ANUVI - DEVELOPMENT OF IN-HOUSE SCIENTIFIC VISUALIZATION FRAMEWORK

Venkata P.P.K., S.K. Bose, Dinesh M. Sarode, Pritam P. Shete, Mohini M. Laghate and R.S. Mundada
Computer Division

Shri P.P.K. Venkata is the recipient of the DAE Young Applied Scientist / Technologist Award for the year 2011

Abstract

Rapid improvements in the hardware and supercomputing technologies are enabling users to generate Gigabytes of data over millions of grid points through simulations. Fields like Fluid dynamics, medical imaging, monte-carlo simulations require high end visualization systems to understand and the simulated processes. Though graphics hardware is growing rapidly, the software required for scientific visualization is still lagging in terms of usability, customization and compatibility. In this article we describe about the development and customization of AnuVi, an in-house scientific visualization framework, which is used by large section of the scientific community to support their diverse visualization needs.

Introduction

Though the performance of graphics hardware is growing at astronomical speed, most of the efforts are aimed at the gaming community. The scientific community is lagging behind in terms of graphics usage due to its diverse needs, which include but are not limited to customized GUI, light weight visualizer, advanced visualization system, high resolution display system, scripting interfaces to augment the available algorithms, data conversion, automated data extraction etc. It is very difficult to have a common framework catering to all the diverse requirements of the scientific community.

To cater to maximum of the scientific community requirements, we developed AnuVi as a scientific visualization framework with most of the desired features along with possibility for easier customization and made sure that it is portable on most of the environments available ranging from Windows, Linux as well as on the high resolution tiled-display system. Various aspects about AnuVi development are described in the following sub sections.

Design & Implementation

To facilitate the ease of development, maintenance etc. we have adopted Object Oriented approach for the development of AnuVi. To reduce the common errors in the design, we extensively used the Design Patterns. The usage of design patterns in the design of AnuVi is detailed by Venkata et. al.

AnuVi is developed in C++/Python hybrid language environment. All the computational and input-output intensive modules are developed in C++ for the sake of high performance where as all the extended features like interface development, scripting facility etc. are implemented in Python with help of its rapid application development features. Instead of using the low level OpenGL library for rendering the graphics primitives, high level VTK library is used for supporting both structured and unstructured grids. GUI is developed using the cross-platform WxWidgets library using its python port WxPython. CGNS is used extensively as a standard file format for data exchange.
Portability

Both for C++ and Python languages, only the standard features available across all platforms were used to reduce the portability problems. To create the compilation scripts we used cross-platform compilation assistance tool CMake to ease the development. To support the multi user development model, source code repository is maintained using the CVS. The recent methodologies like instant integration, document generation through parsing etc. are put to use. The documentation is embedded inside the source code and is generated using the Doxygen automated document generation tool.

To support the high resolution and large visual area required to render the huge datasets generated through the modern simulation software, a parallel graphics cluster with 47 million pixel resolution involving 36 LCD panels arranged in a 6x6 grid is developed and deployed in SCF building. The entire system runs on open source Scientific Linux environment. To mitigate the seams of the LCD panels which are obscuring some of the pixels, new projector based seam-less high resolution display system with 9 projectors and 7 million pixel resolution is developed and is deployed in the SCF building.

We ported AnuVi on both of the high resolution tiled-display systems using the Chromium middleware and by customizing the SPU chains required for rendering the visualization on both the tile as well as in the local node. Fig. 1 shows a snapshot of the high resolution Tsunami visualization movie generated using AnuVi over LCD tiled display. Fig. 2 shows a snapshot of the AnuVi over projector tiled display, the application being stream lines generated over a chemical reactor simulation data.

Customization & Scripting

To customize AnuVi for various applications, customization of GUI with modified properties and parameters is essential. To ease the customization, base GUI classes are developed in Python and following the Open-Close philosophy of Object Oriented methodology using the inheritance, sub-classes will be derived and the required content will be over-ridden. A sample property sheet for creating a customized GUI page with a checkbox and two textboxes is displayed below.

```python
page = ["ObjectName:","StaticText",self.name],
("ShowAxis:","CheckBox",1),
("NumberOfLabels:","TextCtrl","3"),
("LabelFormat:","StringCtrl","%-#6.3g")]
```

Providing all functionality as buttons and clickable objects is impossible. Further the diverse nature of the scientific applications always need advanced facilities, so a python shell is integrated with bindings to WxPython GUI module, VTK visualization module as well as with the AnuVi high level classes to enable user to develop short scripts for enhancing the functionality of the AnuVi. Sample code for changing the visualization to wireframe model is shown below.

```python
p = AnuviViewer.GetActor('cgns1').GetProperty()
p.SetRepresentationToWireframe()
AnuviViewer.GetRenderWindow().Render()
```
**Usage & Features**

Many users from the scientific community within and outside BARC used AnuVi to generate visualization as well as to publish their work. Fig. 3 shows an advanced visualization done using the scripting interface of AnuVi to extract multiple cut-planes each 15 degree apart from a helical heat exchanger coil spanning a total of 360° and displaying velocity contours on each of these cut-planes.

![Fig. 3: Contours drawn at various planes over a helical coil.](image)

Some of the standard features of AnuVi are listed below. **Plotting**: scalar, vector, tensor, contour, stream line, particle trace, ray casting, iso-surface etc. **Manipulation**: cut plane, clipping the dataset, extrusion, reflection, triangulation etc. **Export**: Images, movies, sessions, scripts, Print etc. **Availability**: Windows, Linux, Tiled-display environment etc.

**Conclusion**

AnuVi is tested by a large section of scientific community for their diverse scientific visualization requirements. It catered positively to many users in customizing applications, visualizing data, automating data extraction etc. By using AnuVi under the high resolution tiled-display environment, simulations can be done in large visual area with very high resolution.

**Acknowledgements**

We sincerely thank Mr. P.S. Dhekne, former associate director E&I group, BARC and Mr. A.G. Apte, former head, Computer Division, BARC for their valuable suggestions and encouragement. We also thank all the members of computer division as well as the users for their cooperation.

**References**