Data Visualization in Parallel Environment Based on the OpenGL Standard

Liberios Vokorokos, Jan Perhač, Branislav Madoš

Department of Computers and Informatics, Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
E-mail: liberios.vokorokos@tuke.sk, jan.perhac@tuke.sk,
branislav.mados@tuke.sk

Abstract: Data visualization on a computer system is a multi – discipline process. The main aim of this process is the representation of measured and / or simulated data in a form of a graphical model. This model can better help to visualize and to understand dependencies in the main frame of this data. The design and implementation of a visualization technique is dependent on wide spectrum of different factors, which have effect on the output quality. Parallel environment requires utilization of special techniques and hardware, which is specialized for this kind of tasks. Thanks to general knowledge the VRML VISUALIZER was developed.

Keywords: data visualization, parallel environment, SLI, cluster, volumetric visualization, VRML, OpenGL

1 Introduction

The visualization software VRML VISUALIZER is designed in terms of the VEGA projects No.: 1|1064|04 and 1|4071|07 on the Department of Computers and Informatics, Technical University of Košice. The design aims on creation of a multi-platform visualization software, which is optimalized for different types of parallel environments. Based on provided benchmarks it is possible to determine the effectivity rating of difficult visual algorithms execution on concrete platform compared to PC platform, which contains of a single CPU and a single GPUThis paper reflects the available 'know-how' of visualization area. It also summarize the main factors, which have dependence on visualization quality, shows possibilities how to classify or choose the right visualization technique(s), provide basic review of pararallel environments and graphic algorithms and finally determine requirements for software and hardware solutions of parallel visualization.

2 Data Visualization

2.1 Data Visualization Layout

Generally speaking, visualization is a process of mapping / transformation of computer representation of reality in the form of data, into a form of a pictures, to maximize human understanding and communication, as shown in Figure 1.

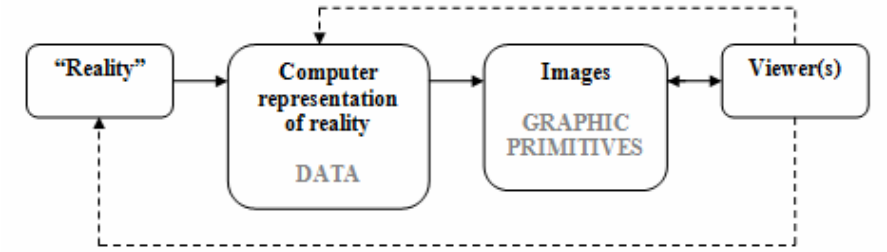


Figure 1
The visualization process

Common aspect of all visualization techniques is the series of four steps / phases of data manipulation:

- **collection and archiving of data** – modeling or simulation and basic data archiving
- **data pre-processing** – transformation of data for a better execution of visualization technique(s)
- **data visualization** - using visualization software and hardware for a visual representation of data
- **human interaction and data analysis** – taking advantage of a human perception of reality

Common modes of visualization inclusion into the scientific process:

- **movie mode** - consists of acquiring the data, producing an animation tape of the data and then analyzing the data
- **tracking mode** - consists of acquiring the data and visualizing it and observing it directly on the computer
- **interactive post-processing** - introduces user interaction in that the user is able to interactively control the visualization parameters.
- **interactive steering** - allows the user to interactively control both the actual computation of the data, e.g., by changing parameters as the computation progresses, and the visualization of the data.

2.2 Data Visualization in Parallel Environment

Because of different ways of implementation and drawbacks of logical and graphical operations during visualization process it is necessary to consider which form of parallelism will be applied in which time of data execution. There are three ways of parallel execution during visual program execution, as shown in Figure 2.

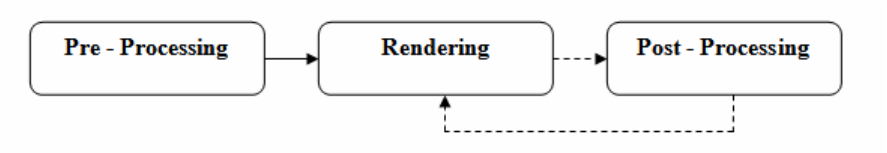


Figure 2
Application execution phases

- **Data Pre** – processing – represent the data transformation to a better execution form during computation or rendering. In this phase it is rationally to use parallelism on CPU level.
- **Rendering** – represent the visualization of received / simulated / executed data. It is possible to reach the parallelism on this level with using a computer system based on higher number of graphic adapters, which can render the output scene by blocks. The possibilities of this block – rendering management are dividen in two ways: driver solution (for example nVidia SLI, ATi CrossFire) with limited number of adapters of software solution (for example Chromium), which allows to manage the visualization on more graphic adapters installed in cluster systems.
- **Post – processing** – is the manipulation with the graphical output. Because the output is basically a sequence of image collection, the parallelism in this level can be represented by image recognition techniques, which depends on computational power of computer system.

3 Application of Multi-Dimensional Data Visualization

This chapter provides a detail design specification of a visualization software designed in terms of the VEGA projects No. 1|1064|04 and 1|4071|07 on the Department of Computers and Informatics, Technical University of Košice. During the visualization software design process it was necessary to synchronize four basic factors, which are in this proces interconnected, as shown in Figure 3.

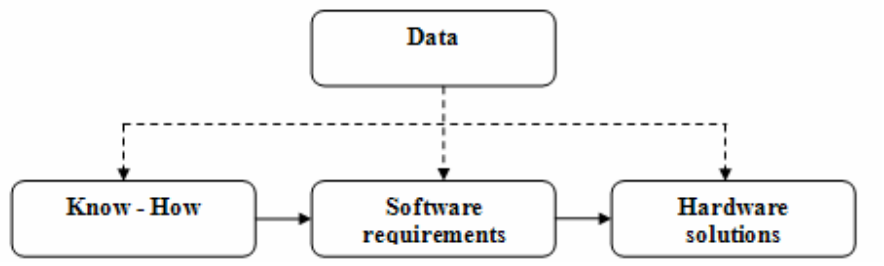


Figure 3
Visualization design primary factors

- **Data** assigned for the visualization process represent information acquired by scientific measurement in pre-processed form (transformed to a VRML format).
- **The Know – How** are in this case knowledge not only in software programming area, but also knowledge in mathematics, graphics, physics, sciences and knowledge of interactions between human and machine (computer).
- **Software requirements** for the visualization applications represent the whole decomposition of visualization problem to an elementary tasks.
- **Hardware solutions** represent all possibilities for parallel data visualization. For VRML VISUALIZER software a single CPU and a multi GPU system was established. The GPU subsystem consisted from two nVidia Geforce 6800 graphic cards interconnected into a SLI mode. This system was compared to a default system which consisted from a identical CPU but with only one identical graphic adapter.

3.1 Design of a Visualization Software

By means of a basic division of a visualization application into elementary modules it is possible to define requirements for whole visualization software, as shown in Figure 4.

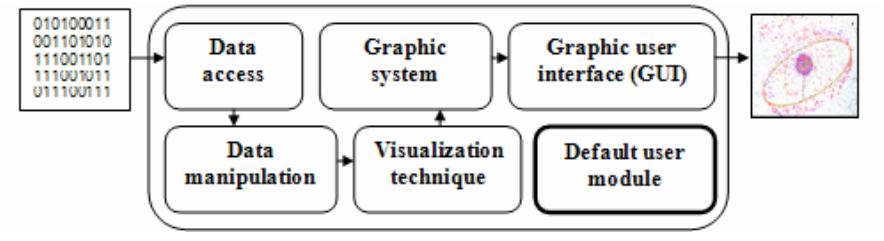


Figure 4
Visualization application as a set of basic modules

- **Whole application** – the quality and effectivity of a visualization software is dependent on the choice of a right computer language and graphic standards, in which will the application be realized.
- **Default user module** is a module by which the user controls whole system.
- **Data access module** is a module for data reading from an external storage medium into the operating system memory.
- **Data manipulation module** is a module for data-transformation-related operations (pre-processing).
- **Visualization technique module** is a module for transformation of the internal data format (raw text) to its graphical representation format which will be next used by the graphical system.
- **Graphical system module** is a module for object rendering by using primary graphical standard(s) like OpenGL. This module is also used for all virtual-reality-object-related operations.
- **Graphic user interface (GUI) module** is a module for processing of all interactions between human and a computer.

3.2 Visualization Software Implementation and Benchmarks

The VRML Visualizer, as shown on Figure 5, is a multi-platform graphical visualizer of the VRML format, based on the OpenGL graphic standard. This visualizer renders data stored in pre-processed form in an interactive passive mode. This means that it is possible to change the parameters of visualization in runtime, but it is not possible to change the parameters of visualized data.

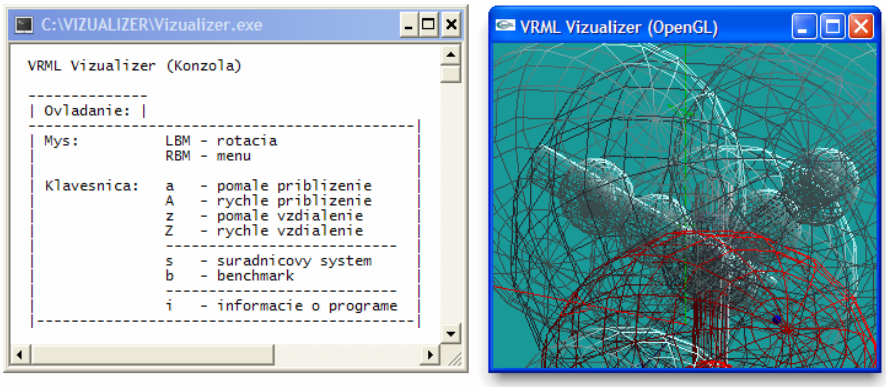


Figure 5
VRML Vizualizer

To reach the multi-platform specification for this application, the C++ language and the GLUT library was used. The GLUT library allows creating a simplified graphical application based on OpenGL standard and can additionally manage all system calls and proceed all system handlers (keyboard, mouse, …).

The application is object - oriented and can be divided into six basic modules:

- **Data access module** – is used to read data in *.wrl format from storage medium into operating memory.
- **Data manipulation module** – is a parser of a VRML format ver. 2.0. This parser also transforms objects described in *.wrl file into graphical objects, which are stored into a display list. This list is finally displayed in the graphical module.
- **Visualization technique module** – this module is based on and surface-fitting (SF) algorithm. Data are rendered in viewer-to-screen direction (front-to-back).
- **Graphical system module** – this module is based on the OpenGL standard and renders output graphic primitives of the SF visualization technique.
- **Default user module –** this module is based on the GLUT library. Its inputs are keyboard and mouse. The keyboard is used for static operations with object(s) of visualization and the mouse is used for interactive operations with object(s) of visualization.
- **Graphic User Interface (GUI) module** - the GUI is composed from two parts. First part is the graphical subsystem. This part contains from a simple menu created by the GLUT library and from a scene-rendering window. The second part is the terminal subsystem. This part has all terminal-like related controls to provide control of VRML Visualizer on administrator-level.

Other program characteristics:

- **Interactive application parts** – the interactivity consists of object(s) rotation, lighting, shadowing, wireframe display of an object.
- **Benchmark** – the application is suited for system(s) benchmarks.

The benchmark module for this application is a simple implementation of basic rotating graphical algorithm. This algorithm forces the object to change all its points by rotating whole object over the ‘y’ axis two times. This module counts the time and number of frames needed for this operation. The output of the benchmark module is the average number of frames per second (FPS).

Test Systems:

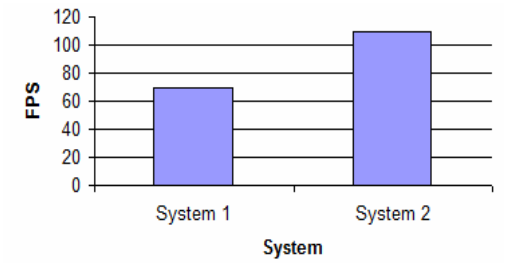
System 1. AMD Athlon 64 3700+, 2GB RAM, GeForce 6800GS 256 MB DDR3

System 2. AMD Athlon 64 3700+, 2GB RAM, SLi GeForce 6800GS 256 MB DDR3

Benchmark Results:

System:	1) GeForce 6800GS	2) SLi GeForce 6800GS
FPS:	70	110

Table 1
Benchmark results



Graph 1
Benchmark results

Conclusion

The main goal for design of visualization software was to analyze all data visualization related information and to apply this information into a parallel environment. The main goal for the VRML Visualizer application was to demonstrate the main model or concept of a visualization application. Every application of this kind has to read, transform and to visualize (render) data on screen. There are many possibilities to do these operations.

The result of benchmark results (Table 1, Graph 1) is, that VRML Visualizer application provides better results (higher FPS) in parallel environment.

References

[1] McCormick, B. H.: Visualization in Scientific Computing, Computer Graphics, 21(6), 1987

[2] G. Scott Owen: Definitions and Rationale for Visualization. Last modified on February 11, 1999 Dostupné na internete: <http://www.siggraph.org/education/materials/HyperVis/visgoals/visgoal2.htm>

[3] Brodlie K. W., Carpenter L. A., Earnshaw R. A. ,Gallop J. R., Hubbold R. J., Mumford A. M.,Osland C. D., Quarendon P.: Scientific iii - Visualization - Techniques and Applications, 1992

[4] Chromium: Parallel, Distributed OpenGL Rendering On Commodity Clusters Dostupné na internete: <http://chromium.sourceforge.net/doc/LLNLcopy.html>

[5] Vokorokos, L., Jelšina, M.: Počítače: základy technických prostriedkov, Mercury – Smékal, s.r.o., Košice, 2004, ISBN 80-89061-90-7

[6] McReynolds, T., Blythe D.: Advanced Graphics Programming Using OpenGL, First Edition, Morgan kaufman Publishers – Elseivier Inc., San Francisco 2005, ISBN 1-55860-659-9

[7] Vokorokos, L.: Digital Computer Principles, 1.vydanie, Typotex Publish House, Budapešť, 2004, ISBN 9639548 09 X

[8] Jirka T.: Multidimensional Data Visualization, Technical Report No. DCSE/TR-2003-03, March, 2003

[9] Hudák S.: Vizualizácia dát z konfokálneho mikroskopu. Prehľad problematiky. 29. júna 2005, Dostupné na internete: <http://www.realtime.sk/~skip/dipl/dokumenty/prehlad.pdf>

Supported by VEGA project No. 1|4071|07