Abstract—Objective: Recent advances in electron microscopy have, for the first time, enabled imaging of single cells in 3D at a nanometer length scale resolution. An uncharted frontier for in silico biology is the ability to simulate cellular processes using these observed geometries. However, this will require a system for going from EM images to 3D volume meshes which can be used in finite element simulations. Methods: In this paper, we develop an end-to-end pipeline for this task by adapting and extending computer graphics mesh processing and smoothing algorithms. Our workflow makes use of our recently rewritten mesh processing software, GAMer 2, which implements several mesh conditioning algorithms and serves as a platform to connect different pipeline steps. Results: We apply this pipeline to a series of electron micrographs of dendrite morphology explored at three different length scales and show that the resultant meshes are suitable for finite element simulations. Conclusion: Our pipeline, which consists of free and open-source community driven tools, is a step towards routine physical simulations of biological processes in realistic geometries. Significance: We posit that a new frontier at the intersection of computational technologies and single cell biology is now open. Innovations in algorithms to reconstruct and simulate cellular length scale phenomena based on emerging structural data will enable realistic physical models and advance discovery.

Index Terms—Mesh Generation, Finite Elements Simulation, Volume Electron Microscopy, Cell Modeling, Reaction-Diffusion, Dendritic Spines

LIST OF ACRONYMS
BPAP Back Propagating Action Potential
EM Electron Microscopy
EPSP Excitatory Postsynaptic Potential
ER Endoplasmic Reticulum
FEA Finite Element Analysis
FIB-SEM Focused-ion Beam Milling Scanning Electron Microscopy
LST Local Structure Tensor
NMDAR N-methyl-D-aspartate Receptor
PM Plasma Membrane
PSD Postsynaptic Density
SBF-SEM Serial Block-Face Scanning Electron Microscopy

I. INTRODUCTION

UNDERSTANDING structure-function relationships at the cellular length scales (nm to µm) is one of the central goals of modern cell biology. While structural determination techniques are routine for very small and large scales such as molecular and tissue, high-resolution images of mesoscale subcellular scenes were historically elusive. This was primarily
due to the diffraction limits of visible light and the limitations of X-ray and Electron Microscopy (EM) hardware. Over the past decade, technological improvements such as improved electron direct detectors have enabled the practical applications of techniques such as volume electron microscopy. Advances in microscopy techniques in recent years have opened windows into cells, giving us insight into cellular organization with unprecedented detail. Volumetric EM enables the capture of 3D ultrastructural datasets (i.e., images where fine structures such as membranes of cells and their internal organelles are resolved, as shown in (Fig. 1A)). Using these geometries as the basis of simulations provides an opportunity for in silico animation of various cellular processes and the generation of experimentally testable hypotheses. However, to the best of our knowledge, there is no current free and open-source system for going from EM images to 3D meshes, which can be used in finite element simulations.

Here, we introduce our recently redesigned software, GAMer (Geometry-preserving Adaptive MeshER), as a tool connecting and implementing methods for 3D meshing and reconstruction of cellular electron micrographs (Fig. 1). The input to this pipeline are stacks of high-resolution images (Fig. 1A) from which contours of different organelles and cellular components that are separated by membranes are segmented and identified (Fig. 1B). These steps are often done in experimental and imaging labs. Superposition of these 2D contours based on the known z-separation gives us the first 3D contour data set (Fig. 1B). Traced contours are then constructed into a primitive 3D surface mesh using IMOD (Fig. 1C) [2]. The surface meshes of the external and internal membranes are then conditioned to improve element quality via GAMer; faces are also marked using a GAMer Blender interface to allow for different boundary conditions during simulation (Fig. 1D) [3]. TetGen takes these high-quality surface meshes as inputs and generates an unstructured tetrahedral mesh (Fig. 1E) [4]. This mesh is now ready for finite element based physical simulation of different cellular processes (Fig. 1F).

II. IMAGE ACQUISITION AND SEGMENTATION

We briefly summarize sample preparation in this section for completeness. Sample preparation begins with either cell culture or harvesting of biological tissues of interest. Subsequent preparation steps can vary depending upon the particular volume EM imaging modality used but primarily include sample dehydration, fixation/staining, embedding, and imaging through the different cross-sections [5]–[8]. Once the images are captured, in order to construct an initial mesh model from the data, the boundaries of features must first be identified. Much of this relies on the expertise of biologists for recognition of organelles and membrane domains in cells. During the segmentation process, the algorithm or researcher must carefully separate boundary signal from noise. Various schemes ranging from manual tracing, thresholding and edge-detection, to deep-learning based approaches have been employed to perform image segmentation [8].
A. Meshing challenges

A variety of challenges for meshing and subsequent physical simulations can arise at each step. Even with perfect experimental execution, and despite the enhanced surface contrast from heavy metal stains, the membranes of cells and their internal organelles are often poorly behaved and contain sharp and otherwise irregular geometries that are difficult to segment. In more serious cases, thinly sliced samples can tear or become contaminated during handling. Methodological errors are also possible. For example, Serial Block-Face Scanning Electron Microscopy (SBF-SEM) datasets in optimum conditions may have 3 nm lateral (x,y) resolution but 25 nm axial (z) resolution, limited by the slicing capability of the ultramicrotome [5]. Anisotropic resolution in tandem with variable slice thickness can cause loss of axial detail.

There are many EM software which post-process image stacks to correct for these and other artifacts not mentioned. Most of our datasets have been manually segmented and corrected in software such as IMOD [2]. IMOD has the capacity to perform a contour-tiling operation to generate a preliminary surface mesh suitable for basic 3D visualization. These meshes, however, are not directly suitable for physical simulations due to various mesh artifacts. Some of these include intersecting faces, non-manifold features, and high aspect ratio faces, as shown in Fig. 2. To resolve these problems, we perform surface mesh conditioning steps described below.

III. METHODS

A. Mesh Processing

Our main contribution in this work is a surface mesh processing library GAMer 2, which features algorithms as described by Yu et al. [9], [10]. We have recently rewritten GAMer in C++ using the CASC data structure [11] as the underlying mesh representation. The code is licensed under LGPL v2.1 and can be obtained from GitHub (https://github.com/clee/gamer) [12]. GAMer can be used as a stand-alone library, or alternatively the algorithms can be accessed through a GAMer Blender add-on. Blender not only provides a customizable mesh visualization environment, but also tools such as sculpt mode, which allows users to flexibly manipulate the geometry [3]. We briefly review the concepts behind the mesh processing algorithms from Yu et al. [9], [10].

1) Local Structure Tensor: The mesh processing operations in GAMer are designed to preserve the local geometry; algorithms seek to respect the geometric ground truth observed in the micrographs. We use a Local Structure Tensor (LST) to account for the local geometry [13]–[15]. The LST is defined as follows,

\[
T(v) = \sum_{i=1}^{N_v} n_i \otimes n_i = \sum_{i=1}^{N_v} \begin{pmatrix} n_i^x n_i^x & n_i^x n_i^y & n_i^x n_i^z \\ n_i^y n_i^x & n_i^y n_i^y & n_i^y n_i^z \\ n_i^z n_i^x & n_i^z n_i^y & n_i^z n_i^z \end{pmatrix},
\]

where \(v\) is the vertex of interest, \(N_v\) is the number of neighbors in the \(r\)-ring neighborhood, and \(n_i^x, n_i^y, n_i^z\) form the normal of the \(i\)th neighbor vertex. Vertex normals are defined as the weighted average of incident face normals. Performing the eigendecomposition of the LST, we obtain information on the

principal orientations of normals in the local neighborhood [16]. The magnitude of the eigenvalue corresponds to the amount of curvature along the direction of the corresponding eigenvector. Inspecting the magnitude of the eigenvalues gives several geometric cases:

- Planes: \(\lambda_1 \gg \lambda_2 \approx \lambda_3 \approx 0\)
- Ridges and valleys: \(\lambda_1 \approx \lambda_2 \gg \lambda_3 \approx 0\)
- Spheres and saddles: \(\lambda_1 \approx \lambda_2 \approx \lambda_3 > 0\)

2) Feature preserving mesh smoothing: Finite elements simulations are sensitive to the quality of the mesh. Poor quality meshes can lead to unbounded error, numerical instability, long times to solution, and non-convergence. Generally, triangulations with high aspect ratios produce larger errors compared with equilateral elements [17].

To improve the conditioning of the surface meshes derived from microscopy images, we use an angle-weighted Laplacian smoothing approach, as shown in Fig. 3A. This scheme is an extension to meshes embedded in 3D of the angle weighted smoothing scheme described by Zhou and Shimada for meshes embedded in 2D [18]. In essence, this algorithm applies local torsion springs to the 1-ring neighborhood of a vertex of interest to balance the angles.

Given a vertex \(x\) with the set of 1-ring neighbors \(\{v_1, \ldots, v_N\}\), where \(N\) is the number of neighbors, ordered
such that $\mathbf{v}_i$ is connected to $\mathbf{v}_{i-1}$ and $\mathbf{v}_{i+1}$ by edges. The 1-ring is connected such that $\mathbf{v}_{N+1} := \mathbf{v}_1$ and $\mathbf{v}_{-1} := \mathbf{v}_N$. Traversing the 1-ring neighbors, we define edge vectors $\mathbf{e}_{i-1} := \mathbf{v}_i \mathbf{v}_{i-1}$ and $\mathbf{e}_{i+1} := \mathbf{v}_i \mathbf{v}_{i+1}$. This algorithm seeks to move $\mathbf{x}$ to lie on the perpendicularly bisecting plane $\Pi_i$ of $\angle(\mathbf{v}_{i-1}, \mathbf{v}_i, \mathbf{v}_{i+1})$. For each vertex in the 1-ring neighbors, we compute the perpendicular projection, $\mathbf{x}_i$, of $\mathbf{x}$ onto $\Pi_i$. Since small surface mesh angles are more sensitive to change in position than large angles, we prioritize their maximization. We define a weighting factor, $\alpha_i = \frac{|\mathbf{e}_{i-1}| \cdot |\mathbf{e}_{i+1}|}{|\mathbf{e}_{i-1}|^2 + |\mathbf{e}_{i+1}|^2}$, which inversely corresponds with $\angle(\mathbf{v}_{i-1}, \mathbf{v}_i, \mathbf{v}_{i+1})$. The average of the projections weighted by $\alpha_i$ gives a new position of $\mathbf{x}$ as follows,

$$
\bar{\mathbf{x}} = \frac{1}{\sum_{i=1}^{N} (\alpha_i + 1)} \sum_{i=1}^{N} (\alpha_i + 1) \mathbf{x}_i.
$$

There are many smoothing algorithms in the literature; the angle-weighted Laplacian smoothing algorithm described here can outperform other popular smoothing strategies such as those described in [19] and [20] which are primarily focused on optimizing the smoothness of surface normals for computer graphics applications and not mesh angles. Our goal is not to provide an elaborate comparison against existing algorithms in this manuscript but to demonstrate the utility of our pipeline for biological images. These images can produce meshes that often contain hundreds of thousands to millions of faces and can cause global optimization based algorithms to fail. Therefore, our approach is a local operation, making it particularly suitable for cellular images.

Conceptually the fidelity of the local geometry can be maintained by restricting vertex movement along directions of low curvature. This constraint is achieved by anisotropically dampening vertex diffusion using information contained in the LST. Although the weighted vertex smoothing scheme, as described, will reasonably preserve geometric structure, the structure preservation can be further improved by using the LST. Computing the eigendecomposition of the LST, we obtain eigenvalues $\lambda_1, \lambda_2, \lambda_3$ and eigenvectors $\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$, which correspond to principal orientations of local normals. We project $\bar{\mathbf{x}} - \mathbf{x}$ onto the eigenvector basis and scale each component by the inverse of the corresponding eigenvalue,

$$
\hat{\mathbf{x}} = \mathbf{x} + \sum_{k=1}^{3} \frac{1}{1 + \lambda_k} [(\bar{\mathbf{x}} - \mathbf{x}) \cdot \mathbf{E}_k] \mathbf{E}_k.
$$

This has the effect of dampening movement along directions of high curvature i.e., where $\lambda$ is large. In this fashion, our algorithm not only improves triangle aspect ratios, but does so while preserving local geometric features. We note that our actual implementation iterates between rounds of vertex smoothing and conventional angle based edge flipping to achieve the desired smoothing effect. Edge flips are common in mesh processing, and provide a mechanism for both improving angles and reducing the valency of vertices. A comparison of the angle-weighted smoothing algorithm with and without LST correction is shown in Fig. 4.

3) Feature preserving mesh decimation: The number of degrees of freedom in the mesh influences the computational burden of subsequent physical simulations. One strategy to reduce the number of degrees of freedom is to perform mesh decimation or simplification.

There are many strategies for decimation including topology preserving Euler operators and other algorithms such as vertex clustering which do not guarantee topological invariance. It is typically desirable to preserve the mesh topology for physical simulation based applications. Conventional Euler operations for mesh decimation include vertex removal, edge collapse, and half-edge collapse. As noted earlier, finite elements sim-
ulations are sensitive to angles of the mesh. Edge and half-edge collapses can sometimes lead to vertices with high or low valency and therefore poor angles. To avoid this problem we employ the vertex removal algorithm. First, vertices to be decimated are selected based upon some criteria, discussed below. We then remove the vertex and re-triangulate the resulting hole. This is achieved using a recursive triangulation approach, which heuristically balances the edge valency. Given the boundary loop, we first connect vertices with the fewest incident edges. This produces two resulting holes that we then fill recursively using the same approach. When a hole contains only three boundary vertices, they are connected to make a face. We note that while this triangulation scheme balances vertex valency, it may degrade mesh quality. We solve this by running the geometry preserving smoothing algorithm on the local region.

We employ two criteria for selecting vertices to remove. First, to selectively decimate vertices in low or high curvature regions, again information from the LST can be used. By comparing the magnitudes of the eigenvalues of the LST we can select for regions with different geometries. For example, to decimate vertices in flat regions of the mesh, given eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$, vertices can be selected by checking if the local region satisfies,

$$\frac{\lambda_2}{\lambda_1} < R_1,$$  

(4)

where $R_1$ is a user specified flatness threshold (smaller is flatter). In a similar fashion, vertices in curved regions can also be selected. However, decimation of curved regions is typically avoided due to the potential for losing geometric information.

Instead, to simplify dense areas of the mesh, we employ an edge length based selection criterion,

$$\max_{i=1}^{N_1} d(x, v_i) < D,$$  

(5)

where $N_1$ is the number of vertices in the 1-ring neighborhood of vertex $x$, $d(\cdot, \cdot)$ is the distance between vertices $x$ and $v_i$, $D$ is the mean edge length of the mesh, and $R_2$ is a user specified threshold. This criterion allows us to control the sparseness of the mesh. We note that the aforementioned criteria are what is currently implemented in GAMer, however the vertex removal decimation scheme can be employed with any other selection criteria.

4) Feature preserving anisotropic normal-based smoothing: To remove additional bumpiness from the mesh, we use a normal-based smoothing approach [21], [22], as shown in Fig. 3B. The goal is to produce smoothly varying normals across the mesh without compromising mesh angle quality. Given a vertex $x$ of interest, for each incident face $i$, with normal $n_i$ we rotate $x$ around a rotation axis defined by opposing edge $e_i$ such that $n_i$ aligns with the mean normal of neighboring faces $n_i = \sum_{j=1}^{3} n_{ij}/3$. We denote the new position which aligns $n_i$ and $n_{ij}$ as $R(x; e_i, \theta_i)$. Summing up the rotations and weighting by incident face area, $a_i$, we get an updated position,

$$x = \frac{1}{\sum_{i=1}^{N_1} a_i} \sum_{i=1}^{N_1} a_i R(x; e_i, \theta_i).$$  

(6)

This is an isotropic scheme which is independent of the local geometric features; meaning that many iterations of this algorithm may weaken sharp features.

Instead, we use an anisotropic scheme [22], [23] to compute the mean neighbor normals,

$$\bar{n}_i = \frac{1}{\sum_{j=1}^{3} e^{K(n_i - n_{ij})}} \sum_{j=1}^{3} e^{K(n_i - n_{ij})} n_{ij},$$  

(7)

where $K$ is a user defined positive parameter which scales the extent of anisotropy. Under this scheme, the weighting function decreases as a function of the angle between $n_i$ and $n_{ij}$ resulting in the preservation of sharp features.

5) Boundary marking and tetrahedralization: To support the definition of boundary conditions on the mesh, it is conventional to assign boundary marks or identifiers which correspond to different boundary definitions in the physical simulation. In simplified and idealized geometries it is possible to define functions to assign boundary values. However, in subcellular scenes where the geometry may be tortuous and local receptor clusters can be arbitrarily distributed on the manifold, boundary definition is a non-trivial challenge. The GAMer Blender add-on supports the facile user-based definition of boundary markers on the surface [3]. Users can utilize any of the face selection methods which Blender provides to select boundaries to mark. Boolean operations and other geometric strategies provided natively in Blender can also be used for selection. Boundary markers are associated with a unique material property which helps visually delineate marked assignments. After boundaries are marked, stacks of surface meshes corresponding to different domains can be grouped and passed through GAMer into TetGen for tetrahedralization [4].

B. Mesh Generation Pipeline

Electron micrographs and segmentations from Wu et al. [1] were graciously shared by de Camilli and coworkers (Fig. 1A, B). We generated preliminary meshes of the geometry using the imod2obj utility included with IMOD [2] (Fig. 1C). The quality of the mesh was improved using algorithms described in Section III-A and implemented in GAMer 2.0.1 [12] (Fig. 1D). Some features, such as disconnections of the Endoplasmic Reticulum (ER), were manually reconnected using Blender mesh sculpting features. Additional discussion of the mesh artifacts and the curation process is described in the example applications. Boundaries were marked and the conditioned surface mesh was tetrahedralized using TetGen [4] (Fig. 1E). Simulations of a reaction-diffusion system were run for a full dendritic branch geometry using FEniCS [24] (Fig. 1F).

IV. APPLICATION OF THE PIPELINE TO DIFFERENT SYSTEMS

We demonstrate the application of our pipeline to dendritic spine reconstructions of different sizes. All images of dendritic spines are from neurons taken from mouse cerebral cortex or nucleus accumbens from a recent publication using Focused-Beam Milling Scanning Electron Microscopy (FIB-SEM).
In addition to their important role in synaptic and structural plasticity, these cellular structures demonstrate highly tortuous morphologies, high surface-to-volume ratios, and a geometric intricacy that serves as a good test-bed for our pipeline.

We have generated Finite Element Analysis (FEA) compatible meshes of several geometries of increasing length scale: the ER of the single spine geometry which requires nanometer precision (Fig. 5A–F), the Plasma Membrane (PM) of the single spine which has a length scale of a couple of microns (Fig. 5G), the two spine geometry, a few microns (Fig. 5H), and the dendrite with about 40 spines, with a length scale in the tens of microns (Fig. 7). We describe some of the challenges associated with the mesh conditioning process for each system. We demonstrate the utility of this pipeline by showing a simulated time-series of calcium influx due to N-methyl-D-aspartate Receptor (NMDAR) by a Back Propagating Action Potential (BPAP) and Excitatory Postsynaptic Potential (EPSP) in the full dendritic segment geometry (Figs. 8 and 9). We also illustrate this process in more detail. This spine also contains a specialized form of ER termed the spine apparatus, Fig. 5C, F, which consists of seven folded cisternae. This highly organized structure bears geometrical similarities to a parking garage structure and helicoidal geometries [26]–[29]. The geometric detail of the spine apparatus is preserved by the conditioning process in our pipeline.

In Fig. 5 we show the distribution of the angles of the surface mesh before and after conditioning. One metric of a well-conditioned mesh is that all the surface triangles are nearly equilateral [17]. Prior to conditioning, the angle distribution is spread out and contains many large and small angles. After processing using GAmer, the angles of the mesh are improved, as indicated by the peaked distribution around 60 degrees. Fig. 5C shows a closeup of high aspect ratio triangles before and after processing with GAmer.

It is worth noting that even though this mesh of PM of the single spine is the simplest geometry we examine, there are many intersecting faces in the initial mesh. This is a common artifact which appears in most initial meshes but is easily and automatically resolved by GAmer’s vertex smoothing algorithm as shown in Fig. 5B.

Although the ER structure is significantly more complex, the angles of the mesh are also improved, albeit to a lesser extent than the PM. In scenarios such as this where the length scales of interest are closer to the acquisition resolution, it may be necessary to increase the number of triangles to accurately capture the fine details. Table I summarizes the number of vertices and triangles in the initial vs conditioned meshes as well as vertices and tetrahedra in the resultant volumetric meshes. To accurately capture the curvature of the PM mesh in Fig. 5D about 48% more triangles were needed compared to the powers of EM.

For example, if the ER undergoes large conformational changes between z-slices then tubules may appear disconnected. Alternatively, the boundaries of the ER membrane may have poor contrast and can sometimes be missed during segmentation. In these cases, we manually curate the mesh to reconnect the broken ER segments, Fig. 5B. Fig. 5A1 and A2 illustrate this process in more detail. This spine also contains a specialized form of ER termed the spine apparatus, Fig. 5C, F, which consists of seven folded cisternae. This highly organized structure bears geometrical similarities to a parking garage structure and helicoidal geometries [26]–[29]. The geometric detail of the spine apparatus is preserved by the conditioning process in our pipeline.

| Geometry          | Initial PM | Conditioned PM | Initial ER | Conditioned ER |
|-------------------|------------|----------------|------------|----------------|
| Surface Mesh      | # Vertices | # Triangles    | # Vertices | # Tetrahedra   |
| Single Spine      | 4,695      | 9,330          | 6,546      | 19,654         |
| Conditioned PM    | 6,924      | 13,844         | 13,734     | 62,557         |
| Conditioned ER    | 36,294     | 72,620         | 53,134     | 211,018        |
| Two Spines        | Initial PM | 160,733        | 320,976    |                |
| Conditioned PM    | 18,027     | 36,050         | 28,989     | 122,082        |
| Conditioned ER    | 36,294     | 72,620         | 53,134     | 211,018        |
| Dendritic Segment | Initial PM | 207,448        | 410,896    |                |
| Conditioned PM    | 126,336    | 252,668        | 194,848    | 798,626        |

*Non-manifold and other mesh artifacts prevent the tetrahedralization of initial meshes
Fig. 5. Meshes of the single spine geometry pre-(A, B, C) and post-conditioning (D, E, F). Columns show (left) plasma membrane, (center) spine apparatus and endoplasmic reticulum, and (right) zoom in of spine apparatus. Note that there are two large holes in the preliminary plasma membrane mesh (A) corresponding to the start of the sample block-face. The distribution of angles for each mesh is shown. After processing in GAMer, the minimum and maximum angles of the mesh are improved and the angle distribution becomes peaked at 60 degrees. Scale bars: left, middle: 500 nm, right: 100 nm.
Fig. 6. Surface meshes of two neighboring dendritic spines. A) before conditioning and B) after. The angles of the mesh are improved by GAMer processing. Scale bars: top row: 1 µm, middle row: 50 nm.

The pipeline described here is also applicable for larger systems as we demonstrate with two spines and a full dendrite. The two spine geometry shown in Fig. 6 is a few microns in length. Based on the length scales we would expect a well conditioned mesh for this geometry to contain approximately double the number of triangles in the single spine mesh; however, the orientation of z-stacks in this mesh is different from that in the single spine geometry which led to an abnormally large number of triangles: 320,976 versus just 9,330 in the mesh of PM in the single spine. After GAMer conditioning algorithms were applied, the number of triangles was reduced to 36,050, a much more reasonable count. As demonstrated, our pipeline is robust and can handle cases where the initial mesh either generates too few or too many triangles as required for capturing geometric details.

At the tens of microns length scale, we constructed a mesh of a full dendritic segment. We show a zoomed in section of the mesh before and after conditioning in Fig. 7. As in the one and two spine cases, our system robustly handles artifacts such as poor quality triangles and intersecting faces; Fig. 7F shows that the distribution of the angles post conditioning are comparable to the one and two spine examples, showing that size does not alter the pipeline’s capability to produce well-conditioned meshes. Fig. 7B shows an intricate spine head with many different regions of Postsynaptic Density (PSD) shown in purple; Fig. 7E shows the preserved geometry post-conditioning and the PSD marked with GAMer for use as boundary conditions.

B. Simulation of reaction-diffusion equations in dendrites

Using the mesh of the dendritic segment (Fig. 7) we simulated NMDAR activation due to a BPAP and EPSP along the entire dendrite in Fig. 8, see also Movie S1 available online. Because the goal of this simulation was not to show biological accuracy, but rather to demonstrate that our pipeline is capable of producing biophysically relevant FEA simulations, we use a simplified version of the model found in Bell et al. [30].

We model a BPAP and EPSP which stimulates NMDAR opening and calcium ion influx into the cell. The reaction-diffusion of $u$, corresponding to calcium ion concentration, is described as follows,

$$\frac{\partial u}{\partial t} = D \nabla^2 u - \frac{u}{\tau} \quad \text{in } \Omega,$$

where $D$ is the diffusion coefficient, $\nabla^2$ is the Laplacian operator, $\tau$ is a characteristic decay time, and $\Omega$ is the volumetric domain. We define boundary conditions corresponding to the ionic flux through NMDA receptors, $J_{\text{NMDAR}}$, lining the post synaptic density, $\partial \Omega_{\text{psd}}$,

$$D (n \cdot \nabla u) = J_{\text{NMDAR}}(t) \quad \text{on } \partial \Omega_{\text{psd}},$$

where $n$ is the outwardly-oriented unit normal vector, and $J_{\text{NMDAR}}$ is of the form,

$$J_{\text{NMDAR}} = G_{\text{NMDAR}}(t) B(V)(V(t) - V_{\text{rest}}) \alpha.$$
Fig. 8. Time series of calcium dynamics from NMDA receptor opening in response to a prescribed membrane voltage trace in a full dendritic segment. Boundaries demarcating the plasma membrane (PM) and post synaptic density (PSD) are shown in blue and orange respectively (top). Snapshots of calcium ion concentration throughout the domain are also shown for several time points. We apply a voltage corresponding to a back propagating action potential and excitatory postsynaptic potential. NMDA receptors localized at the PSD membrane, open in response to the voltage and calcium flows into the cell. Over time, the NMDA receptors close, and calcium is scavenged by calcium buffers. Scale bar: 4 µm.

\[ G_{\text{NMDAR}}(t) \] is a variable conductance which accounts for deactivation of the receptor, \( B(V) \) is a term which accounts for Mg\(^{2+}\) inhibition, the voltage difference \( V(t) - V_{\text{rest}} \) is prescribed to emulate a BPAP and EPSP, and \( \alpha \) is a scaling term which groups factors such as probability of opening, receptor area density at the PSD, etc.

On the remainder of the plasma membrane which we denote as \( \partial \Omega_{\text{pm}} \), we enforce no-flux boundary conditions,

\[ D(n \cdot \nabla u) = 0 \quad \text{on} \quad \partial \Omega_{\text{pm}}, \quad (11) \]

At time \( t = 0 \), we set the initial concentration of calcium ions to naught throughout the volume of the dendrite,

\[ u(x, t = 0) = 0 \quad \text{in} \quad \bar{\Omega}. \quad (12) \]

Where \( \bar{\Omega} \) is the union of the volumetric and boundary domains,

\[ \bar{\Omega} \equiv \Omega \cup \partial \Omega. \quad (13) \]

The surface of the geometry is composed of only post synaptic density and plasma membrane,

\[ \partial \Omega \equiv \partial \Omega_{\text{psd}} \cup \partial \Omega_{\text{pm}}. \quad (14) \]

The model was solved using FEniCS [24] to run finite elements simulations. A backward Euler time discretization
scheme was used for its property of unconditional stability when solving linear problems; a timestep size of 417 μs was used. The resultant linear system of equations was solved using LU-decomposition.

The back propagating potential stimulates the entirety of the dendritic branch simultaneously, leading to the opening of NMDARs localized to the PSD. Calcium ions enter into the dendritic spine heads through the open NMDA receptors localized at the post synaptic density. We first observe that the calcium dynamics in spines are spine size, spine shape and PSD-dependent. We observe that the narrow spine necks act as a diffusion barrier to calcium, preventing diffusing calcium ions from entering the dendritic shaft, consistent with observations in the literature [31]–[34]. Fig. 9 shows several representative traces of calcium ion dynamics in different regions of the geometry.

This example demonstrates that the meshes produced by GAMer through the workflow are directly compatible with finite element simulations and will allow for the generation of biophysically relevant hypotheses.

C. Mesh convergence analysis

A common error metric used in FEA is the $L_2$ norm of the difference between a solution computed on a coarser mesh ($u'$) and a solution computed on a very fine mesh, which is taken to be the ground-truth ($u$), i.e.,

$$
\varepsilon_{L_2} = \left( \int_{\Omega} (u' - u)^2 \, d\Omega \right)^{1/2}.
$$

(15)

This is a standard procedure that is used to measure $h$-refinement convergence rates; however between iterations of GAMer algorithms the boundaries of the mesh are perturbed slightly. Attempting to use $\varepsilon_{L_2}$ as an error metric is problematic as its integrand is undefined in regions where $\Omega'$, the domain of $u'$, and $\Omega$ do not intersect.

Therefore, to illustrate the convergence of solutions as the mesh quality is improved using GAMer, we used an error metric based on the relative difference in total molecules,

$$
\varepsilon_{rel} = \left| \frac{\int_{\Omega'} u' \, d\Omega' - \int_{\Omega} u \, d\Omega}{\int_{\Omega} u \, d\Omega} \right|.
$$

(16)

Fig. 10 (A-D) shows intermediary steps of the GAMer refinement process. For each mesh, a given number of smoothing iterations was performed; any remaining artifacts that would prevent tetrahedralization such as intersecting faces were removed and the resultant holes were re-triangulated. The surface meshes were all tetrahedralized using TetGen with the same parameters. The first time step of a reaction-diffusion partial differential equation with a constant Neumann boundary condition was solved and the resulting solutions were compared using Eq. (16), where Fig. 10E was assumed to be the ground-truth. Fig. 10F shows that the relative error consistently decreased as a result of further mesh conditioning in GAMer. The mesh with no smoothing operations applied had almost five times more molecules than the ground-truth mesh after just one time step. This analysis highlights not only the importance of using a high quality mesh in FEA but also that GAMer can generate such high quality meshes.

V. Discussion

The relationship between cellular shape and function is being uncovered as systems, structural biology, and physical simulations converge. Beyond traditional compartmentalization, plasma membrane curvature and cellular ultrastructure have been shown to affect the diffusion and localization of molecular species in cells [35], [36]. For example, fluorescence experiments have shown that the dendritic spine necks act as a diffusion barrier to calcium ions, preventing ions from entering the dendritic shaft [31]. Complementary to this and other experiments, various physical models solving reaction-diffusion equations in idealized geometries have been developed to further interrogate the structure-function relationships [30], [35], [37]–[40].

An important next step will be to expand the spatial realism of these models to incorporate realistic geometries as informed by volume imaging modalities. Our tool GAMer serves as an important step towards filling the need for community driven tools to generate meshes from realistic biological scenes. We have demonstrated the utility of the mesh conditioning algorithms implemented in GAMer for a variety of systems across several length scales and upwards of hundreds of thousands of triangles. The volume meshes that result from our tools are high quality (Fig. 10E) and we show that they can be used directly in finite elements simulations of reaction-diffusion systems (Fig. 8).

Bundled with GAMer we include a Blender add-on which exposes our mesh conditioning algorithms to the Blender environment. Blender acts as a user interface that provides visual feedback on the effects of GAMer mesh conditioning operations. Blender also enables the painting of boundaries using its many mesh selection tools. Beyond the algorithms in GAMer, Blender also provides an environment for manual curation of mesh artifacts.
Current meshing methods are limited by the need for human biological insight. Experimental setups for volume electron microscopy are arduous and often messy. Microscopists take great care to optimize the experimental conditions, however small variations can lead to sample contamination, tears, precipitation of stain, or other problems. Many of these issues will manifest as artifacts on the micrographs, which makes it challenging to evaluate the ground truth. Automated segmentation algorithms using computer vision and machine learning approaches can fail as a result of these artifacts, and biologists will default to the time-tested, reliable but error-prone mode of manually tracing boundaries.

This is a unique opportunity for biological mesh generation to differentiate from other meshing tools employed in other engineering disciplines. To account for the problems induced by biology and wet experiments along with physical simulations, we anticipate that, to realize an automated mesh generation pipeline will require the development of specialized algorithms which tightly couple information across the workflow. As additional annotated datasets become available, machine learning models can be trained to perform tasks which are currently manually executed such as reconnecting disconnected ER tubules.

To accelerate the goal of achieving routine cell modeling with realistic geometries, experimentalists can contribute by sharing segmented datasets from their work along with biological questions of interest. In exchange, modelers can generate testable predictions and measurements inaccessible to current experimental methods. Through this interdisciplinary exchange, any gaps in our current meshing workflows can be identified and patched.

VI. CONCLUSIONS

In this study, we have described a computational pipeline that uses contours from electron microscopy images as input to generate surface and volume meshes that are compatible with finite element simulations for reaction-diffusion processes. Using this pipeline, we have demonstrated the spatio-temporal dynamics of calcium influx in multiple spines along a dendrite. Future efforts will focus on the development of biologically relevant models and generation of experimentally testable hypotheses.

ACKNOWLEDGMENTS

We would like to thank Prof. Pietro De Camilli and coworkers for sharing their datasets from Wu et al. [1], Dr. Tom Bartol, Dr. Matthias Haberl, Mr. Evan Campbell, Profs. Brenda Bloodgood and Mark Ellisman for many enlightening discussions during the course of this work. CTL especially thanks Dr. John B. Moody, and Mr. Mason V. Holst for discussions on GAMer 2 code development and SWIG wrapping. We thank Dr. Tom Bartol for additional help with using Blender and the design of Blender addons. We thank Ms. Miriam Bell, Ms. Kiersten Scott, Ms. Jennifer Fromm, and Dr. Donya Ohadi for critical comments and suggestions for improving this manuscript.
