frontiers

Robust Quasi-uniform Surface Meshing of Neuronal Morphology using Line Skeleton-based Progressive Convolution Approximation

Xiaoqiang Zhu [1], **Xiaomei Liu** [1], **Sihu Liu** [1], **Yalan Shen** [2], **Yimin Wang**[2,3,*]

[1] *School of Communication and Information Engineering, Shanghai University, Shanghai, China*
[2] *School of Computer Engineering and Science, Shanghai University, Shanghai, China*
[3] *Guangdong Institute of Intelligence Science and Technology, Hengqin, China*

Correspondence*:
Yimin Wang
wangyimin@gdiist.cn

ABSTRACT

Creating high-quality membrane surfaces of neuronal morphology for both visualization and numerical simulation is always a challenging task. In this paper, we developed a novel approach of reconstructing watertight 3D neuronal membrane surfaces from abstract point-and-diameter data. The membrane shapes of the neurons are reconstructed by progressively deforming an initial sphere, and it can be taken as a digital sculpting process. In the dynamic sculpting, the embedded skeleton with radii can serve as a guidance for surface mesh evolution. In order to efficiently deform the surface, a local mapping is adopted to simulate the animation skinning. Therefore, only the vertices within the ROI of the current skeletal position have to be updated. The actual region of influence (ROI) can be determined based on the adopted finite-support convolution kernel, which is convolved with the neural line skeleton to generate a potential field. The progressive convolution potential field guides the mesh evolution to smooth the overall surface regardless of the dendrites or the neural arborizations. On the other hand, the mesh quality during the whole evolution is always guaranteed by the quasi-uniform rules, which splits too long edges, collapses too short ones and moves the vertices within the tangent plane to get regular triangles. Finally, the vertices density on the iso-surface is adaptively distributed according to the neural radii and the surface curvatures.

Keywords: Quasi-uniform mesh, Dynamic sculpting, multiresolution techniques, geometry-based techniques, local mapping query, finite-support convolution kernel

1 INTRODUCTION

With a rapid technical development, increasing attention has been paid to biologically detailed brain visualization and simulation. As the most significant information-processing cells in the brain, neurons are made up of dendrites and axons, which are responsible for receiving and sending signals respectively.

A212 Each neuron is an electrical entity and various methods have been proposed to simulate the electrical behavior of neurons. However, due to the complexity of neuronal structure, most of these methods simulate the conduction behavior of electrical and biochemical signals in a low-dimensional space. Recently, there is a growing requirement to simulate cellular behavior based on 3D high-fidelity models (Mörschel et al., 2017)A109,A213. Numerical simulation of a reaction-diffusion problem requires a fully-defined neuronal geometry, therefore, a key challenge in neuroscience is to robustly generate a high-quality neuronal membrane surface, which can be used to create tetrahedrons for reaction-diffusion simulationA214 in full neurons or networks of neurons.

From a morphological point of view, the anatomy of neurons can be obtained by interactively tracing neuron elements from microscope images, or accelerated from the microscopyA215 image stack using a series of software tools. The neuronal morphology tracking program usually provides a tree-like structure: the unique morphological point in the center of a soma serves as the root node, and the morphology of the neurite is composed of an ordered sequence of interconnected nodes. Besides the 3D coordinates of the skeletal nodes, thickness at each node can also be included to form a point-and-diameter structure. The anatomical features of neurons captured through morphological reconstruction can be used for detailed electrophysiological simulations of voltage dynamics throughout the 3D structure of the neuron (Carnevale and Hines, 2006; Wilson et al., 1988; Gleeson et al., 2007). Unfortunately, the point-and-diameter approximation becomes a problem when one wishes to use the morphologies for visualization or simulationA216. Thus, the visualization of such spatiotemporal data poses a particular challenge because of the complexity of neuron morphologies and the limitations of the morphological point-and-diameter representations.

For the neuronal morphologies of tree-like structures, branch modelling is usually the most complicated partA217 of producing neuronal membrane surfaces, as a self-intersection tends to occur at branches. However, we note that smooth branch ramifications were not taken into consideration in most of previous approaches, and the majority of them concentrate on uniform remeshing, which try to generate a whole mesh with the same edge length (Vorsatz et al., 2003; Botsch and Kobbelt, 2004; Kil et al., 2006; Stanculescu et al., 2011). To be able to represent fine geometric details, these approaches inevitably place too many triangles in regions with low-curvatures. To solve these problems, a novel approach is proposed in this paper for robust neural morphology modeling based on skeleton-driven progressive adaptive remeshing, which achieves both smooth surface approximation and high-quality mesh vertex distributions at the branches (seeA226 Figure 1).

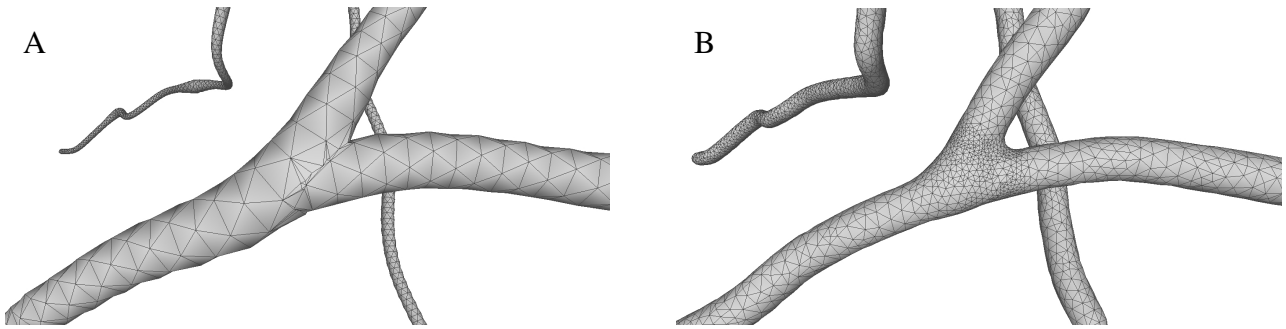


Figure 1. A neuron branch with coarse mesh vertices **(A)** and high-quality adaptive vertex density distributions **(B)**.

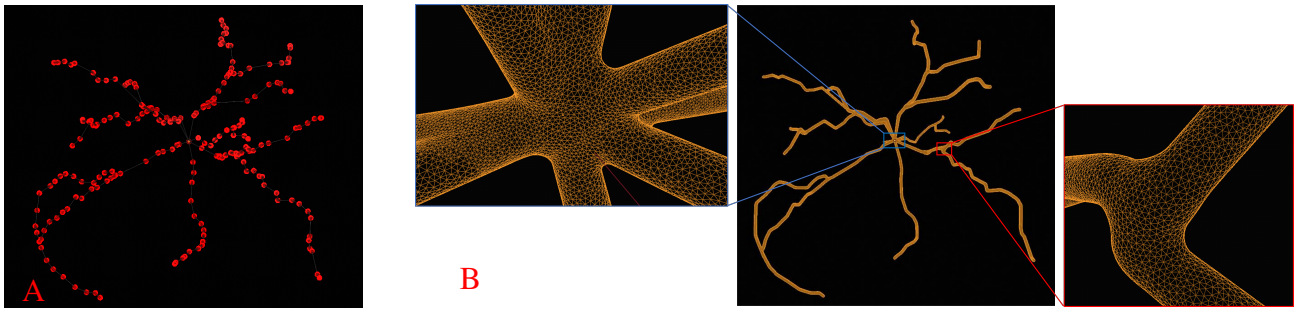


Figure 2. The Morphology of ANDERSON's basal ganglia. **(A)** The structure of ANDERSON's basal ganglia. **(B)** Full mesh.[A234,A235]

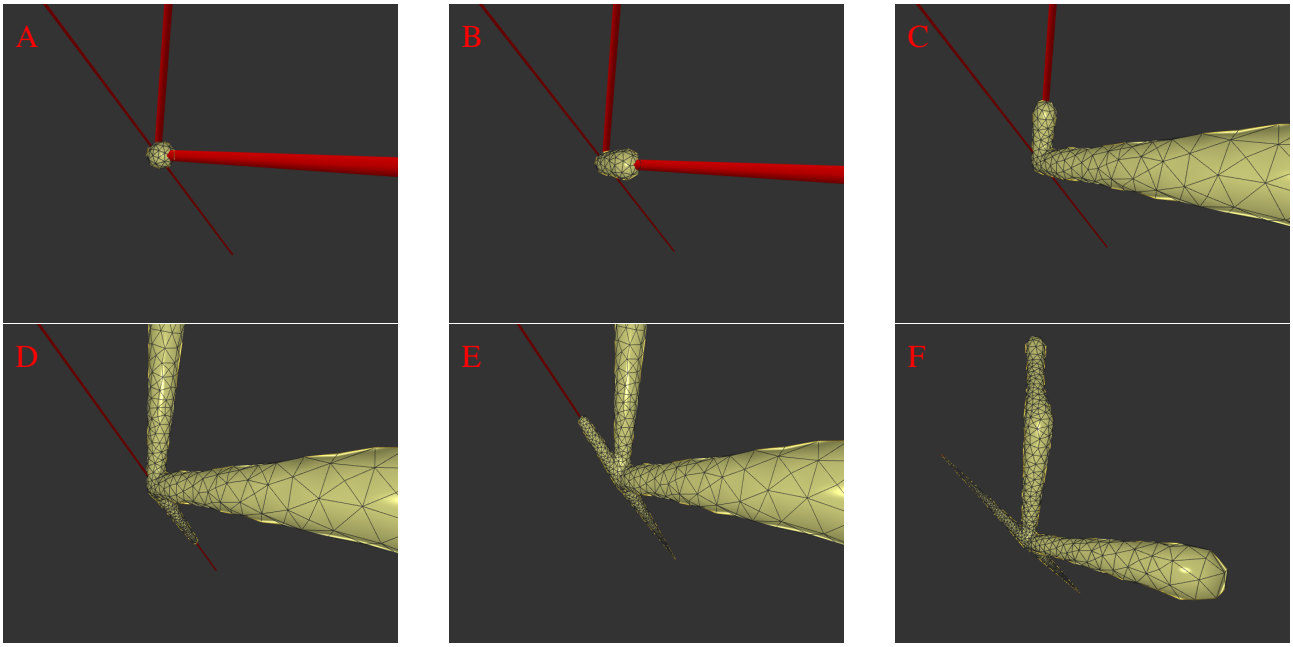


Figure 3. Iterative deformation from an initial spherical mesh at the root node (A) produces the whole neuronal membrane mesh procedurally (B-F) driven by the embedded neuronal skeletons.[A218]

This paper presents a sculpting-like mesh generation for 3D visualization of neuron surface and reaction-diffusion simulation applications. A pair of neuronal skeleton and surface mesh are illustrated in Figure 2. In the presented method, the skeletons are firstly loaded from neuronal morphology files, which are then used to drive iterative deformations of the initial triangular mesh placed at the root node (see Figure 3)[A218]. This method allows subsequent adaptive mesh refinements according to different geometric details and it creates high-quality neuronal membranes robustly. Additionally, the watertight nature of our surface can also be used to create tetrahedral volumes for stochastic reaction-diffusion simulations[A220]. In summary, the main contributions of this paper are as follows:

1. A high-quality neuronal membrane modeling approach is proposed, and the topological robustness can be easily guaranteed through a progressive surface evolution, as the shape of each updated triangle satisfies the adopted quasi-uniform remeshing rules which result in regular triangles.
2. A local convolution surface approximation is presented to modeling branching neuron morphology to achieve a pleasing smoothness and efficiency of the progressive mesh evolution.

3. Procedural[A204] adaptive mesh refinements are utilized to capture varying scales of geometric details, which balances the mesh quality and the computational expense.
4. A skeletal vertex mapping scheme is introduced to accelerate the neighboring queries, and it also relates mesh vertices to the original morphological skeleton, which is a requirement for other visualization and simulation applications.

2 METHODS

2.1 Architecture

The neuronal membrane mesh is gradually generated through iterative ROI (Region of Influence) queries of an initial model, weighted deformations and local topology reconstructions, which can be represented with vertices and triangles[A1061]. The whole workflow consists of the following steps (see[A221] flow chart in Figure 4).

1. Preconditioning the morphology tree constructed from input files.
2. The meshing algorithm starts along the respective root neurite paths.
3. Querying the ROI in skeletal mappings.
4. Stretching and deforming the mesh in the affected area to the approximated positions.
5. Optimizing the mesh vertex distribution using remeshing with adaptive resolutions according to various geometric details.
6. Iteratively performing the same steps for all neurite paths.

2.2 Morphology Preconditioning

Our approach takes skeleton-based representation of neuronal trees (e.g. the SWC files) as input. A neuronal tree typically origins from the soma[A223] node, and consists of a number of neurite segments that further bifurcate at branching points. While such segments should have reasonable length in order to faithfully and effectively portray the geometry of the neuron, it is not unusual to find neurons with overly short segments in common data repositories such as NeuroMorpho.org[A225]. Such artifacts in a neuronal tree could not only lead to redundant representation and thus slow down the overall mesh generation speed, but cause malfunction of the approach as well if the length of a segment is even smaller than the marching step. Therefore, as a pre-processing step, we eliminate all such trivial segments whose length $l_{seg} < \epsilon \cdot R_{neuro}$ in the neuronal tree.[A110,A205,A222,A224]. (see[A226] Figure 4).

2.3 ROI Query Strategy Based on Skeletal Mapping

In order to speed up the generation of neuronal membrane mesh, a skeletal mapping-based ROI strategy[A227] is proposed, which allows local deformation region queries. A list container is attached to each skeleton node for storing those vertex indices which are the closed vertices to the current node. The mappings are created procedually, and the mesh vertex mappings are created based on the exact projection position of the current neuronal segment that deforms the vertex in question.[A1062] We first query skeleton nodes in the ROI, and then query their mapping vertices on the membrane surface mesh. Compared with the ROI query algorithm based on space partition proposed in (Zhu et al., 2013), our method further accelerates the generation of skeleton-driven mesh modeling. Figure 5 shows the ROI query of skeletal skins.

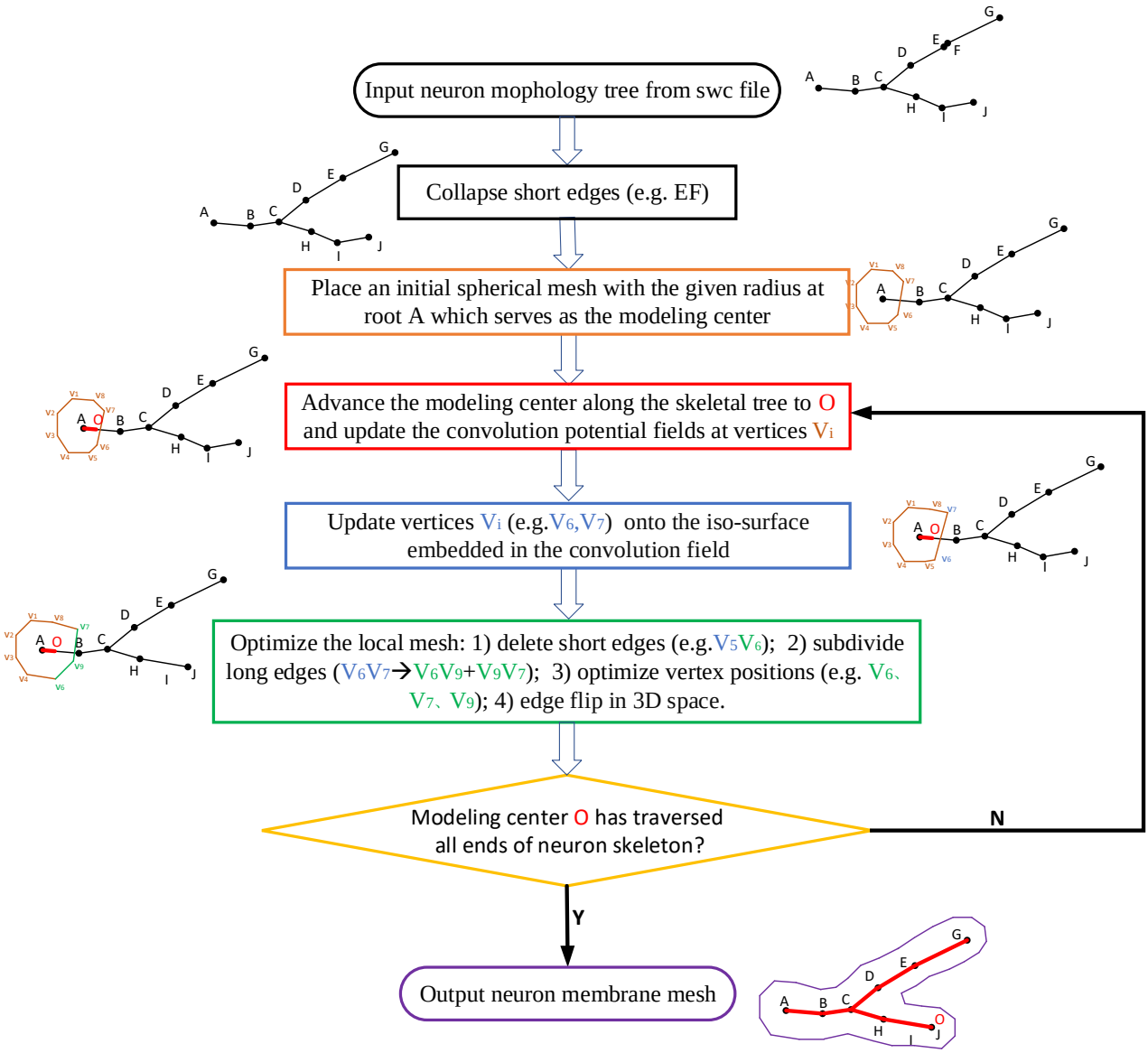


Figure 4. The progressive neuronal morphology modeling workflow. The procedure starts from an initial soma mesh at the root of the neuron tree, then searches the vertices affected by the skeleton-based implicit surfaces, and finally project these vertices onto the implicitly defined convolution surfaces for further remeshing if relevant edges have been longer than the predefined threshold.[A110,A222,A302]

2.4 Local Convolution Surface Approximation

Skeleton-based implicit surfaces are usually introduced to create branching structures (Bloomenthal and Shoemake, 1991; Hart and Baker, 1996; Jin and Tai, 2002) due to their smoothness and topological variations, however, the Marching cubes polygonizations (Lorensen and Cline, 1987) of the iso-surface they employed suffer from high computation complexity, limited resolution, and low-quality triangular meshes. Furthermore, it is prone to missing small twigs for complex branch models because the output of the Marching cubes is resolution-dependent. Even though there are a large number of improvements (Akkouche and Galin, 2001; Bloomenthal, 1994; Bottino et al., 1996; Van Overveld and Wyvill, 2004;

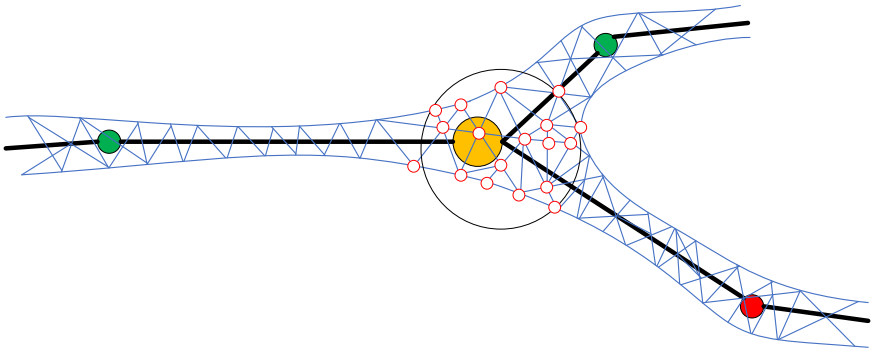


Figure 5. Skeletal mapping. Each mesh vertex (e.g. hollow red circles) is mapped onto some skeleton node (e.g. orange circle), which indicates that these mesh vertices will be influenced by the skeletons connected to the skeleton node and will be deformed accordingly.[A1063]

Wyvill et al., 1986; Zhu et al., 2013), it is still difficult to balance the quality of the iso-surface polygons and the performance. Recently, a point skeleton-based metaball policy (Abdellah et al., 2020) is introduced to create accurate mesh models of brain vasculatures[A228], and metaballs are also used to skin the different structural components of astrocytes Abdellah et al. (2021)[A229] and then blend them in a seamless fashion. However, at straight line skeletons, too many metaballs are essential to placed closely to approximate the cylindrical neuronal or vascular morphologies.

2.4.1 Skeleton-based Convolution Surfaces

In this paper, our approach begins with an initial triangular mesh (soma[A230]), and it will be progressively deformed to approximate the target shape, which is defined as a convolution surface based on embedded neuronal line skeletons. Therefore, the final neuronal membrane surface mesh achieves high-order smoothness at arbitrary branches.

Convolution surfaces can be regarded as an isosurface embedded in a three-dimensional scalar field, which is calculated by convolving the geometric skeleton V_i with a low-pass filter function f[A1064]. Given a skeleton segment:

$$g_i(p) = \begin{cases} 1, & p \in \text{ skeleton } V_i \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

and a kernel function:$f : R^3 \rightarrow R$, the contribution of the potential value of the current skeleton in question at point p is the convolution of the function f and g_i:[A1064]

$$F_i(p) = \int_{V_i} g_i(q) f(p-q) dV = (f \otimes g_i)(p)^{A1064} \tag{2}$$

2.4.2 Finite Support Quartic Kernel

Among them, the kernel function can be divided into infinite support (such as: Cauchy kernel function (McCormack and Sherstyuk, 1998)) and finite support (such as: quartic polynomial kernel function (Jin et al., 2009)). For a finitely supported kernel function, if the distance of a certain point from the skeleton is

greater than the support radius of the kernel function, the potential contribution of the skeleton drops to zero; Instead, for an infinite support kernel function, no matter how far the point in the three-dimensional space is from the skeleton, the potential contribution of the skeleton to the current point is always greater than zero even though it is very tiny. As neuronal morphology usually appears to be a tree-like graph shape, we take line segments as the convolution skeleton, and the fourth-degree polynomial kernel function as the kernel function whose formula can be defined as:

$$f_{\text{Quartic}}(r) = \begin{cases} \left(1 - \frac{r^2}{R^2}\right)^2, & r \le R \\ 0, & r > R \end{cases} \tag{3}$$

where R is the effective radius of the kernel function.

2.4.3 Local Convolution Approximation

The convolution surface S based on a series of skeleton segments can be defined as:

$$S = \left\{ p \mid \sum_{i=1}^{n} \lambda_i F_i(p) - T = 0 \right\} \tag{4}$$

where F_i is the field contribution of the i^{th} skeletal segment as defined in Eq. 2[A1064], λ_i is its weight factor, and T is the threshold for extracting the iso-surface from the embedded potential scalar field.

For a point p on the membrane mesh[A231], the adopted local approximation scheme assumes that the closest skeleton to p is a line segment with infinite length. The assumption could not only reduce unessential convolution calculation beyond support radius, but also create natural blending between adjacent line segments due to their smooth thickness variations. Therefore, it is quite suitable for large amount of neuronal morphology approximation with complex graph structures. Actually, the potential contribution of an infinite skeleton to an arbitrary 3D position p can be calculated as:

$$F_{\text{Quartic}}(p) = 2\lambda_i \int_0^{\sqrt{R_i^2 - d_i^2}} \left(1 - \frac{d_i^2 + x^2}{R_i^2}\right) dx = T \tag{5}$$

$$\Rightarrow \lambda_i = \frac{15TR_i^4}{16\left(R_i^2 - d_i^2\right)^{\frac{5}{2}}} \tag{6}$$

where d_i is the average distance between the i^{th} skeleton and p, and T is still the global iso-value for the entire convolution surfaces.[A1064] R_i is the effective radius of the kernel function corresponding to the current skeleton, which can be set to be an empirical value of $2d_i$ in our experiments. And λ_i is the weight of the current skeleton segment, which can be derived analytically.

The solution to the above kernel function is based on the assumption that the value of the kernel function outside a certain distance is infinitesimal. Therefore, the fourth-order polynomial kernel function is usually preferred for its local support characteristics, which could reduce huge convolution computation.

2.4.4 Convolution Field-guided Mesh Projection

The implementation of approximation is to project each vertex of the remeshed surface onto the implicit convolution surface in their respective gradient directions by performing standard Newton iterations

(Vaillant et al., 2013):

$$p = p + sign(F(p) - T) \cdot l_{step} \cdot \frac{\Delta F(p)}{\mid \Delta F(p) \mid} \tag{7}$$

where, the initial evolution step length l_{step} of the vertex p can be set as $step = \frac{1}{2} l_e$ and l_e is the the length of the shortest edge coincident to p.[A1064] After each subdivision, the projection step length of a new vertex can be derived from its adjacent vertices (Figure 6).

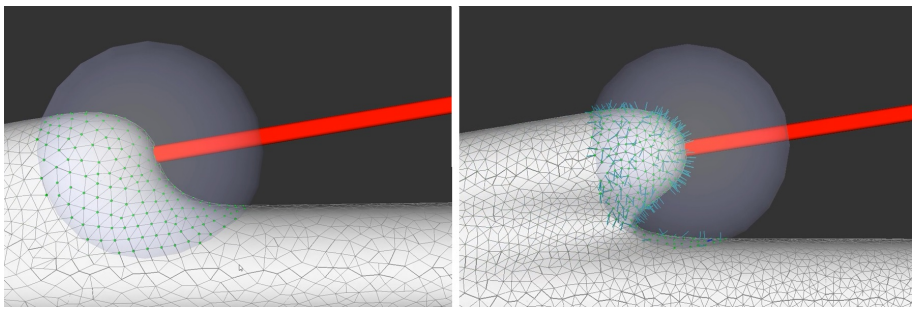


Figure 6. Convolution surface approximation at branches using progressive evolution. In a branching region a new sub-branch grows from the main trunk (left), and the implicit approximation and projection produce smooth branches through not only the current sub-branch deformation but also the blending neighboring sub-branches.

2.5 Remeshing Algorithm based on Quasi-uniform Mesh Structure

Isotropic remeshing is a valid scheme to optimize the distribution of mesh vertices and topological connectivity, and the surveys in (Botsch et al., 2010; Alliez et al., 2008) have proposed various methods for surface remeshing. Numerous techniques like the particle-based methods (Ahmed et al., 2016), the Delaunay refinement methods (Cheng et al., 2013), and the randomized sampling methods (Ebeida et al., 2016) pay attention to generating isotropic meshes. The methods based on Centroidal Voronoi (Du et al., 1999) tessellation attempt to make each point in accord with the centroid of its Voronoi region. There are various ways to calculate a Voronoi diagram on surfaces, such as mesh parameterizations, discrete clustering, restricted Voronoi diagram (Yan and Wonka, 2015), and geodesic Voronoi diagram (Liu et al., 2016). The fundamental theory of the field-based methods is that isotropic triangular meshes can be taken out from sixway rotational symmetry directional fields (Du et al., 2018). In addition, there are edge-based approaches are based on local operators, including edge split, edge collapse, edge flip, and vertex repositioning (Botsch and Kobbelt, 2004; Dunyach et al., 2013; Wang et al., 2018).

Therefore, in our implicit mesh approximation, two significant aspects will be taken into consideration: 1) A triangular mesh representation of the implicit surfaces should approximate the underlying geometry as accurate as possible; and[A232] 2) The mesh vertex distribution and connectivity should satisfy high-quality triangles with similar edge lengths coincident to the same polygon, allowing smooth membrane surface visualization and more stable voxelization for simulation[A111].

A quasi-uniform mesh (Stanculescu et al., 2011) supports successive surface deformation and subdivision at any level of detail, ensuring high-quality triangular mesh during the entire progressive deformation process. Details about how to build and remesh a quasi-uniform mesh will be introduced in the following description. Figure 7 illustrates the dynamic progressive mesh evolution with (right) and without (left)

the adopted optimization operations. In the mesh generation, the remeshing based on quasi-uniform configuration can optimize the membrane mesh effectively, thereby achieving robust progressive modeling.

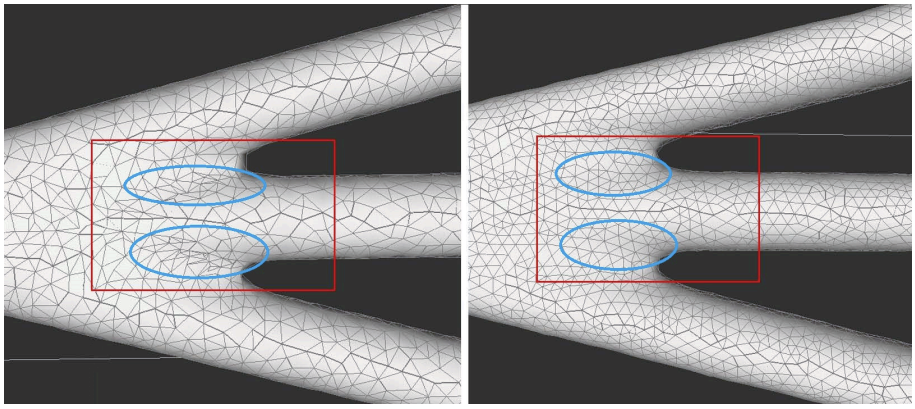


Figure 7. An optimized remeshing based on quasi-uniform mesh. The mesh without vertex optimization (left) includes many low-quality triangles especially in the branching region which have been highlighted in red and blue, while these triangles can be improved using an iterative remeshing optimization (right).A236

2.5.1 Quasi-uniform Mesh Configuration

A uniform mesh is composed of polygon elements of roughly the same size, while a non-uniform mesh has elements of different sizes. In many aspects, uniform meshes enjoy such advantages as stable structures and efficient high-level geometric algorithms over non-uniform ones. However, creating uniform meshes is non-trivial especially for surfaces with complex topology. Therefore, a relaxation on constraints on uniform meshes is employed here. As defined in (Stanculescu et al., 2011), a mesh M is said to be quasi-uniform if there exists l_T and d, such that M results from iterative remeshing which assures a maximum edge length l_T and a minimum edge d. That is to say, for each edge $e \in edges_M$, it should falls in the definite interval $length_e \in (d, l_T)$.

Given a closed manifold mesh M and a threshold l_T , we can ensure that all edges of M are smaller than l_T by iteratively splitting those edges larger than l_T, known as l_T tight mesh. The split operation amounts to adding a vertex at the midpoint of an edge, and correspondingly splitting the relevant triangle face into two adjacent triangles. The iteration over the edges is performed using a simple queue, and the resulting new edges are inserted into the queue.

The l_T tight property is not the only factor to ensure high quality triangles, since edge collapses should be performed to delete too short edges to generate higher-quality triangle meshes. Edge collapse moves the two vertices of a short edge to the midpoint of the collapsed edge, ensuring that all the edges of M will not be shorter than the collapse threshold d (that is, the minimum edge length). Moreover, it is necessary to check the validity of the collapse operation before each implementation. If a collapse operation produces intersected triangles, the operation is illegal and will not be performed for the current edge.

Obviously, it is difficult guarantee that the lengths of all the edges generated after the collapse operation are greater than the minimum length d, and the collapse operation may also destroy the l_T tightness property on the mesh. Therefore, before establishing the l_T tight mesh M, it is necessary to perform the collapse operation to ensure that the minimum length of the mesh edges is greater than d, and then the l_T tightness property should be restored. In fact, d can be taken as an internal parameter of a quasi-uniform

mesh, since it is bound to $l_T : \mathrm{d} <= l_T/2$, ensuring that the edges greater than l_T will not be divided into edges shorter than d while establishing the l_T tight mesh after a collapse operation.

2.5.2 Local Quasi-uniform Mesh Remeshing in ROI

Based on the quasi-uniform mesh configuration, too long or too short edges should be splitted or collapsed respectively, which will be followed by mesh optimizations including vertex valence equalization, triangle area equalization and implicit surface projection (see Algorithm 1).[A112]

Algorithm 1 Remeshing

Input: $init\_surface, target\_edge\_length$
Output: quasi-uniform mesh

function REMESH($init\_surface, target\_edge\_length$)
  $result \leftarrow init\_surface$
  $low \leftarrow \alpha * target\_edge\_length$
  $high \leftarrow \beta * target\_edge\_length$
  for $vHandle : vHandles$ **do**
    $spilt\_long\_edges(high)$
    $collapse\_short\_edges(low, high)$
    $equalize\_valences()$
    $tangential\_relaxation()$
    $project\_to\_surface()$
  end for
  return $result$
end function

Vertex Valence Equalization.[A112] In order to optimize each triangle of a mesh as regularly as possible, an edge flip operation can be applied to achieve the vertex valence of 6. Similar to edge collapses, a legality check is also needed before each flip implementation.

Triangle Area Equalization.[A112] After iteratively performing the above optimization operations on the mesh M, we can get a quasi-uniform triangular mesh. The edge lengths of the mesh are basically close to the target length l, which satisfies $\mathrm{d} < l < l_T$, and the valences of the vertices are close to 6. However, the triangle ares and the edge lengths may still vary with each other, which can be further optimized. Therefore, we carry out a vertex fine-tuning through an area-based tangential smoothing. Each vertex p_j is assigned a weight that equals to its averaged area $A(p_j)$, and a tangential smoothing moves p_j to its weighted barycentric position

$$g_i = \frac{1}{\sum_{p_j \in N(p_i)} A\left(p_j\right)} \sum_{p_j \in N(p_i)} A\left(p_j\right) p_j. \tag{8}$$

Iterative Implicit Surface Projection and Smoothing.[A112] Following each tangential smoothing, a projection onto the implicit surfaces can be performed for each smoothed vertex (Figure 8), otherwise jitter artifacts maybe occurs. The new position can be updated according to the projection formula

$$p_i \leftarrow p_i + \lambda \left(I - n_i n_i^T\right)\left(g_i - p_i\right), \tag{9}$$

where n_i is the normal vector and λ is the damping factor for oscillation avoidance. Vertices with relatively large Voronoi regions have more weight, attracting mesh vertices and thus reducing their own area. Experiments show that the mesh area of vertex-related region can be averaged without too many iterations.

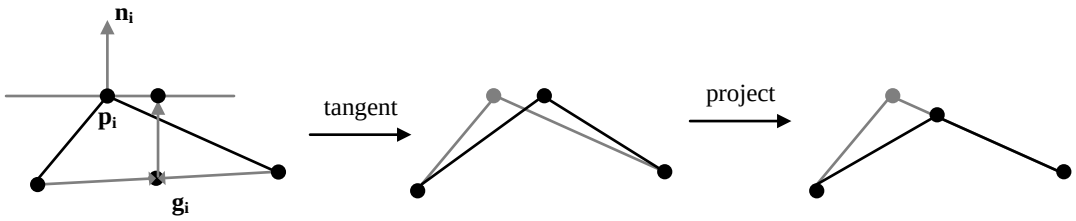


Figure 8. Tangential relaxation and vertex projection.

2.6 Strategies for Adaptive Mesh Vertex Density

To ensure robust generation of high-quality mesh models for neuronal morphologies, a mesh optimization for high-resolution levels of details is required. However, a high-resolution representation usually results in too many tiny triangles on the surface parts with low curvatures, which not only imposes high burden on memory but also wastes unnecessary computational time. On the other hand, in order to generate topologically correct surfaces at branches robustly, a relative high-resolution surface mesh is preferred. Therefore, the marching step size of the mesh evolution should be related to both the skeletal structures and mesh vertex densities so as to create neuronal morphology with adaptive edge lengths.

The robustness of the surface mesh generation at branches and the corresponding mesh vertex density depend on the angles between the branches, which are used to control the marching step size along the skeletons. Actually the mesh vertex density can be determined according to the target edge length in the current region, which are indirectly related to the branching angles. It should be noted that the mesh vertices in the ROI ought to be queried as fast as possible for efficient mesh evolution, so the step sizes of both the marching along skeletons and the convolution surface approximation should not exceed the lower length limit of the mesh edges. Details for determining vertex densities at a branch are as follows:

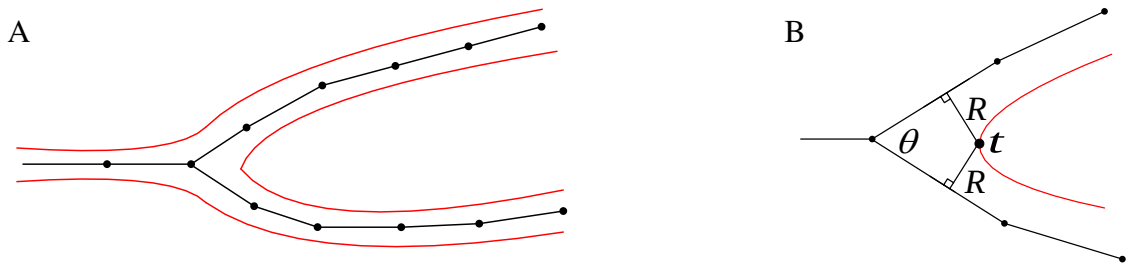


Figure 9. Adaptive mesh resolutions at branches. **(A)** presents a branch, and **(B)** shows the separating position t.[A108]

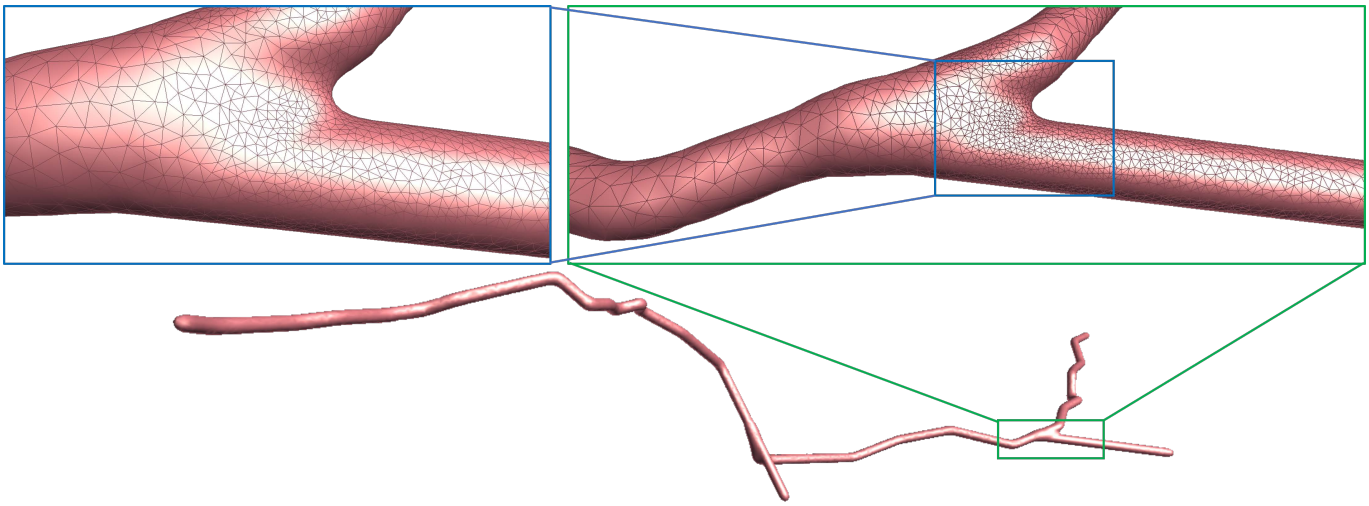


Figure 10. Adaptive mesh vertex distributions in a separating region. High-density vertices are placed at the branch separating region to capture details. A238

1. **Separating Point.** According to the angle of a neuron branch and the support radius R of the target convolution surfaces, the separating position t at the branch can be pre-calculated. Since the distances between t and the skeleton segments of the both branches are R, the projection points of t on the skeleton segments can be deduced based on the branch angle θ, which are the corresponding positions with the maximum vertex densities in dynamically adaptive mesh evolution (Figure 9).
2. **Maximum Vertex Density.** As the neuronal morphology at the branching t usually creates a surface with the locally highest curvature, the maximum vertex densities are assigned for the mesh in the nearby region. Therefore, triangles with the minimum edge length are generated in the separating region based on a marching along the skeleton with the minimum step size.
3. **Minimum Vertex Density.** On the other hand, the maximum target edge length can be usually set as the average edge length of the initial mesh, as our mesh evolution begins with an initial soma with the largest radius. The local maximum edge lengths at other positions that are far away from branching can be calculated according to the ratio of the current skeletal radii and the soma radius.
4. **Edge Length Interpolation.** After the locally minimum target edge lengths and the maximum ones are determined, edge lengths for intermediate regions between each pair of the minimum $l_{\min}$ at t and maximum length $l_{\max}$ can be linearly interpolated accordingly. A108

Moreover, the adaptive mesh evolution could not only optimize the vertex distribution but also accelerate the surface generation and subsequent rendering, as illustrated in Figure 10.

3 RESULTS AND DISCUSSION

3.1 Results

To validate our proposed mesh evolution-based convolution approximation, our method has been applied to many cases from one of the largest cell morphology databases NeuroMorpho.Org, as shown in Figure 11. It can be seen A233 that the underlying convolution potential field could smooth the whole membrane surface regardless of the dendrites or the neural arborizations. Due to the advantageous superposition

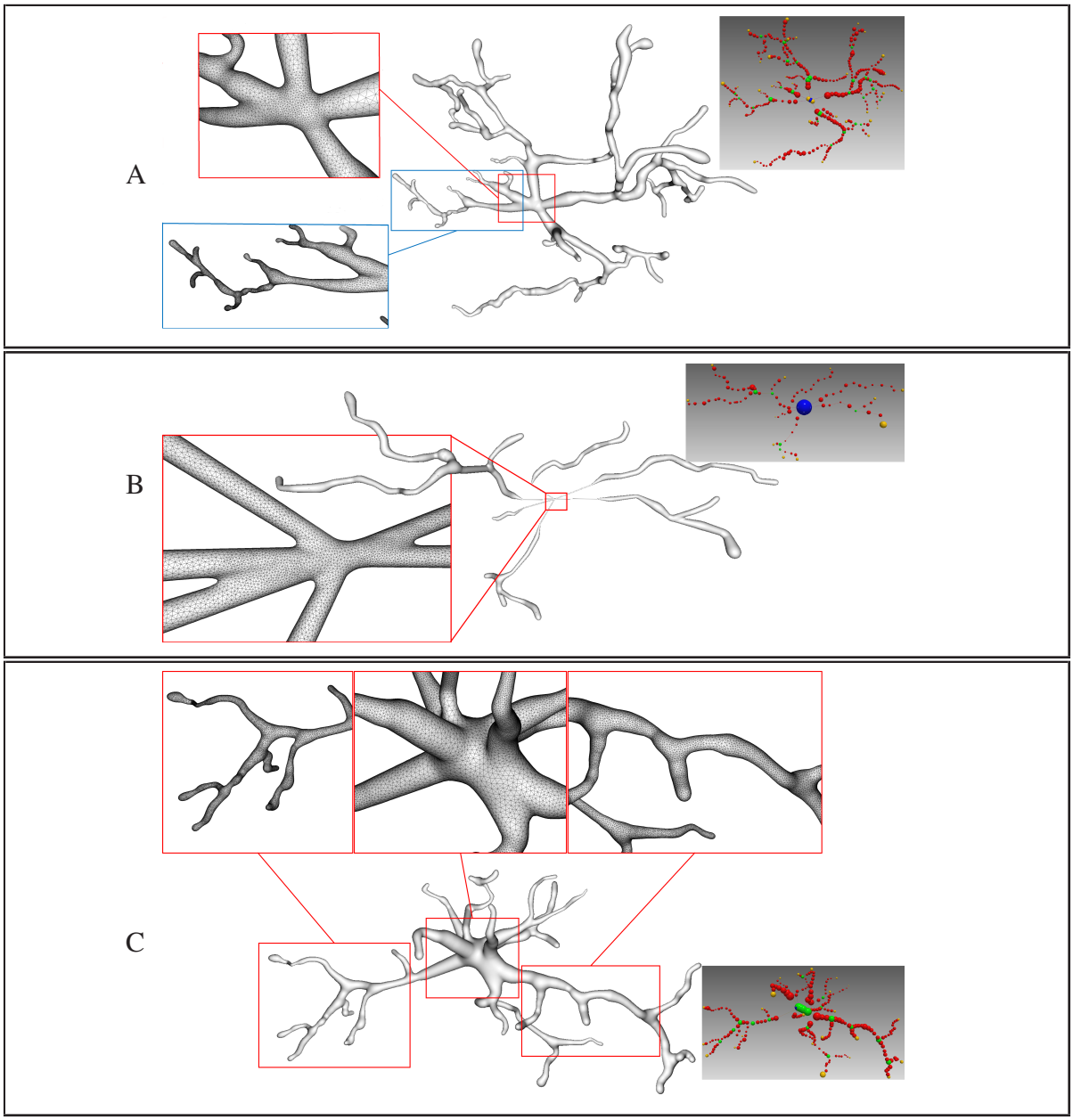


Figure 11. Some cells of Abdolhoseini_Kluge.[A303]

property of convolution surfaces, no advanced blending operations are introduced to produce crease-free rounded neuronal surfaces. Moreover, in order to improve the efficiency of the modeling system, we propose the ROI query strategy based on skeleton-vertices mapping, which can effectively accelerate the mesh evolution.

On the other hand, the mesh vertex density distribution is optimized to represent adaptive details . The density of the surface vertices is related to both the thickness (Figure 12) of the neuronal morphology and the surface curvature (Figure 13). Although the thickness can be easily obtained from the morphology files, it is nontrivial to calculate the curvature in a dynamic mesh evolution. The main reason lies in that the curvature has to be pre-determined before the quasi-uniform remeshing which is used to the edge length limitations. To this end, our insight is that high curvatures are distributed at branching regions especially

288 at the separating regions of skeleton-based convolution surfaces. Therefore, a proportionate ratio to the
289 distance between an edge and the separating points is employed to the current edge length.[A108]

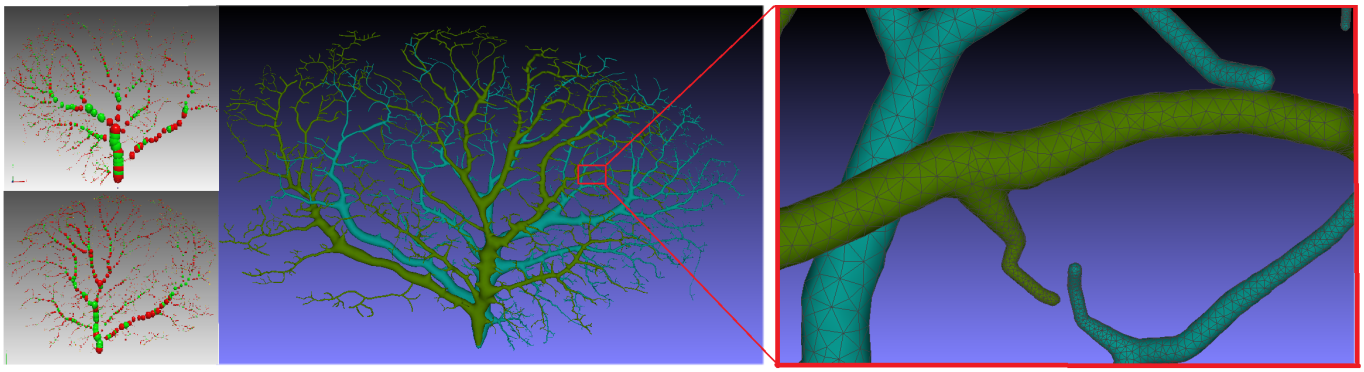


Figure 12. Membrane surfaces with adaptive edges for different thickness: surfaces with larger radius are created with lower-density vertices and vice verse.

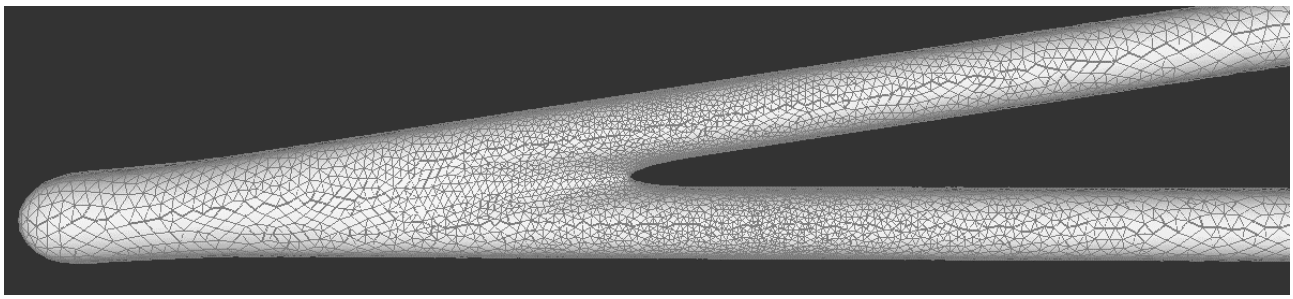


Figure 13. Adaptive vertex distribution at arborizations. Special care is taken at the branch separating position to create smooth high-curvature surfaces.

290 As the final mesh is progressively deformed from an initial sphere (Figure 14), no deliberate stitching
291 operations have to be involved[A114]. In the dynamic mesh evolution, a quasi-uniform mesh structure is
292 used to guide the edge split and collapse, and it does not allows too long or too short edges which are
293 detrimental to a robust mesh generation.

294 3.2 Comparison with Similar Tools

295 Neuronal membrane surface meshing is a fundamental preparation for neuronal electrical and chemical
296 simulations. Earlier software packages, such as Neurolucida (Glaser and Glaser, 1990), NeuroConstruct
297 (Gleeson et al., 2007), NeuGen (Eberhard et al., 2006), and Genesis (Wilson et al., 1988) provide
298 approximations of neuron surfaces with mesh-based methods, but they tend to create meshes with low
299 qualities, in which self-intersected parts usually occur in branching regions. Other methods such as the
300 one presented in (Lasserre et al., 2011) starts from a sphere (made with quads) with a fixed resolution,
301 and the dendrites are then generated by quad-extrusions starting from the soma. At the end of the method,
302 a Catmull-Clark subdivision is employed to smooth the whole mesh, generating realistic, smooth, and
303 closed meshes. However, the preservation of desirable properties of being 2D-manifold are not stated as
304 an objective in the work.

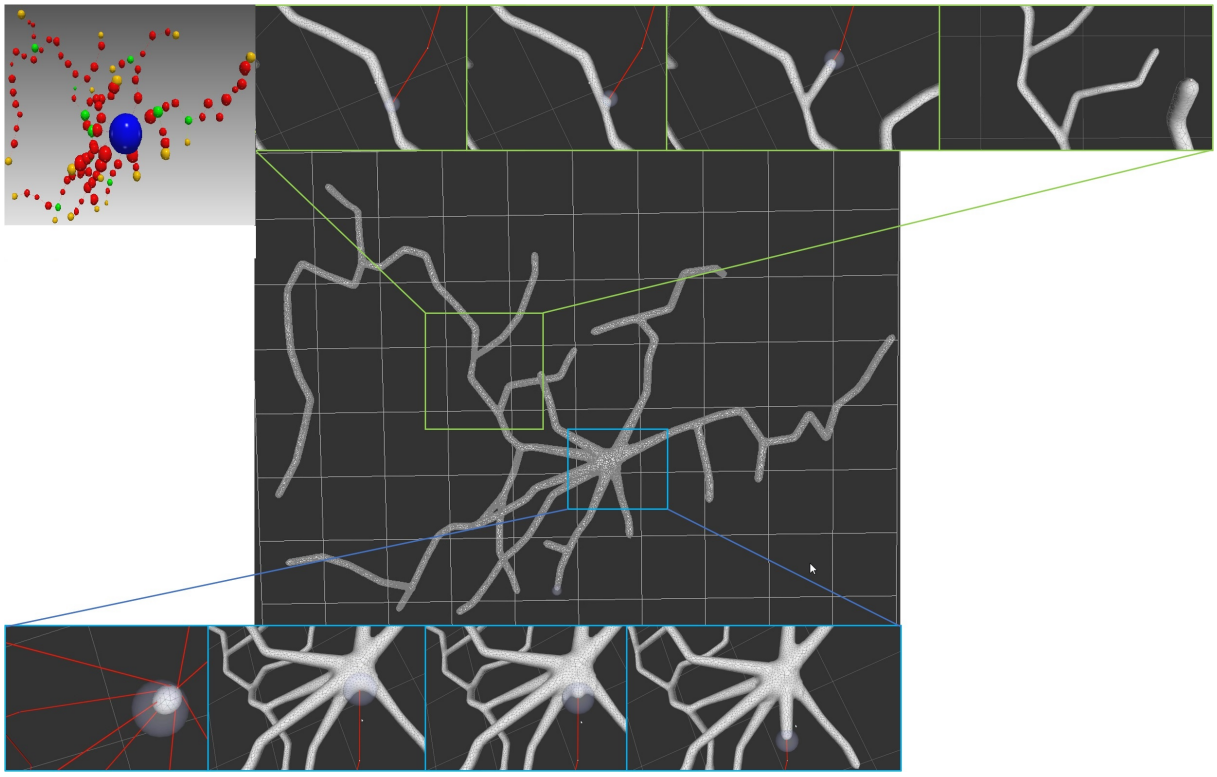


Figure 14. Progressive surface evolution. Mesh growing processes at a branch (up) and at the root node (bottom) are illustrated in details.

CTNG (McDougal et al., 2013) uses constructive solid geometry to define a plausible reconstruction without gaps, which represents a geometry as the unions and intersections of three-dimensional graphics primitives. It then uses ”constructive cubes” to produce a watertight triangular mesh of the neuron surface. However, it depends on the underlying Marching Cubes which create triangles with greatly varying aspect ratios.

Neuroize (Brito Menéndez et al., 2013) defines a physics-based mass-spring system to generate a neuron mesh, which could create a high-quality mesh. However, due to the versatility of the mass-spring system, complicated fine-tuning of several simulation parameters is required to achieve an accurate reconstruction.

NeuroTessMesh (Garcia-Cantero et al., 2017) improves the technique in Neuronize by applying an FEM (Finite Element Method) (Erleben et al., 2005) to simulate the deformation, which enables a convenient control over the mesh deformation. The approach approximates the cell bodies and the dendritic and axonal arbors in independent procedures that are later merged, resulting in a closed surface that approximates a whole neuron. However, NeuroTessMesh does not deal with mesh generation at branch points, and in particular, the junction between neurite and soma^{A230} is not smooth, which is obviously not desirable as shown in Figure 15(A). Although NeuroTessMesh can adapt mesh resolution based on the distance to the camera, it cannot adapt mesh resolution based on different geometric details when generating individual neuron mesh models.

AnaMorph (Mörschel et al., 2017) uses a recursive tessellation algorithm starting from an icosahedron to construct the soma mesh as an initial mesh. The AnaMorph framework can produce relatively high-quality meshes, but it is not robust for building some complex neuronal structures, especially for high resolution

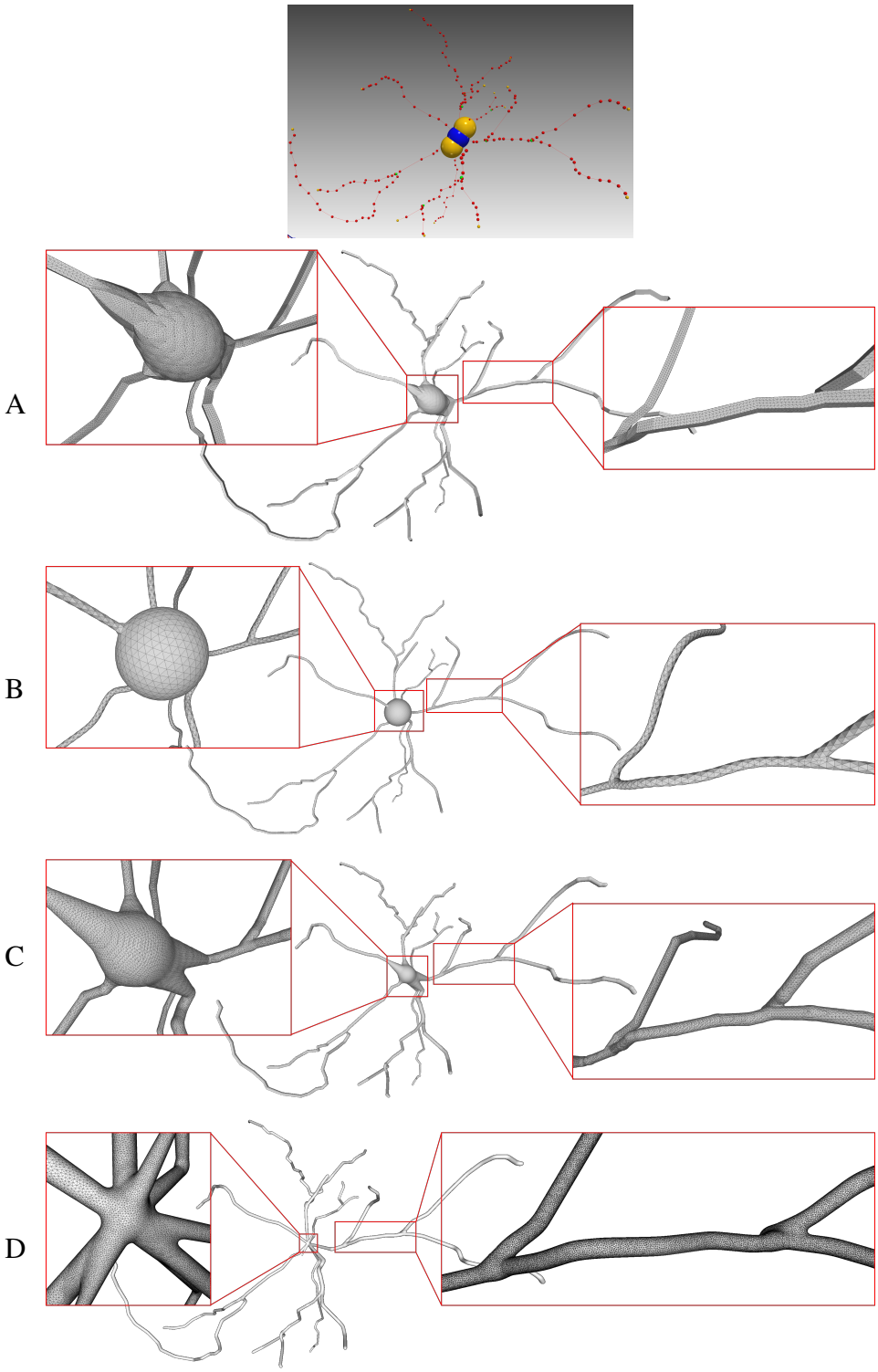


Figure 15. Comparison of the same neuron using different 3D representations. Models generated using NeuroTessMesh (A), AnaMorph (B), Neuromorphovis (C) and our proposed method (D).[A239]

modeling at branches. Moreover, a complex stitching operation has to be performed on adjacent branches in order to produce a watertight surface (Figure 15(B)).

Table 1. Performance evaluation[A104,A202,A203,204,A206,A207,A304]

Models	Fig.2	Fig.3	Fig.10	Fig.11(A)	Fig.11(B)	Fig.11(C)	Fig.12	Fig.14
Skeleton Nodes	217	10	35	345	93	247	3324+3480	108
Mesh Vertices	293556	3880	4652	99383	112499	69749	346819+334591	174445
Time (s)	2608	21	166	2720	2291	2092	6647+14794	821
Memory (Mb)	70.6	42.1	44.4	63.9	64.7	53.5	100+181.4	56.7

Neuromorphovis (Abdellah et al., 2018) extends an earlier meshing algorithm (Abdellah et al., 2017) capable of reconstructing piecewise watertight meshes that could be employed to visualize detailed electrophysiological activities obtained from voltage dynamics simulations. However, too many metaobjects have to be placed for creating a smooth-varying implicit field for extracting a neuronal membrane iso-surface. In addition, a post-optimization is required to improve the extracted triangles with greatly varying aspect ratios (Figure 15(C)).

3.3 Evaluations

As our approach employs a progressive mesh generation, much time is required to complete the growing process. However, acceptable performance can be obtained for regular neuronal membrane mesh creations, and the timings for the examples in our experiments are listed in Table 1. All the data are tested on a PC with a 2.9 GHz Intel Core i7-10700 CPU (only 1 core is used) with 16 GB memory. Although our approach is computation-intensive due to its procedural deformation, very few memories are enough for successful executions. Therefore, our system can be performed on regular PCs. [A104,204]

For 3D simulations, our produced membrane meshes can be directly input into TetGen (Si, 2015) to generate 3D tetrahedrons without any post-processing. In our experiments, 8 of our created 9 surface meshes could be successfully voxelized. More testing results will be updated at our website `https://github.com/shugraphics/NeuroSkelMeshEvolution`. In addition, post-processings such as laplacian smoothing and global anisotropic analysis can be beneficial to TetGen voxelization.[A101,A102,A301]

In order to validate our approach, currently 424 neuronal morphology files from NeuroMorpho.Org are tested without deliberate selection. Totally 381 of them can be used to create membrane meshes, some of which are grouped in Table 3 for average attributes. In addition, all of the produced meshes are manifold and watertight with average vertex valence of 6. On the other hand, more results and corresponding mesh attributes will be updated on the website.[A103,A202,A203,A206,A207,A208,A210,A211,A219,A304]

3.4 Conclusion and Future Work

This paper presents a novel progressive approach for robustly generating high-quality 3D neuron models based on widely used point-and-diameter input of morphological tracings, such as those available in public repositories NeuroMorpho.Org. The final mesh is created by iteratively evolving an initial soma along the neuron skeletons. The adopted skeletal mapping policy assigns each vertex of the mesh to a definite skeletal node, which enables an efficient query of ROI. As described above, a sphere with uniform distributed vertices is set as the initial soma for subsequent evolution, which performs dynamic local refinement, simplification and convolution surface approximation to generate a smooth neuronal membrane mesh with adaptive vertex density distribution. Therefore, the whole neuronal surface is always a closed 2D manifold mesh throughout all the evolution stages. Due to the superposition property of the adopted convolution surfaces, the soma surface can be automatically created based on the embedded skeletons which are connected with each other at the soma. Actually the by-product of bumps at branches assists to

Table 2. Voxelization using TetGen (Si, 2015)

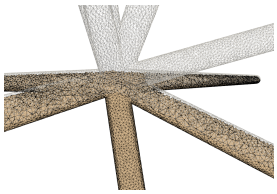
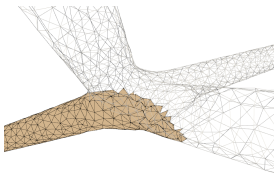
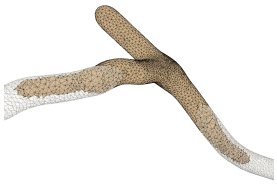
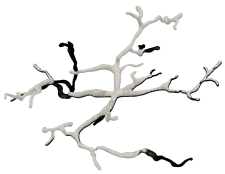
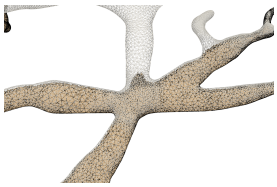
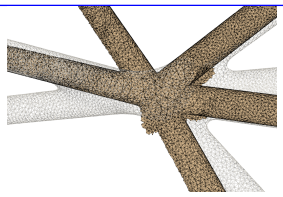
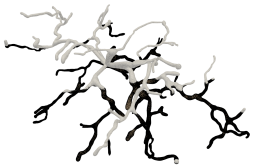
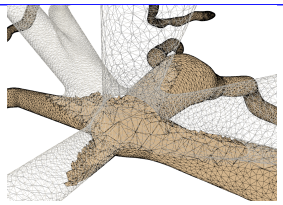
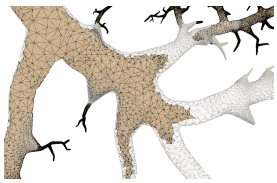
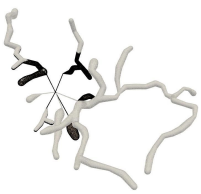
	Cell Name	Global	Local
Fig.2	A-MSN-1		
Fig.3	5-Som-3d-trace.CNG-part		
Fig.10	Fish015a_1		
Fig.11(A)	cell001_GroundTruth		
Fig.11(B)	cell021		
Fig.11(C)	cell002_GroundTruth		
Fig.12_part1	dHSE_011		
Fig.12_part2	dHSE_01r	failure	
Fig.14	cell011		

Table 3. Attributes of neuronal morphologies from NeuroMorpho.Org and produced membrane meshes averaged from groups of data.[A208,A210,A304]

Morphology Attributes				Membrane Mesh Attributes					
Archives	Species	Brain Regions	Models	Facets	Average Facet Area	Vertices	Average Valence	Manifold	Watertight
A. Mortensen	rat	neocortex	13	1014973	0.501	507484	6	100.00%	100.00%
Aharon_Zuo	mouse	neocortex	12	1063122	0.934	531562	6	100.00%	100.00%
Abrous	mouse	hippocampus	18	1189064	0.400	594158	6	100.00%	100.00%
Acharya	mouse	hippocampus	15	1303338	0.728	674744	6	100.00%	100.00%
Bock	drosophila melanogaster	protocerebrum	15	1875781	0.270	959308	6	100.00%	100.00%
Cardona	drosophila melanogaster	protocerebrum	14	576926	0.272	288457	6	100.00%	100.00%
Baier	zebrafish	optic lobe	15	97976	0.819	75959	6	100.00%	100.00%
Borst	drosophila melanogaster	optic lobe	20	877115	3.226	438454	6	100.00%	100.00%
Badea	mouse	retina	5	200818	0.910	100400	6	100.00%	100.00%
Baier	zebrafish	peripheral nervous system	5	71776	1.094	35890	6	100.00%	100.00
Adori	mouse	peripheral nervous system	8	453986	0.500	226984	6	100.00%	100.00
Badea	mouse	peripheral nervous system	11	453986	1.488	226984	6	100.00%	100.00
Almuhtasib	mouse	basal ganglia	14	1764502	0.276	1596530	6	100.00%	100.00
Anderson	mouse	basal ganglia	5	843365	0.340	421684	6	100.00%	100.00
Alzheimer	rat	basal ganglia	11	1100110	1.664	550028	6	100.00%	100.00
Arenkiel	mouse	main olfactory bulb	5	502111	0.284	251068	6	100.00%	100.00
Baier	zebrafish	main olfactory bulb	5	287455	0.844	143726	6	100.00%	100.00
Adke_ Carrasquillo	mouse	amygdala	5	138940	0.924	175644	6	100.00%	100.00
Argue	rat	amygdala	7	212161	0.661	106082	6	100.00%	100.00

create a plump shape without any special soma processing. In addition, to accelerate the computation-intensive convolution field generation, an analytical local convolution surface approximation is employed to approximate the line-skeletons in the form of point-and-diameter.

neuronal membrane surfaces with high-quality meshes and smooth branches can be robustly achieved using our approach, but the iterative deformation sacrifices the generation efficiency compared to a global mesh creation. Therefore, the advantageous superposition of convolution surfaces can be exploited to perform a "divide-and-conquer" policy for the whole neuron surface mesh generation in our future work.

Moreover, a better soma can be modeled based on advanced physical simulation and contour-constrained implicit surface fitting, and it is worthwhile to extract a more precise soma model from raw volume data as our initial evolution mesh. Finally, our created mesh model is a 2D manifold surface which can be directly 3D printed for physical visualization and education purposes.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

XZ and YW contributed to conception and design of the study. XL programed the functions of the project and wrote the first draft of the manuscript. SL and YS designed the workflow of the adopted algorithms and prepared the required testing data. All authors contributed to manuscript revision, read, and approved the submitted version.

SUPPLEMENTAL DATA AND SOURCE CODE

More information about our experiments, executable program, source code and online portal can be found at `https://github.com/shugraphics/NeuroSkelMeshEvolution`. The input swc files can be found at NeuroMorpho.Org, and the testing swc data and the produced corresponding mesh are all listed for downloading. Moreover, the statistical data are also illustrated for reference. If needed, the executable program can be downloaded for testing, and the updated versions will be provided in future. Source codes should also be uploaded onto github after we checking that all adopted packages have the required licences. Finally, we are configuring an online portal for converting swc data into membrane meshes. ^{A105,A209}

REFERENCES

Abdellah, M., Foni, A., Zisis, E., Guerrero, N. R., Lapere, S., Coggan, J. S., et al. (2021). Metaball skinning of synthetic astroglial morphologies into realistic mesh models for in silico simulations and visual analytics. Bioinformatics 37, i426–i433

Abdellah, M., Guerrero, N. R., Lapere, S., Coggan, J. S., Keller, D., Coste, B., et al. (2020). Interactive visualization and analysis of morphological skeletons of brain vasculature networks with vessmorphovis. Bioinform. 36, i534–i541. doi:10.1093/bioinformatics/btaa461

Abdellah, M., Hernando, J., Antille, N., Eilemann, S., Markram, H., and Schürmann, F. (2017). Reconstruction and visualization of large-scale volumetric models of neocortical circuits for physically-plausible in silico optical studies. BMC bioinformatics 18, 402

Abdellah, M., Hernando, J., Eilemann, S., Lapere, S., Antille, N., Markram, H., et al. (2018). Neuromorphovis: a collaborative framework for analysis and visualization of neuronal morphology skeletons reconstructed from microscopy stacks. Bioinform. 34, i574–i582. doi:10.1093/bioinformatics/bty231

Ahmed, A. G., Guo, J., Yan, D.-M., Franceschia, J.-Y., Zhang, X., and Deussen, O. (2016). A simple push-pull algorithm for blue-noise sampling. IEEE transactions on visualization and computer graphics 23, 2496–2508

Akkouche, S. and Galin, E. (2001). Adaptive implicit surface polygonization using marching triangles. In Computer Graphics Forum (Wiley Online Library), vol. 20, 67–80

Alliez, P., Ucelli, G., Gotsman, C., and Attene, M. (2008). Recent advances in remeshing of surfaces. Shape analysis and structuring , 53–82

Bloomenthal, J. (1994). An implicit surface polygonizer. Graphics gems 4, 324–350

Bloomenthal, J. and Shoemake, K. (1991). Convolution surfaces. In Proceedings of the 18th annual conference on Computer graphics and interactive techniques. 251–256

Botsch, M. and Kobbelt, L. (2004). A remeshing approach to multiresolution modeling. In Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing. 185–192

Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Lévy, B. (2010). Polygon mesh processing (CRC press)

Bottino, A., Nuij, W., and van Overveld, C. (1996). How to shrinkwrap a critical point: an algorithm for the adaptive triangulation of iso-surfaces with arbitrary topology. In Proceedings of IS96, the second Eurograhics/Siggraph workshop on Implicit Surfaces. 73

Brito Menéndez, J. P., Mata Fernández, S., Bayona Beriso, S., Pastor Pérez, L., DeFelipe, J., and Benavides-Piccione, R. (2013). Neuronize: a tool for building realistic neuronal cell morphologies. Frontiers in Neuroanatomy

Carnevale, N. T. and Hines, M. L. (2006). The NEURON book (Cambridge University Press)

Cheng, S.-W., Dey, T. K., Shewchuk, J., and Sahni, S. (2013). Delaunay mesh generation (CRC Press Boca Raton)

Du, Q., Faber, V., and Gunzburger, M. (1999). Centroidal voronoi tessellations: Applications and algorithms. SIAM review 41, 637–676

Du, X., Liu, X., Yan, D.-M., Jiang, C., Ye, J., and Zhang, H. (2018). Field-aligned isotropic surface remeshing. In Computer Graphics Forum (Wiley Online Library), vol. 37, 343–357

Dunyach, M., Vanderhaeghe, D., Barthe, L., and Botsch, M. (2013). Adaptive remeshing for real-time mesh deformation. In Eurographics 2013 (The Eurographics Association)

Ebeida, M. S., Rushdi, A. A., Awad, M. A., Mahmoud, A. H., Yan, D.-M., English, S. A., et al. (2016). Disk density tuning of a maximal random packing. In Computer Graphics Forum (Wiley Online Library), vol. 35, 259–269

Eberhard, J. P., Wanner, A., and Wittum, G. (2006). Neugen: a tool for the generation of realistic morphology of cortical neurons and neural networks in 3d. Neurocomputing 70, 327–342

Erleben, K., Sporring, J., Henriksen, K., and Dohlmann, H. (2005). Physics-based animation (Charles River Media Hingham)

Garcia-Cantero, J. J., Brito, J. P., Mata, S., Bayona, S., and Pastor, L. (2017). Neurotessmesh: A tool for the generation and visualization of neuron meshes and adaptive on-the-fly refinement. Frontiers in Neuroinformatics 11. doi:10.3389/fninf.2017.00038

Glaser, J. R. and Glaser, E. M. (1990). Neuron imaging with neurolucida-a pc-based system for image combining microscopy. Computerized Medical Imaging and Graphics 14, 307–317

Gleeson, P., Steuber, V., and Silver, R. A. (2007). neuroconstruct: a tool for modeling networks of neurons in 3d space. Neuron 54, 219–235

Hart, J. C. and Baker, B. (1996). Implicit modeling of tree surfaces. In Proceedings of Implicit Surfaces' 96 (Citeseer), 143–152

Jin, X. and Tai, C.-L. (2002). Analytical methods for polynomial weighted convolution surfaces with various kernels. Computers & Graphics 26, 437–447

Jin, X., Tai, C.-L., and Zhang, H. (2009). Implicit modeling from polygon soup using convolution. The Visual Computer 25, 279–288

Kil, Y. J., Renzulli, P., Kreylos, O., Hamann, B., Monno, G., and Staadt, O. G. (2006). 3d warp brush modeling. Computers & Graphics 30, 610–618

Lasserre, S., Hernando, J., Hill, S., Schuermann, F., de Miguel Anasagasti, P., Abou Jaoudé, G., et al. (2011). A neuron membrane mesh representation for visualization of electrophysiological simulations. IEEE Transactions on Visualization and Computer Graphics 18, 214–227

Liu, Y.-J., Xu, C., Yi, R., Fan, D., and He, Y. (2016). Manifold differential evolution (mde): a global optimization method for geodesic centroidal voronoi tessellations on meshes. ACM Trans. Graph. 35, 243–1

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. In Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987, Anaheim, California, USA, July 27-31, 1987, ed. M. C. Stone (ACM), 163–169. doi:10.1145/37401.37422

McCormack, J. and Sherstyuk, A. (1998). Creating and rendering convolution surfaces. In Computer Graphics Forum (Wiley Online Library), vol. 17, 113–120

McDougal, R. A., Hines, M. L., and Lytton, W. W. (2013). Water-tight membranes from neuronal morphology files. Journal of Neuroscience Methods 220, 167–178

Mörschel, K., Breit, M., and Queisser, G. (2017). Generating neuron geometries for detailed three-dimensional simulations using anamorph. Neuroinformatics 15, 247–269. doi:10.1007/s12021-017-9329-x

Si, H. (2015). Tetgen, a delaunay-based quality tetrahedral mesh generator. ACM Trans. Math. Softw. 41, 11:1–11:36

Stanculescu, L., Chaine, R., and Cani, M.-P. (2011). Freestyle: Sculpting meshes with self-adaptive topology. Computers & Graphics 35, 614–622

Vaillant, R., Barthe, L., Guennebaud, G., Cani, M., Rohmer, D., Wyvill, B., et al. (2013). Implicit skinning: real-time skin deformation with contact modeling. ACM Trans. Graph. 32, 125:1–125:12

Van Overveld, K. and Wyvill, B. (2004). Shrinkwrap: An efficient adaptive algorithm for triangulating an iso-surface. The visual computer 20, 362–379

Vorsatz, J., Ro¨ ssl, C., and, and Seidel, H.-P. (2003). Dynamic remeshing and applications. J. Comput. Inf. Sci. Eng. 3, 338–344

Wang, Y., Yan, D.-M., Liu, X., Tang, C., Guo, J., Zhang, X., et al. (2018). Isotropic surface remeshing without large and small angles. IEEE transactions on visualization and computer graphics 25, 2430–2442

Wilson, M., Bhalla, U., Uhley, J., and Bower, J. (1988). Genesis: A system for simulating neural networks. Advances in neural information processing systems 1

Wyvill, G., McPheeters, C., and Wyvill, B. (1986). Soft objects. In Advanced Computer Graphics (Springer). 113–128

Yan, D.-M. and Wonka, P. (2015). Non-obtuse remeshing with centroidal voronoi tessellation. IEEE transactions on visualization and computer graphics 22, 2136–2144

Zhu, X., Guo, X., and Jin, X. (2013). Efficient polygonization of tree trunks modeled by convolution surfaces. Science China Information Sciences 56, 1–12