Visualisation Tools for Semi-Conductor Modelling Software

Duncan Stevenson
CSIRO Division of Information Technology
GPO Box 664, Canberra, ACT 2601
Australia
E-mail: duncan@csis.dit.csiro.au

Abstract
This case study looks at the issues involved in using a visualisation software package to extend the scope of an existing suite of semi-conductor modelling software.

The visualisation software and its hardware platform represent the “state of the art” in powerful interactive workstation visualisation systems. The semi-conductor modelling software is the result of joint development between research groups and industry, and is at the leading edge in terms of the underlying models used and the solution techniques employed.

The result of this case study is the identification of a range of important issues to be considered when applying “off the shelf” visualisation software to a real-world scientific problem.

Introduction
In spite of the current proliferation of visualisation software packages, a vast amount of engineering and scientific data continues to be displayed with very simple techniques. This case study looked at one instance of using a visualisation software package in an environment where mature engineering software had been previously displaying quite complex spatial data in very limited manner. The purpose of this study was not to force the visualisation software solution upon the engineers working on the project, but rather to investigate the issues and problems, the costs and benefits involved if such a solution were to be attempted.

The questions being addressed were: How well do the generic features offered by the visualisation software package meet the particular needs of the application; what particular expertise is needed to implement a visualisation solution; how does one realistically assess the costs and benefits of introducing visualisation tools into a mature software environment.

The Visualisation Software AVS
The visualisation software in this case study was the Application Visualisation Software package (AVS) from Stardent Inc., running on a Stellar GS2000 workstation. AVS contains a set of applications (modules) for performing standard visualisation operations on data, a graphical interface for operating each of those modules and a graphical interface for controlling the flow of data from one module to another. The user can interact with displayed data using a supplied “geometry viewer” module which provides rotation, expansion, translation and variations in lighting conditions [1].

AVS also contains a programming environment, giving the programmer high level access to the graphical user interface, the data flows and the program control [2]. The Stellar GS2000 used here had 4 processors, vectorising capability and a 24-bit colour hardware-assisted display. The combination of AVS and the Stellar GS2000 was sufficiently powerful to let the user perform interactive visualisation tasks in real time.

The Semi-Conductor Modelling Package
The application used in this case study was the Everest semi-conductor modelling software suite developed jointly by the Rutherford Appleton Laboratory (SERC, UK) together with other academic and industrial partners in UK and Europe [3]. The software provides an environment for simulating the behaviour of semi-conductor devices, and includes modules for:
1. Setting the geometry of the device.
2. Setting the impurity doping.
3. Generating and adaptively refining the finite-volume mesh structure.
4. Solving the highly non-linear partial differential equations which govern the electrical behaviour in the device.
5. Post-processing and displaying the results of the solution.

Visualisation Requirements for Everest
Each of the five stages of the Everest suite produces a need for the user to see the data involved:
1. The geometry of the device is specified numerically, and the user needs to view the resulting device to check the correctness of the specification.
2. The user needs to confirm that the impurity doping matches the geometry of the device.
3. The software developer needs tools to view the finite-volume mesh, though the eventual aim is to isolate the applications user from the details of the mesh structure.
4. The solution of the equations for semi-conductor devices is time consuming. A visual display of partial solutions, though not currently available, would be useful in monitoring progress.

5. The final solution contains, among other data, the electric potential at each point in the device, or more correctly at each node in the finite-volume mesh. Visual display of this data, and of quantities derived from it, is the main target of the software suite.

Visualisation Limitations of Everest

The visualisation limitations of Everest stem mainly from the hardware environment in which it was developed and lack of availability of, or awareness of, modern visualisation techniques. Computing power was limited to, typically, Sun 3 series workstations. Graphical display was limited to monochrome or 8-bit colour workstations and simple colour plotters, using a 16-level rainbow sequence for colour coding. The main limitations are:

1. Lack of interactive three-dimensional display modes for displaying the three-dimensional objects (device geometry, mesh structure) of the problem.
2. Lack of control over the colour sequences used to display the resulting data sets.

Use of a Software Demonstrator

This case study produced demonstrator software to show how the usefulness of a software package such as Everest might be extended with modern visualisation techniques provided by a visualisation package.

Two attributes of such an extension were being explored. First, the extension should be cheap to implement in terms of software development effort, without disturbing existing mature production code, and second, the extension should offer the engineer benefits that are significantly better than those currently available.

The maturity of the Everest software meant that wholesale incorporation of Everest source code into locally written AVS modules was not feasible, considering the cost of design, coding and testing such an exercise. The engineers who use Everest generally want to interact with the results of the computation, and in real time, rather than interact with the process of computation. The exception is the solver module, where views of partial solutions would be of interest, but this was not attempted because of the programming overheads involved.

The decision, therefore, was to tap directly into the data flows between the Everest modules rather than disturb the existing code. The problem then became one of matching Everest data types and visualisation requirements with AVS data structures and display capabilities.

Demonstrator AVS modules were written to use Everest data sets in two places in the semi-conductor modelling sequence. First, the device geometry and mesh structures were displayed as wire-framed three-dimensional objects so that the engineer could visually inspect them as a quality control step before starting on the costly process of solving the simulation. Second, one set of results (the electric potential at each node of the mesh) was displayed as a colour-coded three-dimensional object with interactive cross-sectioning capability and interactive colour adjustment so that the engineer could inspect the solution both globally and via interactively chosen cross sections. Two types of output were produced, image-based with pixels representing the nodes in the finite element mesh and polygon-based, with polygons representing the facets of the finite element mesh.

The examples shown in the colour figures are for a simple test case, the "corner diode", where a cube of semi-conductor material sits on a larger block to form the diode. A voltage is applied across the diode via contact plates, one on the top of the smaller block, and one on the bottom of the larger block.

The particular colour sequence chosen to represent the voltages is a saturated rainbow, being the colour sequence commonly used in Everest. Figure 1 shows an isometric snapshot of the diode, taken from the AVS geometry renderer. Figures 2 and 3 show the electric potential in cross sections through the X and Z axes, with the potential shown by both height and colour. Figure 4 shows the voltages associated with the particular colours.

Figures 5 and 6 show what can go wrong when the user does not take into account the colour models upon which the visualisation software is based. They represent the minimal test case, two units wide, two units deep and two units high. The voltages were given for the 13 nodes involved, and AVS interpolated them to colour fill the surfaces of the polygons for display. The interpolation was done in HSV space, rather than along the saturated rainbow, and produced deep magentas rather than the expected yellows and greens.

Conclusions

The conclusions from this case study take the form of recommendations to a system designer who is considering using a visualisation package to enhance a proposed or existing application software package.

1. The underlying models used in the visualisation should match the physical reality of the application, and the system designer should be able to draw on expertise in the physics of the application at an early stage in the design.

This case study was a good example of a potential mismatch between model and reality. The visualisation tools relied to a large extent on local linearity of the attributes being displayed. The AVS geometry renderer applied filled polygons to convert mesh structures into solids, and the cross-section image displays were filled with square pixels. In reality, the diode semi-conductor behaviour is a surface effect, operating in a thin layer where the outer surfaces of two solids meet, and is highly non-linear. Linear representations of behaviour will be approximately correct in the body of each component of the device, but not in
the region of most interest, namely near the junction between components.

An awareness of this type of problem would allow for appropriate data and display structures to be designed at the beginning of the project.

2. The system designer should be able to draw on expertise in the field of modern visualisation techniques. Most application scientists have not had much experience of modern visualisation techniques, and are not likely to allow for their use unless assisted. This means that many existing application software packages, if not designed with visualisation of the data in mind, may not be suitable for enhancement using visualisation packages. In the Everest case, good access to the data flows between the five components was part of the original design, and the AVS demonstrator had no problem accessing that data. This was not the case within the equation solving component, and easy access to the partial solutions during the solving phase was not possible.

3. The system designer should understand, and allow for, the colour model and colour interpolation schemes used in the visualisation package.

The old mode of a colour display, with a limited set of discrete colours as a legend, breaks down with 24-bit colour visualisation like that provided by AVS. AVS interpolates colours according to an inbuilt colour space, without explicitly informing the user, and unless the chosen colour legend is linear in that colour space, intermediate colours will be generated that are not recognisable as part of the colour sequence.

Further, the viewer may well be accustomed to quantising the data, and displaying only those colours shown in a colour legend. In creating colour filled polygons for display, AVS performs some form of interpolation of the colours at the vertices of the polygons to generate the filling colours, and these filling colours are not generally shown in a discrete colour legend.

This will be even more significant at the software development stage, where, typically, very few data points are used, and the interpolation is most noticed. In the case study, a three node mesh, using a standard (AVS supplied) saturated "rainbow", interpolated from red to blue to get magenta instead of green, having selected a hue at 180 degrees away from the expected hue in HSV colour space.

4. The overall system should be properly balanced with respect to the effort involved and the desired results.

The system designer should keep in mind the end result of an exercise such as is shown in this case study, which is to provide a good display of the data at hand. Where there is value in using new display forms, such as three-dimensional rendering of a solid, and it is cost-effective in terms of programmer time, user training and configuration management, then it is worth considering. Where the existing simple display mechanisms work, such as the monochrome display of Voltage-Current curves (the "bread and butter" of semi-conductor design) in Everest, then the existing situation should not be disturbed.

5. The system designer should consider the use of a demonstrator prototype.

This case study was done with a demonstrator prototype. Simple test cases were chosen to show the users the "look and feel" of the visualisation package, and to assess the scale of difficulties involved in the particular case of this application package and visualisation software. A good appreciation was gained for the areas of greatest visualisation benefit and for the development effort needed to achieve them.

Acknowledgements

The author would like to acknowledge the hospitality of the Informatics Department of the Rutherford Appleton Laboratory (Science and Engineering Research Council, UK) who hosted him for seven months in 1990 when this work was done, and also the support given by his employer, the CSIRO Division of Information Technology, for the visit.

References

Stardent Computer Inc. (1990)

Stardent Computer Inc. (1990)

C.Greenough, D.Gunasekera, C.J.Fitzsimons, P.A.W.Mawby, M.S.Towers,
Figure 1: Isometric view of the corner diode, with surface voltage colour-coded on a saturated rainbow colour legend.

Figure 2: Voltage at points in a cross-section, parallel to the YZ plane, through the corner diode, shown both by height and colour.

Figure 3: Voltage at points in a cross-section, parallel to the XY plane, through the corner diode, shown both by height and colour.

Figure 4: Rainbow colour legend showing voltages displayed in Figures 1, 2, and 3.

Figure 5: Corner diode showing spurious colour-coding of voltages due to incorrect interpolation in the native colour space of AVS.

Figure 6: Cross-section through the diode displayed in Figure 5, again showing the spurious colour-coding of voltages due to incorrect interpolation in colour space.

(See color plates, page 433.)

Figure 1: Isometric view of the corner diode, with surface voltage colour-coded on a saturated rainbow colour legend.

Figure 2: Voltage at points in a cross-section, parallel to the YZ plane, through the corner diode, shown both by height and colour.

Figure 3: Voltage at points in a cross-section, parallel to the XY plane, through the corner diode, shown both by height and colour.

Figure 4: Rainbow colour legend showing voltages displayed in Figures 1, 2, and 3.

Figure 5: Corner diode showing spurious colour-coding of voltages due to incorrect interpolation in the native colour space of AVS.

Figure 6: Cross-section through the diode displayed in Figure 5, again showing the spurious colour-coding of voltages due to incorrect interpolation in colour space.