIASACCHI, ANDREA, SEVERSKY, LEE, et al. "State of the art in surface reconstruction from point clouds". *EUROGRAPHICS star reports*. Vol. 1. 1. 2014, 161–185 2, 3.

[Con17] CONLAN, CHRIS. "The bmesh Module". *The Blender Python API*. Springer, 2017, 27–42 4.

[EBA*12] EILEMANN, STEFAN, BILGILI, AHMET, ABDELLAH, MARWAN, et al. "Parallel rendering on hybrid multi-gpu clusters". *Eurographics Symposium on Parallel Graphics and Visualization*. EPFL-CONF-216016. The Eurographics Association. 2012, 109–117. DOI: 10.2312/EGPGV/EGPGV12/109-117 1.

[FBN*15] FAVREAU, CYRILLE, BILGILI, AHMET, NACHBAUR, DANIEL, et al. *Brayns: Visualizer for large-scale and interactive ray-tracing of neurons*. 2015. URL: https://github.com/BlueBrain/Brayns 7, 8.

[FFB*10] FERNÁNDEZ, JAIME, FERNÁNDEZ, LAURA, BENAVIDES-PICCIONE, RUTH, et al. "A model for generating synthetic dendrites of cortical neurons". *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer. 2010, 129–138 2.

[Fou16] FOUNDATION, BLENDER. *Blender - 3D modelling and rendering package*. Blender Foundation. Blender Institute, Amsterdam, 2016. URL: %7Bhttp://www.blender.org/%7D 2.

[GBM*17] GARCIA-CANTERO, JUAN J, BRITO, JUAN P, MATA, SUSANA, et al. "NeurotessMesh: A tool for the Generation and Visualization of Neuron Meshes and Adaptive on-the-Fly Refinement". *Frontiers in neuroinformatics* 11 (2017), 38 2, 3, 7, 8.

[GG90] GLASER, JACOB R and GLASER, EDMUND M. "Neuron imaging with Neurolucida—a PC-based system for image combining microscopy". *Computerized Medical Imaging and Graphics* 14.5 (1990), 307–317 2, 3.

[GSS07] GLEESON, PADRAIG, STEUBER, VOLKER, and SILVER, R ANGUS. "neuroConstruct: a tool for modeling networks of neurons in 3D space". *Neuron* 54.2 (2007), 219–235 2, 3.

[Hes07] HESS, ROLAND. *The essential Blender: guide to 3D creation with the open source suite Blender*. No Starch Press, 2007 2.

[HKD93] HALSTEAD, MARK, KASS, MICHAEL, and DEROSE, TONY. "Efficient, fair interpolation using Catmull-Clark surfaces". *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. ACM. 1993, 35–44 4, 5.

[HSP12] HERNANDO, J.B., SCHURMANN, F., and PASTOR, L. "Towards real-time visualization of detailed neural tissue models: View frustum culling for parallel rendering". *Biological Data Visualization (BioVis), 2012 IEEE Symposium on*. Oct. 2012, 25–32 2.

[JLW10] JI, ZHONGPING, LIU, LIGANG, and WANG, YIGANG. "B-Mesh: a modeling system for base meshes of 3D articulated shapes". *Computer Graphics Forum*. Vol. 29. 7. Wiley Online Library. 2010, 2169–2177 4.

[KBH06] KAZHDAN, MICHAEL, BOLITHO, MATTHEW, and HOPPE, HUGUES. "Poisson surface reconstruction". *Proceedings of the fourth Eurographics symposium on Geometry processing*. Vol. 7. 2006 3.

[KRS*19] KANARI, LIDA, RAMASWAMY, SRIKANTH, SHI, YING, et al. "Objective Morphological Classification of Neocortical Pyramidal Cells". *Cerebral Cortex* 29.4 (2019), 1719–1735 1.

[KTvH*09] KOENE, RANDAL A, TIJMS, BETTY, van HEES, PETER, et al. "NETMORPH: a framework for the stochastic generation of large scale neuronal networks with realistic neuron morphologies". *Neuroinformatics* 7.3 (2009), 195–210 2, 3.

[LBB*15] LUENGO-SANCHEZ, SERGIO, BIELZA, CONCHA, BENAVIDES-PICCIONE, RUTH, et al. "A univocal definition of the neuronal soma morphology using Gaussian mixture models". *Frontiers in neuroanatomy* 9 (2015), 137 3.

[LHH*12] LASSERRE, SÉBASTIEN, HERNANDO, JUAN, HILL, SEAN, et al. "A neuron membrane mesh representation for visualization of electrophysiological simulations". *IEEE Transactions on Visualization and Computer Graphics* 18.2 (2012), 214–227. DOI: 10.1109/TVCG.2011.55 2, 4, 7.

[MBQ17] MÖRSCHEL, KONSTANTIN, BREIT, MARKUS, and QUEISSER, GILLIAN. "Generating neuron geometries for detailed three-dimensional simulations using anamorph". *Neuroinformatics* 15.3 (2017), 247–269 3.

[Mei10] MEIJERING, ERIK. "Neuron tracing in perspective". *Cytometry Part A* 77.7 (2010), 693–704 2.

[MHL13] MCDOUGAL, ROBERT A, HINES, MICHAEL L, and LYTTON, WILLIAM W. "Water-tight membranes from neuronal morphology files". *Journal of neuroscience methods* 220.2 (2013), 167–178 3.

[MMR*15] MARKRAM, HENRY, MULLER, EILIF, RAMASWAMY, SRIKANTH, et al. "Reconstruction and simulation of neocortical microcircuitry". *Cell* 163.2 (2015), 456–492. DOI: 10.1016/j.cell.2015.09.029 2, 5–7.

[PA13] PAREKH, RUCHI and ASCOLI, GIORGIO A. "Neuronal morphology goes digital: a research hub for cellular and system neuroscience". *Neuron* 77.6 (2013), 1017–1038 2.

[RCA*15] RAMASWAMY, SRIKANTH, COURCOL, JEAN-DENIS, ABDELLAH, MARWAN, et al. "The neocortical microcircuit collaboration portal: a resource for rat somatosensory cortex". *Frontiers in neural circuits* 9 (2015). DOI: 10.3389/fncir.2015.00044 5.

[Val13] VALENZA, ENRICO. *Blender 2.6 Cycles: Materials and Textures Cookbook*. Packt Publishing Ltd, 2013 7.

[Vil14] VILLAR, OLIVER. *Learning Blender: a hands-on guide to creating 3D animated characters*. Addison-Wesley Professional, 2014 3.

[WK05] WILKINS, MARK R and KAZMIER, CHRIS. *MEL Scripting for Maya Animators*. Elsevier, 2005 2.

[XMF13] XIONG, GUANGLEI, MUSUVATHY, SURAJ, and FANG, TONG. "Automated structured all-quadrilateral and hexahedral meshing of tubular surfaces". *Proceedings of the 21st International Meshing Roundtable*. Springer, 2013, 103–120 2.