LOG(M) FINAL REPORT

SYED ABBAS AHMED

1. Background

Cells are the basic building block of life, and a normal human body can contain over 37.2 trillion cells! If you're not familiar, cells make up the human body and perform a variety of functions. A subset of all these cell are considered excitable cells. Excitable cells use chemical signals in order to communicate and execute vital functions. Examples of excitable cells include muscle cells which use chemical signals to contract, touch receptor cells which use these signals to register the feeling of touch on our skin, and neurons which use them to transport information across the body and brain. Chemical signals are really just ions moving. Ions are particles or atoms with a charge. For our purposes, all we need to know is that ions trigger specific actions when inside of a cell. Essentially, these chemical signals are regulated by the ability of cells to be able to release ions out of and uptake ions into the cell. The solid cellular membrane separates the inside of the cell from the outside. However, for these excitable cells to work they need be able to move ions through the cell membrane at will. This is where ion channels come in.

Ion channels make this movement of ions possible. An ion channel is composed of a pore forming membrane protein embedded in the cellular membrane of a cell. A pore forming protein is a protein that contains a hole which acts as a gateway for ions to move through; this protein is also able to change its shape so that the gateway can close or open. Figure 1 below demonstrates this ability of the protein to regulate the movement of ions across the cell membrane. The behavior of the ion channels in our cells dictates and allows some of the body's most crucial functions, yet researchers struggle to understand how ion channels work because of how small our cells are. I'm going to explain at a high level to a general audience how my group's project works around this limitation and is able to help improve our understanding of the ion channel by providing an alternative representation of an ion channel.

2. Introduction

Normally, researchers would be able to better understand how the ion channel works through simple experiments and observation. However, as I said before, cells and ion channels are so small that conventional observation through a microscope is not viable. Since, we can't observe a real ion

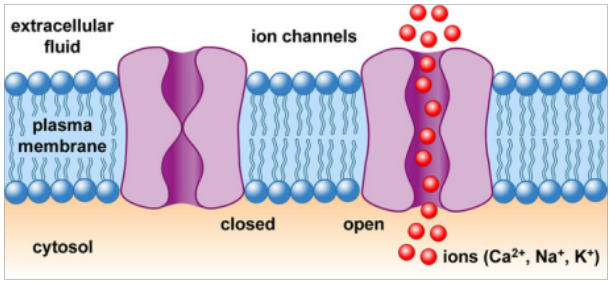

FIGURE 1. Ion Channel Example [1]

channel, a model of the ion channel is required. A model will just remove the need for a physical ion channel, and function like a real ion channel (on a computer). Modelling the ion channel is an important problem because ion channels play a crucial role in the proper functioning of a cell and as a whole the human body. Better understanding of the ion channel itself will allow us to better diagnose and treat diseases related to the malfunctioning of ion channels in the human body. My group's project was to write a software package that efficiently creates a model of an ion channels on a computer so that biologists and mathematicians can use our computer model as a placeholder for a real ion channel. The first step in constructing such a placeholder is figuring out a way to represent the physical structure of an ion channel on a computer.

Our package uses C++ [5], Python [9] a package called Tetgen [8], and a package called Fenics [6]. Some of our visualizations were generated using mesh visualization software called ParaView [4]. Our package also uses software called TMSmesh [7] to convert protein files to a usable file format.

3. Representing the Ion Channel

3.1. **Surface Mesh.** A common way of representing complex and irregular 3D objects is to generate a surface mesh for the object. A surface mesh is essentially pulling a fish net (where the holes are polygons) taut around a given object. Figure 2 below is an example of a triangular surface mesh of a dolphin. Now that I have shown you we can represent the ion channel as a surface mesh, how does this actually make things better? On first glance, a surface mesh seems more complicated than just the ion channel. However, a surface mesh is actually less complex because the ion channel itself is a smooth and continuous object, whereas the surface mesh is instead a discrete set of points with edges connecting the points. On a computer there is no easy way of storing and physically representing the ion channel, but if instead I work with the surface mesh, I can naturally store a list of all the points and a list of all the triangles that make up the mesh. Points are just represented by their x, y, and z coordinates, while the triangles are just represented by the 3 points that make up the vertices.

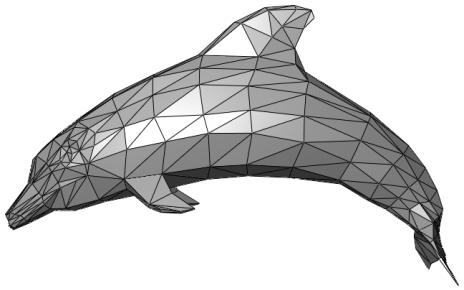

FIGURE 2. Surface Mesh Example [2]

Furthermore, we want to physically represent the ion channel so that we may 'observe' how this ion channel would react to ions. If our ion is a point in our space, and the ion channel is just composed of a bunch of points, it is quite easy to calculate the distance and relationship between two points. Representing the ion channel as a surface mesh lets us convert this very complex object and convert it into a just a list of points that is significantly easier to work with on a computer. The key thing to remember is our model should be able to approximately mimic a real ion channel.

3.2. **Volume Mesh.** A surface mesh is a two dimensional object, but our ion channel is a three dimensional object. Instead of just wrapping the object in a net, we need to build the object out of smaller pieces (think legos). To remedy this, we can extend our definition of surface mesh to be instead a volume mesh made up of tetrahedrons (i.e. the legos), which are just the extension of triangles into 3D (think malformed pyramids). Figure 3 is an example of a volume mesh. Thankfully there is existing software called Tetgen [8] that exists to convert surface meshes to volume meshes. Our package essentially wraps the ion channel components with the nets, while Tetgen [8] proceeds to fill these hollow nets with tetrahedrons.

4. Generating the Mesh

Now that I have convinced you of the usefulness of representing the ion channel as a surface mesh. What components make up the surface mesh of an ion channel, and how do we actually create them? Recall the ion channel is made up of three main components: protein, cell membrane, and the solvent. The solvent refers to the actual fluid that fills the inside of the cell, the pore through the protein, and the outside of the cell. Because we are interested in the activity of a single ion channel, all we require is a small snapshot of the cell which contains the solvent, protein, and section of a cell membrane. And we enclose this snapshot in a box mesh in order to constrict and simplify the space. Figure 4 demonstrates a cross section of this snapshot. The ability of our model to distinguish between the 3 components is crucial because ions interact differently with each component. An ion is able to move freely through the solvent, but is unable to move through the

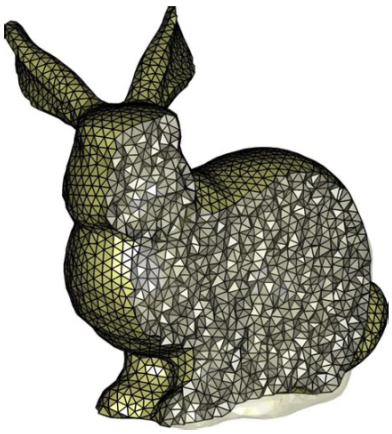

FIGURE 3. Volume Mesh Example [3]

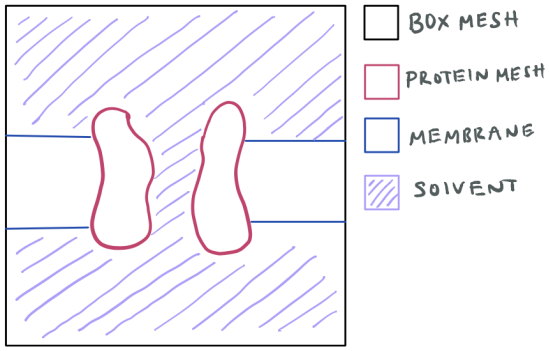

FIGURE 4. Snapshot

protein and cell membrane itself. In order for a researcher to simulate a real ion channel with out representation, they require the ability to distinguish these 3 regions.

Because proteins are highly researched, surface meshes for membrane proteins already exist. The file format of these available protein meshes is not appropriate, however, using TMSmesh [7], our package is able to convert to the appropriate file format. My package generates a surface mesh for the box, inserts the given protein mesh into the box, and creates the membrane that extends from the box mesh to the protein.

4.1. **Representing the Mesh.** Recall in order to represent the surface mesh on a computer, we just store a list of nodes for all the points in the mesh and a list of triangles for every triangle that makes up the mesh. To

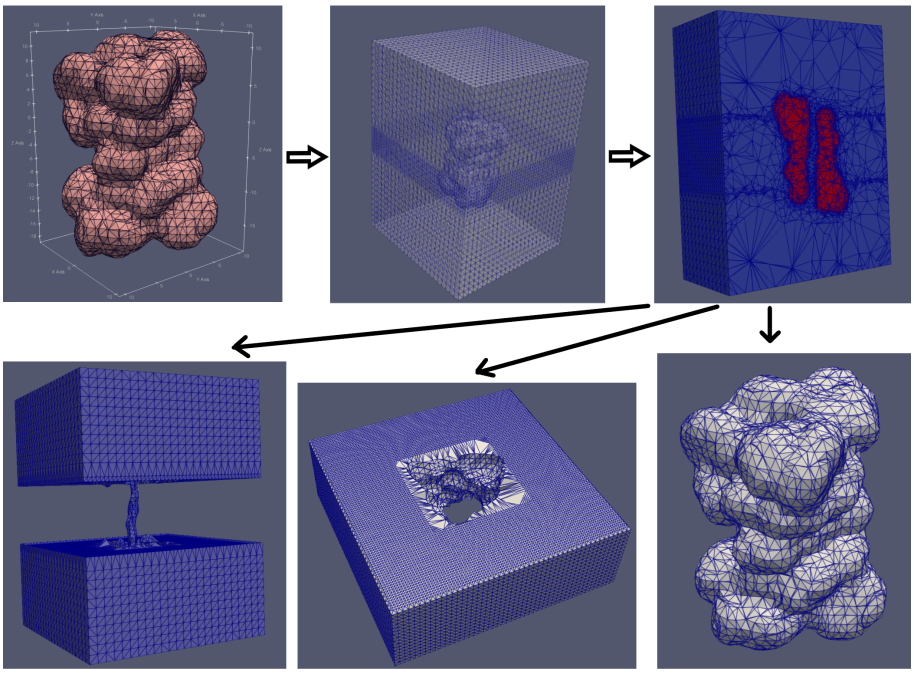

FIGURE 5. Visualization of Process

store a point, we can just store the node's x, y, and z coordinates. A triangle is just made up of its three nodes.

4.2. **Overview of Process.** Armed with an understanding of how we represent the ion channel on a computer, I want to give a high level overview of the process our package goes through in order to generate our final ion channel model. Refer to Figure 5 for a visualization of this process. Looking at the figure, we start with our given protein surface mesh. Then we generate the box surface mesh that encloses the protein. Now we take our completed surface mesh and input it into Tetgen [8] to produce a complete volume mesh (last image on first row). We then proceed to break up this volume mesh into its respective components: solvent, cell membrane, and protein. To do this we use a package called Fenics [6].

4.3. **Box Mesh.** In order to generate the surface mesh that is inputted into Tetgen [8], we first generate a nonuniform box mesh. Using the dimensions of the protein mesh, we create an appropriately sized box by generating all the nodes that go on the surface. Then, we go through and connect these nodes to form all our triangles. The method through which we do this is outside the scope of this report.

The center section of the mesh in Figure 6 is denser (i.e. more triangles) because we can think of the density of the mesh as the resolution of a

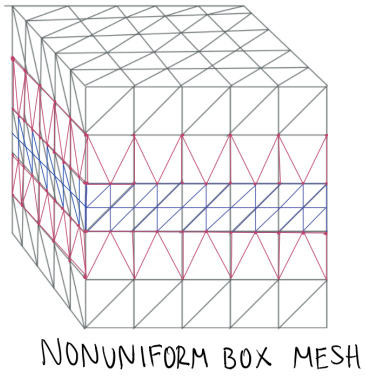

FIGURE 6. Box Mesh

picture. The higher the resolution, the clearer the picture. The center section has higher resolution because the cell membrane is at the center, and it is extremely important for researchers to be able to see this region clearly. The area at the cell membrane is most important because the protein sits right in the membrane, and we want to have a very clear image of how the ions pass through the ion channel itself. Why not make the whole box have higher resolution? Creating these meshes is computationally expensive so we have to balance efficiency with resolution, and resolution is most important at the cell membrane.

4.4. **Inserting the Protein.** With our generated nonuniform box mesh, we just insert the protein into the center of the box. Using the dimensions of the protein, we adjust the dimensions and position of our box mesh so that the protein mesh lies in the center of the box mesh. Look at Figure 7 for this intermediate result.

5. Generating Volume Mesh

Now that we have a completed box mesh enclosing our protein, we are ready to fill this hollow mesh and convert it to volume mesh. We use Tetgen [8] to convert this to a volume mesh. This results in the third image from the left in Figure 5. Looking closely, you might notice that the volume mesh has only two colors, or regions: solvent, and protein. If you recall, our final output mesh requires the separation of 3 components: protein, solvent, and cell membrane. Tetgen [8] is only able to assign 2 regions to the volume mesh because if you recall, our input to Tetgen [8] was merely a surface mesh composed of a box mesh and a protein mesh. The space in between the box mesh and the protein mesh is made up of both solvent and cell membrane. However, Tetgen [8] has no way of knowing this; it can only distinguish between the closed protein mesh and the area surrounding the protein (solvent and membrane). Thus, our final step before using Fenics [6]

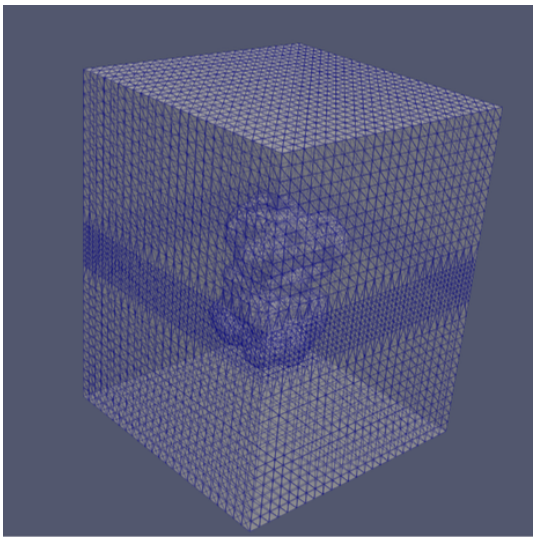

FIGURE 7. Nonuniform Box with Protein

to break up all 3 components is to extract the cell membrane region from the solvent region.

5.1. **Separation of Solvent and Membrane.** The output of Tetgen [8] is a volume mesh with 2 regions. Recall that a volume mesh is just built out of tetrahedrons (think triangular legos). The way in which regions are specified in this volume mesh (solvent, protein etc.) is just by assigning labels to each tetrahedron in the mesh. Currently the tetrahedrons of the protein are correctly labelled. All the cell membrane tetrahedrons are mislabelled as solvent. To remedy this, we look at each tetrahedron and check if it is labelled as a solvent, lies within the height of where the cell membrane would be, and is not in the pore of the protein (because this must be solvent for the ions to pass through). Using these checks we can relabel the mislabelled tetrahedrons and mark them as being a part of the cellular membrane. Now we have a complete volume mesh where each region is correctly labelled.

5.2. **Separation of Components.** Taking our complete volume with correct labels, we input this into Fenics [6], and this software uses our labelling to appropriately separate each component of the volume mesh into its own mesh. For reference, this gives us the last three images in Figure 5.

6. Conclusion

We first started with the goal of representing the ion channel as a volume mesh. We simplified this task into creating appropriate surface meshes because Tetgen [8] enables us to convert the surface mesh into a volume mesh. We first created a nonuniform box mesh to contain the protein. After combining our protein surface mesh with our nonuniform box mesh. We

inputted this final surface mesh into Tetgen [8] which generated a volume mesh of the ion channel. This mesh didn't distinguish between the solvent and membrane, so using a series of checks on the tetrahedrons (think triangular legos) in the mesh, we relabelled the solvent tetrahedrons that should be in the cell membrane. Now with a complete and appropriately labelled volume mesh, we used a package called Fenics [6] to partition our single mesh into multiple meshes based on our 3 components: solvent, cell membrane, and protein.

We were motivated to come up with a model for the ion channel because many areas of research stand to benefit from a better understanding of real ion channels in cells. However, real ion channels are too small to be studied experimentally and through observation. Our alternative represention provides a very high quality approximation of an ion channel with crucial labelling of different components so that researchers will be able to use our ion channel model to simulate how ions would interact with real ion channels.

References

[1] *Example of ion channel.*
[2] *Example of surface mesh.*
[3] *Example of volume mesh.*
[4] Geveci Berk Law Charles Ahrens, James. Paraview: An end-user tool for large data visualization, visualization handbook. 2005.
[5] ISO. *ISO/IEC 14882:1998: Programming languages — C++*. pub-ISO, pub-ISO:adr, September 1998.
[6] J. Hake A. Johansson B. Kehlet A. Logg C. Richardson J. Ring M. E. Rognes M. S. Alnaes, J. Blechta and G. N. Wells. The fenics project version 1.5. *Archive of Numerical Software*, 3, 2015.
[7] Benzhuo Lu Minxin Chen, Bin Tub. Triangulated manifold meshing method preserving molecular surface topology. *Journal of Molecular Graphics and Modelling*, 38:411, 2012.
[8] Hang Si. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. on Mathematical Software*, 11:36, 2015.
[9] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.