VISIOME: A TOOLKIT FOR VISUALISATION OF DYNAMIC BIOMEDICAL MODELS

Guy Tsafnat  
Computer Science & Engineering  
guyt@cse.unsw.edu.au  
University of New South Wales  
Sydney, New South Wales, Australia

Shaun L. Cloherty  
Graduate School of Biomedical Engineering  
s.cloherty@gsbme.unsw.edu.au  
University of New South Wales  
Sydney, New South Wales, Australia

Tim D. Lambert  
Computer Science and Engineering  
lambert@cse.unsw.edu.au  
University of New South Wales  
Sydney, New South Wales, Australia

Abstract

Standards for the storage, description and visualisation of the large datasets produced by numerical computational models in biomedicine are essential to ensure the success of international efforts such as the Physiome Project [12]. In this paper we present Visiome, a toolkit for the description and visualisation of these datasets. Visiome enables investigators to use intuitive mathematical expressions to define the shading, colour, texture and animation of an interactive 3D rendering. Visiome has been developed in conjunction with our own computational models of activation in the mammalian sinoatrial node and we show a number of visualisations of those models.

Key Words

Visiome, visualisation, model results, biomedicine

1 Introduction

Computational modeling is a powerful way of gaining insight into complex problems encountered in many disciplines. In biomedicine, the combination of increasing quantities of experimental data and decreasing cost of computing has seen a proliferation of computational models. Several large collaborative projects have already begun combining physiological models across scales from the sub-cellular to whole organs with a view to studying integrative biological function [12, 9]. The technical challenges are significant and demand new approaches to the description, distribution and implementation of such models. Researchers are working on meeting these challenges with the specification of markup languages such as CellML [7] and SBML [8]. However, managing the large numerical datasets produced by solving these models and deriving useful insight from such datasets presents new and relatively unexplored questions to the scientific community.

The international Physiome Project [12, 9] tries to integrate models into higher-level models by standardising model description languages [2, 8]. In the process many challenges arise. For example, a model developed by one researcher might be studied and reused by another researcher. This is even more likely with integrated models.

A common approach to visualisation of model results is to use an application like OpenDX [1] to visualise a particular dataset. However, this is often not suitable in biomedical modeling because model developers constantly modify the models and parameters. Normally, as models evolve, so do the results from those models; and a new visualisation would need to be created for each results set. Another common approach is to use a toolkit such as VTK [11] to create a visualisation application. Such toolkits, while very useful for the development of visualisation tools, require the investigator to be familiar with a complex programming language such as C++ and one of the above APIs to create the visualisations. Often biomedical modelers and researchers are not interested in meeting these requirements.

Numerical models rely on specific values computed for discrete spatial points and in many models these values change over time. Examples of node properties are cell membrane potential, temperature, flow velocity and various ion concentrations. Which properties are used depends on the model. The solution to a model is a set of properties for each node at each point in time. The model may take several days to solve and the solution can easily take up several gigabytes of storage. This raises challenges for sharing models and results between researchers.

2 Methods

2.1 Separation of Results from Visualisation

With very complex models, designing the model and analysing the results become more specialised and technical tasks. Most existing numerical solution packages are used to visualise and solve models. While this simplifies the modeling process, the visualisation decisions are made by the model developer who is not necessarily the person that examines and analyses the data.

We have developed Visiome, a toolkit that separates the visualisation scheme from the results and thus has two
advantages over current commercial systems. First, the same visualisation scheme can be applied to two or more sets of results from models with different parameters. Second, different visualisation schemes can be used to highlight different aspects of the same set of results. This is especially desirable for parameter-rich result sets.

2.2 Results Visualisation Language

Visiome uses familiar mathematical expressions and high-level graphical concepts to create visualisation schemes. A Visiome visualisation specification consists of two parts. The first, the View Description, defines the global viewing properties such as viewing angle and render mode. The second part, the Results Description, defines the mapping of a subset of the numerical results to a graphical representation.

In many dynamic models it is more natural to store the time a certain event occurred rather than the state at each epoch. For example, instead of storing at each epoch whether a node was active or not, one could store the time it became active. For such data we implemented special event-based functions that treat data in the results as times and not as node properties. Properties can then be deduced from that information. Figure 1 shows three event-based functions. All three functions take two parameters, \( t_1 \) and \( t_2 \). When the function is evaluated for time \( t \) it returns the value shown in the appropriate graph.

After the vertex values are computed, we construct a Delaunay triangulation \([10, 6]\) to obtain the polygons to be rendered. In order to achieve good contour visualisation, the Gouraud shading model \([5]\) is used. Gouraud shading requires surface normals to be specified at each vertex. If partial derivatives \( \frac{\partial v}{\partial z} \) and \( \frac{\partial u}{\partial z} \) are provided the vertex normals are computed from them. Otherwise, normals are approximated by averaging the normals of the polygons surrounding each vertex.

The vertex.texture property is the only vertex property that is discrete. The mapping functions return numerical values, but to apply a texture we need the name of the texture. A table maps ranges of values of the vertex.texture property to names of texture images.

3 Results

We have successfully used Visiome to provide insights to atrial and sinoatrial node (SAN) tissue models\([3, 4]\), each focusing on a different aspect of the model.

Figure 2 shows a frame from a visualisation of the AT model showing action potential propagation in a two dimensional annular sheet of atrial tissue. Figure 3 shows the Visiome expressions used to map the results to the visualisation. The results set contains the polar coordinates \((R, \theta)\) of each node and an array of properties, called Em, that contains the membrane potential at each time step. The expression

\[
<\text{map-function property="vertex.z"} >
\text{node.Em[T]}
</\text{map-function}>
\]

means that the visualisation animates the changes in membrane potential over time as changes in the Z coordinate of the corresponding vertex.

Figure 4 shows a visualisation of action potential propagation in the SAN model \([14]\). Three different textures show the three tissue types modeled. In the results set each node has a property called CellType that defines...
the tissue type modeled at the node. We mapped the value of CellType to the vertex.texture property and then mapped ranges of vertex.texture values to texture names in a texture ranges table. If the vertex.texture value \( V \) at a given node and frame meets the criterion \( \leq V < \), the texture is selected. If multiple criteria match, all the matched textures are blended together.

The Visiome toolkit allows the user to use vertex properties of X, Y, Z, colour, texture and opacity to display properties of the results set. Sometimes this is not sufficient to display all the node properties or, we might want to see the precise value of a node property. The Visiome toolkit allows the user to select one element or one node and display the node properties. In figure 5 we show part of the visualisation from figure 4 with a selected element and some chosen numerical properties. Arbitrary Visiome functions can also be evaluated and displayed.

4 Conclusion

The results sets used as examples in this paper are solutions to models described in [4, 3]. Previously we published an effective visualisation of sinoatrial node activation model [14] using a prototype of the toolkit described here. In this paper we added an expressive language that enables the construction of visualisations without the use of a programming language.

5 Future Work

There are no standard file formats for storing model results sets. This limits access to results sets and hence impedes the development of visualisation tools. We are developing such a standard as part of the IUPS Physiome Project [12]. The organisation of the data has to be intuitive and allow fast access. Additionally, we will implement a high-level library and utilities for use in a variety of programming environments.

Currently Visiome specifications are typed into a text file that is parsed and evaluated at run time. We have begun designing a user-friendly authoring tool that uses preset formulae and node properties.

Some implementations of collaborative visualisation environments have been presented [13]. The compact visualisation descriptions used in Visiome are ideal for collaborative visualisation over the Internet.
References


