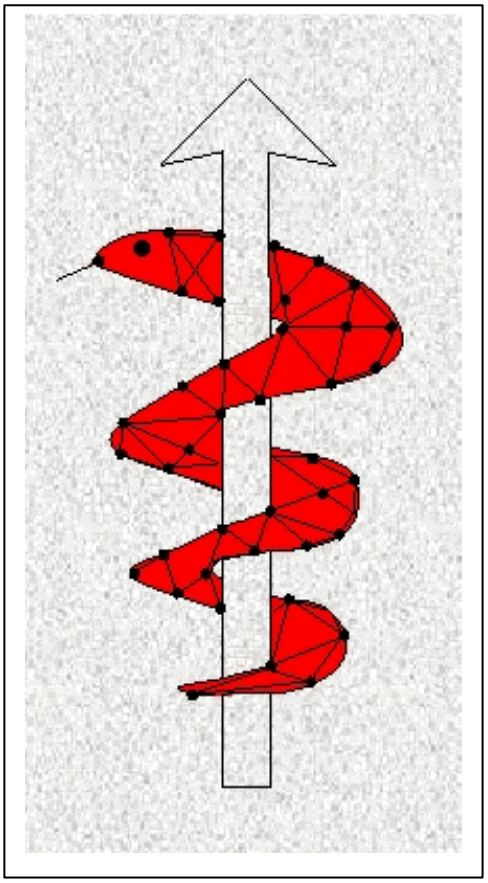
The IST Programme
Project No. 10378

SimBio

SimBio - A Generic Environment for Bio-numerical Simulation

http://www.simbio.de



Deliverable D5c
Release Notes on Final Visualisation Tool

Status:	Final
Version:	1.0
Security:	Public
Responsible:	MPI
Authoring Partners:	MPI

The SimBio Consortium :
NEC Europe Ltd. – UK
A.N.T. Software – The Netherlands
CNRS-DR18 – France
Sheffield University – UK
MPI of Cognitive Neuroscience – Germany
Biomagnetisches Zentrum Jena – Germany
ESI Group – France
Smith & Nephew - UK
U. Maribor - Slovenia
Teaching Hospital Maribor - Slovenia

The SimBio Visualisation Module Release Notes

Markus Svensén

Max-Planck Institute of Cognitive Neuroscience

1. Introduction

This document is the release notes for the final release of the visualisation module (VM) of the SimBio project [SimBio00]. The purpose of this module is, as its name suggests, to visualise the results from other SimBio modules, e.g. stress fields from bio-mechanical simulations or source localisation and influence from bio-electrical simulations.

The purpose of this document is to specify the functionality implemented in this final release, including any deviations from the functionality described in the requirement specification of the VM [Svensen00].

1.1. Overview of this Release

This release extends the basic functionality available in the first release of the VM [Svensen01]. It introduces additional and modified visualisation modes of different kinds of data (scalar, complex, vector and tensor valued data) defined on different topologies (images and geometrical objects), includes means for interacting with geometrical data, allows overlaying of geometrical objects on geometrical objects and image data on image data and offers the use of clip planes for geometrical objects.

This release provide the other partners in the SimBio consortium with means for setting parameters necessary for simulation as well as visualising the simulation results. It constitutes, together with this report, the final deliverable from SimBio workpackage five.

1.2. The Rest of this Document

The following sections provides a detailed listing of the requirements met in this final release as well as the requirements from the requirement specification of the VM [Svensen00] which are not met and discuss the reasons where appropriate. It also describes the dependencies of the VM on other software packages and the potential possibilities and limitations that result. Finally, it gives guidelines for how the VM is to be installed.

The last section outlines how the work on the Vm could be continued in order to meet requirements not met in this final, within-project release, as well as to add new, useful functionality to the Vm.

2. Release Notes

This chapter provides the detailed release notes for the final release of the SimBio visualisation module within SimBio workpackage five.

2.1. Requirements met in this Release

2.1.1. Visualisation of Voxel Volumes

Each 2D view has an associated 3D cursor position.

The VM allows linking of different 2D views, such that the 3D cursor position in one view can be mutually coupled to the 3D positions of other 2D views.

The VM is able to display 2D slices of data (images) extracted from 3D, cubic voxel volumes along any of the three principal axes (centred normals of the sides of the cubic volume).

Scalar data are displayed as pixel images using colour maps selectable for each view.

The VM is capable of scaling scalar data to employ the full range of the available colour map.

RGB data are displayed as pixel images with the corresponding colours.

The VM offers the possibility to zoom 2D images.

Scalar representations of multivariate data, such as magnitude of 3-element vectors and volume of 6-element tensors, are displayed just like scalar values.

Complex numbers are visualised by representing the real and imaginary components by different orthogonal colours (red and blue)with intensity indicating the magnitude.

It is possible to display 3-element vectors projected on the image plane. This can be combined with colour indicating the magnitude. The vectors may be subjected to a global re-scaling.

It is possible to display 6-element tensors by projecting the corresponding ellipsoidal representations on the image plane. This can be combined with colour indicating the volume. The tensors may be subjected to a global re-scaling.

Images of scalar (explicit or derived) data, RGB data, and colour representations of complex and tensor data, constitute *pixel images*.

A pixel image may be overlayed on another pixel image, provided that the number of rows and columns of one of the images are integer multiples of the numbers of rows and columns of the other image, to yield a pixel image. Note that, it is not possible to overlay a geometrical object on an image.

Image scalar data (explicit or derived, e.g. magnitude) can be used to define a display mask, excluding voxels above or below a given limit from visualisation.

Images representing 2D projections of vectors or tensors may be overlayed on pixel images, yielding compound images. Note that, it is not possible to overlay a geometrical object on an image.

The default visualisation for whole images of scalar values is a scaled grey scale map.

The default visualisation for images of 3-element vector values is the corresponding 2D projection in the image plane.

The default visualisation for images of 6-element vector values is the 2D projection of the corresponding ellipsoidal representation in the image plane.

2.1.2. Visualisation of Geometrical Objects

The VM is able to render 3D geometric objects (meshes, surfaces and volumes), with variable rotation and scale.

Scalar and RGB values associated with vertices are displayed through the colour of vertices and, where applicable, edge and surface elements. The colour of the latter will be interpolated between the colours of the vertices defining the surface element, in the scalar case using colour maps selectable for each view.

Scalar and RGB values associated with surface and volume elements are displayed through the colour of the surfaces and/or their outlines, in the scalar case using colour maps selectable for each view.

The VM is capable of scaling scalar data to employ the full range of the chosen colour map.

Scalar representations of multivariate data, such as magnitude of complex number and 3-element vectors and volume of 6-element tensors, are displayed just like scalar values.

Complex numbers associated with vertices are visualised by representing the real and imaginary components by different orthogonal colours (red and blue), specifying the colour of the model surface according to the scheme governing scalar values.

3-element vectors will be displayed directly at the vertex position or surface/volume centre in the 3D space. This can be combined with colour indicating the magnitude. The vectors may be subjected to a global re-scaling.

6-element tensors will be displayed directly at the corresponding position (see prev. para.) in the 3D space, using a 3D ellipsiodal representations, so called smarties. This can be combined with colour indicating the volume. The tensors may be subjected to a global re-scaling.

Scalar data (explicit or derived, e.g. magnitude) can be used to define an element mask, such that elements falling out of range are not displayed.

The VM offers the use of clipping planes for excluding parts of a geometrical dataset from being visualised.

Several geometrical models may be displayed in a single view. Note that, voxel volumes cannot be displayed together with geometrical models.

2.1.3. Special Data Objects

Streamlines are assumed to consists of a connected set of vertices, possibly with an associated scalar attribute representing the strength of the streamline. The are visualised as a set of connected ‘tubes’ along the edges connecting the vertices, where the strength is indicated by the thickness of the tubes.

Electrodes will be drawn as circles aligned perpendicular to the normal of the corresponding vertex.

2.1.4. Generation of animations

The VM offers the possibility to generate animations in the form of sequences of single images that may be transformed into common animation formats.

Animations can be generated from data objects with a temporal dimension, where each time step in the data will be shown in one or more frames in the animation, as selected by the user.

Animations can be generated from a sequence of changes in the viewing parameters, controlling the view of a geometric object. An example would be an animation with a rotating brain.

2.1.5. Other Functional Requirements

In addition to the key functionality described in the previous sections, the VM delivers functionality that allows the user to specify input data for other SimBio routines. It will also offer the possibility to export visualised results in two common graphic formats.

2.1.5.1. User Interaction

The user may interactively probe the value (scalar or vector valued) associated with any voxel position in a SimBio voxel volume data set.

The VM will offer the user the possibility to associate vectors representing mechanical forces or electrical dipoles, with vertices in a given geometrical data set and save these together with the data set.

The VM will offer the user the possibility to associate integer or boolean (1,0) lables with the vertices in a given geometrical data set and save these lables together with the data set.

2.1.5.2. Export Formats

The VM supports export of visualisation results in the PNG and BMP formats.

2.1.6. Non-functional Requirements

2.1.6.1. User Interface

The VM offers a graphical, window based user interface, controlled by the user through keyboard and mouse. Common commands, such as selection of data sets for visualisation, will be available through menus and toolbar buttons; other commands will be available through menus.

Commands that involve direct interaction with the data, such as zooming a region of an image or rotating a geometrical data set (e.g. a geometrical model of the brain surface) on the screen, are accessible through mouse actions. For example, clicking with a mouse button on a particular location on an image zooms the image around the selected location when the viewer is in zooming mode. Similarly, grabbing (pressing a mouse button over) a particular location on a display area of a geometrical data set and dragging it (moving the mouse while keeping the mouse button pressed) results in a corresponding rotation of the geometrical model.

2.2. Requirements not met in this Release

It is not possible to define new attributes.

There is no 3D cursor position for views containing geometrical objects.

It is not possible to link the 3D cursor position of different 3D views.

2.2.1. Visualisation of Voxel Volumes

It is not possible to link the zooming of different image views.

It is not possible to adjust brightness and contrast in the visualisation of scalar data.

It is not possible to visualise vectors as undirectional.

It is not possible to visualise the anisotropy of tensors.

Overlay images may not be transparent.

2.2.2. Visualisation of Geometrical Objects

It is not possible to visualise scalars in geometrical models using iso-lines.

It is not possible to visualise the anisotropy of tensors.

2.2.3. Special Data Objects

The Vm does not provide any special handling for dipoles. These are displayed as any other vector data.

2.2.4. Other Functional Requirements

2.2.4.1. User Interaction

It is not possible to probe values associated with geometrical models.

2.2.4.2. Export Formats

There is no integrated interface to the tool ImageMagick™.

2.2.5. Non-functional Requirements

2.2.5.1. User Interface

There is no script language interface intgrated in the Vm.

2.2.5.2. Performance

There are no means for the user to control the trade-off between quality and speed of visualisation in the Vm.

2.3. Software Dependencies

The VM depends on several software packages, without which it will not be functional. All these software packages are either available free on the Internet or available within the SimBio project.

2.3.1. The SimBio-Vista Library

The SimBio-Vista library, which is a part of the software delivered from SimBio workpackage 1, subtask 1, provides means for creating, accessing, manipulating and storing data structures representing SimBio data sets (images and geometrical objects).

The SimBio-Vista library is built on the Vista library [Vista], with data structures for representing geometrical objects added. More information about the library and its content can be obtained from [Kruggel00] as well as the man-pages delivered with the library.

The library is available to project partners via download from the SimBio web-site.

2.3.2. OpenGL

The VM uses the OpenGL® library for the visualisation. This is library that provides low-level routines for creating 2D and 3D computer graphics. It was initially developed by SGI and today its architecture is under the supervision of a dedicated Architecture Review Board, formed by several computer hard- and software companies active in the field of computer graphics.

There exist several different implementations of the OpenGL library, and the VM should work with any of these. However, to be practically useful with the data sets that can be anticipated in the SimBio project, it should be combined with an OpenGL implementation that exploits dedicated graphics hardware. This will ensure that visualisation of data will be obtained rapidly. Such hardware is today available at a moderate cost, typically in the form of graphic cards fitting standard PC computers. Although not all such cards come with Linux® drivers, the number of cards that do so are steadily increasing. Also many other computer arhitectures that are available with a UNIX® operating system (e.g. Irix®) includes dedicated, hardware supported OpenGL imlpementations.

The VM requires OpenGL version 1.2 or later.

2.3.3. Qt

Qt™ [Qt] is a library for building graphical user interfaces (GUIs) under X-windows™ and Microsoft® Windows®. It supplies the widgets (components), e.g. windows, buttons and other controls, which together form the GUI of the VM. The VM requires Qt version 2.2 or later, which includes dedicated components (windows) for OpenGL rendering.

Qt is available under different conditions on different architectures. For the X-windows system, which is the sole target windowing system for this release of the VM, there is a freely available version distributed under the GNU general public licence or the Qt public licence. There is also a commercial licence available, for the production of commercial software. For Microsoft Windows, there is only a commercial version available

The freely available version of Qt can be downloaded over the Internet.

2.3.4. QGLViewer

The QGLViewer [QGLViewer] package provide a sophisticated Qt widget for 3D OpenGL rendering. Apart from the 3D rendering area it also provides controls for rotation and zooming. Rotation can also be controlled directly by using the mouse.

The QGLViewer package is distributed under the GNU general public licence and is freely available on the Internet. The VM requires version 1.1 of the package.

2.3.5. The GLE Tubing and Extrusion Library

For the optional rendering of streamlines, the VM requires the he GLE Tubing and Extrusion Library [GLE]. Note that this is not included by default, but needs to be requested explicitly, as detailed in the README file for the VM.

2.4. Installing the VM

The VM software is supplied in a package that also includes configuration scripts designed to facilitate the installation. These scripts use tools like automake and autoconf, which are standard on most Linux systems, but which are also available under other UNIX dialects.

Using these installation scripts, assuming that the necessary tools are available, the only additional information that needs to be supplied are ways of accessing (paths to) the software package that are used by the VM, as described in previous section.

Detailed installation instructions are supplied as a part of the complete VM software package.

3. Future Releases

Future releases will at first aim for meeting all the basic requirements outlined in the requirements specification [Svensen00]. Of particular importance is the incorporation of dedicated visualisation modes for specific data objects, such as electrodes and dipoles, and the possibility to visualise several objects together.

Once all the initial requirements have been met, work will be concentrated at the inclusion of more advanced features, such as an interface to the ImageMagick library and the generation of animations. In addition, some of the fairly basic visualisation algorithms in the current release will be replaced by more advanced ditto, yielding an improved performance of the VM.

References

[SimBio00] The SimBio Consortium (2000), *SimBio: A generic environment for bio-numerical simulation*, `http://www.simbio.de/`

[Kruggel00] F. Kruggel and M. Svensén (2000), *File Format Conventions for the SimBio project*, `http://www.simbio.de/`

[Svensen00] M. Svensén and F. Kruggel (2000), *The SimBio Visualisation Module Requirements Specification*, `http://www.simbio.de/`

[Svensen01] M. Svensén, *The SimBio Visualisation Module Release Notes, v.1.0*, `http://www.simbio.de/`

[Vista] The Vista web-site: `http://www.cs.ubc.ca/nest/lci/vista/vista.html`

[OpenGL] The OpenGL web-site: `http://www.opengl.org/`

[Qt] The Qt web-site: `http://www.trolltech.com/products/qt/`

[QGLViewer] The QGL Viewer web-site: `http://www.qglviewer.de/`

[GLE] The GLE web-site: `http://linas.org/gle/`