REAL-TIME RENDERING IN A PC-CLUSTER ENVIRONMENT PROVIDED BY OPENSG

A. Bueschenfeld
Fachhochschule Bielefeld, University of Applied Sciences - Faculty of Architecture and Civil Engineering, Artilleriestr. 9, D-32427 Minden - E-Mail: arne.bueschenfeld.de

KEY WORDS: 3D, Interactive, Real-time, Virtual Reality, Visualization

ABSTRACT

OpenSG is an open source scene graph system. It works under Linux, IRIX and Windows based computers. The OpenSG project was initiated in 1999. The program is written in the compiler language C++ and undergoes ongoing development. OpenSG features real-time rendering and supports multi-threading as well as clustering in a user friendly way. Furthermore the scene graph system offers a base of operations for active and passive stereo-applications.
In addition to Open GL, which can merely use model data as vectors, a scene graph offers the possibility to save scene data hierarchically in a tree structure which refers the dependency of the objects. In case of multiple occurrences of the same detail it only needs to be modelled once as it can be transformed various times later in the displayed scene, e.g. a recurring window shape of a building.
A further advantage offered by the scene graph solution is its ability to display objects very fast. This is achieved by calculating exactly the section of the total scene, that is in the observer's field of vision while those parts of the whole projection which are hidden by obstructing objects are not rendered. For instance, the scene graph system does not render the back of a building as along as the observer is looking at its front. An additional feature is the so called backface-culling, a procedure which excludes all those parts of the scene that are not in the viewer's angle of vision.
Moreover OpenSG enables the depiction of a rendered object according to the viewers distance in various levels of detail (LOD) in order to reduce the rendering time.
All together SGS are able to visualize fairly large and complex amounts of data with an outstanding performance.
This contribution aims at demonstrating the realisation of OpenSG as presentation-environment for object visualisation using multiprojection on a power wall. In this context a simple-scene-manager is implemented in OpenSG in order to set up the viewport and to permit the user's interaction with the scene. This is conducted with commonly used OpenGL capable graphic boards in a local network setup.
This contribution focuses on applications in the field of architecture and preservation of historical buildings as demonstrated in the following examples of a gothic facade and buildings of the classicism of the 19th century.

1. INTRODUCTION

This contribution tries to show the possibilities of visualizing virtual models with clusters. For this purpose the API OpenSG is presented, which allows to display real-time rendering graphics on single or several screens by using clusters. Moreover the chance to generate stereo-projections will be discussed.
There will be an introduction into the basics of scenegraph-applications. In addition to this, the paper tries to give an overall view of OpenSG functions.

2. OPENSG

The idea of OpenSG is to create an open source real-time rendering API which based on OpenGL works independently of the system to guarantee the requirements of today's VR-systems. API is the abbreviation for Application Programming Interface. The API provides routines, protocols and service programs for software productions. The foundations of OpenSG were built in 1999 by Fraunhofer-IGD in cooperation with SGI and other international experts. Initiators were Dirk Reiners, Allen Bierbaum und Kent Watsen. Declared goals that should be realized by OpenSG as scenegraph-system were:

1) Portability between several operating systems
2) Multi-threading support
3) Support of multiple-graphics-pipes and Clusters
4) Expandable and self-reflective system (The skill to provide information about itself – realized by Field Containers)
5) Flexibility

2.1 Features of OpenSG

OpenSG works on IRIX, Linux and Windows-based computers. The Open-Source-code permits experienced users to generate expansions and changes of the system. Furthermore OpenSG is freeware. OpenSG works as scenegraph-system. It supports multi-threading among other things as well as clustering and is able to support active and passive stereo-applications. The system is based on OpenGL-standard and uses for example GLUT to manage windows and generate graphics. OpenSG is capable to import different kinds of file-format, and provides two output-formats to export created scenegraphs. The cluster-feature of OpenSG allows to synchronize different computers and to embed them into the same program-process. This connection of computers is called a cluster. For example, the cluster mode permits to display stereo-pictures fast or serves to run a powerwall or to display a 3-dimensional projection inside a CAVE.

2.2 The structure of OpenSG-systems

Based on one of the following three operating systems Linux, IRIX or Windows, all OpenSG-systems are using the OpenGL-graphic-library. The OpenGL-standard is system-independent and was developed by SGI, which took also part in inventing the OpenSG system. Based on OpenGL the Graphic Library Utility Tool GLUT is also integrated and may be used in OpenSG. GLUT is an OpenGL-wrapper and simplifies window-management and basic OpenGL functions. The low-level functions of OpenSG are constructed on these basic-classes. For example the low-level functions include the system-

architecture, multithreading- and clustering- as well as rendering-functions. Further more the state-handling is defined here which enables the minimization of changes of the state of the chip and to raise the system-performance. On top of the low-level functions there are three high-level functions:
First, the large scene support, which enables the system to present large and complex scenes at the best speed by avoiding unnecessary rendering-processes (Keywords are for example occlusion-culling, backface-culling and LOD).
Second, the high-level primitives which among other things produce view-dependent Levels of Detail, NURBS and morphing and the visualization of scientific data and allow volumetric effects like clouds or fire in OpenSG scenes.
Third, the high-level shading, which aims at improving rendering-qualities by taking advantage of modern hardware to calculate light effects.

3. SCENEGRAPHS

3.1 The Scenegraph

The visualization of larger objects in pixel- or vector-data causes a huge amount of data which only a few systems are able to display fast and with good quality. According to this it is necessary to implement static light sources during the creation of the file, and that all data of the scene are static. Scenegraphs facilitate to store data hierarchically. Information about visualization of the surfaces or light effects, e.g. shadows, are calculated in dependence of the users point of view. OpenSG deals with this in real-time while displaying the scene. Scenegraphs posses knowledge about the stored data, literally they get knowledge about the whole scene they are about to render. This makes it possible to safe calculating time and to minimize state-changes: If 100 objects in green colour and 100 objects in red colour have to be drawn in a mixed order, OpenSG is able to sort these objects depending on their colour. For example the green objects will be drawn before the red ones. This economises the time necessary for the state changes.
The advantages of displaying a graphic with scenegraphs are a better performance and as well as a lower amount of data and multiple possibilities of usage. Thus it is not necessary for example to define a reoccurring geometrical shape more then once as a Scenegraph is able to transform that shape several times depending on its number of occurrence. The knowledge about the interdependences between single objects enables the scenegraph to realize which objects are in the field of view and which are hidden by other objects or are out of range of the shown display. The exclusive calculation of forms shown in the display saves time and raises graphic-speed.

3.2 The tree-structure

In order to construct a tree nodes and cores are used. While nodes represent the structure and the dependencies of the single parts inside of the tree, so to speak the position inside it, the cores define the content of the node, like geometrical forms or surface information. The top of the tree is called root or some times even world. This root node cannot be subordinated to any other node. Higher nodes are called parents, subordinated nodes are called children. The tree-structures are divided into single-parent- and multi-parent-systems. Multi-parent-systems are able to assign several parents to one node. The advantage of a multi-parent-system is that nodes which contain a reoccurring geometrical shape have to be defined only once. Then several transformation nodes are assigned as children of the geometrical node. Single-parent-systems like OpenSG are merely able to provide one parent per node. However, it is possible to assign any number of children whereby changes of the parent are passed down to all their children and grandchildren and so on. In this way physical dependencies can be simulated. If all nodes in a tree have no more than two children this is called a binary-tree.

3.3 Hidden Face Removal - Culling functions

One of the strong points of scengraph-systems is the ability to cut out unnecessary rendering processes. For this reason there have been invented several functions.
OpenSG uses backface-culling, to cut out the away turned sides of objects from the observers view. These areas are not rendered so that calculating-time is saved and the efficiency of presentation is increased.
Besides to backface- there is also occlusion-culling. In this procedure the scenegraph-system identifies interruptions in the line of view between observer and displayed object. Is the object fully or partly hidden from the viewer's eye the scenegraph system does not render the invisible parts.
Moreover, view frustum culling cuts out all objects or parts of objects of the rendering-process that are not in the observer's field of view. The field of view is assumed as a frustum of pyramid. This can affect significant savings of time, depending of the size of the scene.
Small feature culling is used by some scenegraph APIs but is not implemented in OpenSG. When SFC is provided, small objects are not rendered if they do not exceed a defined size in the displayed screen. Hence, very small objects that are not noticed in the visualization are not calculated.
Similar to the small feature culling the classification into several levels of details is used. A LOD, a Level of Detail, defines the accuracy of the rendering of an object. In most cases the LODs are defined in dependence on the distance between viewer and the observed object. When the viewer gets close to the rendered object a higher Level of Detail is activated, because the viewer is able to recognize smaller details in the picture. When the viewer steps away from an observed object the Level of Detail is lowered. Accordingly, details which do not need to be realized in the distance can be culled out. There has to be provided a separate model for each shown LOD. Here the fact that lower Level of Details contain a smaller amount of rendering-objects leads to a significant reduction of calculating-time. If the synchronization of the distance with the shown detail is well chosen the observer will not realize the changes between the particular LODs. Some scenegraph-systems use morphing-methods to interpolate between different LODs. Thus, they achieve smoother model changes which the observer can hardly recognize.

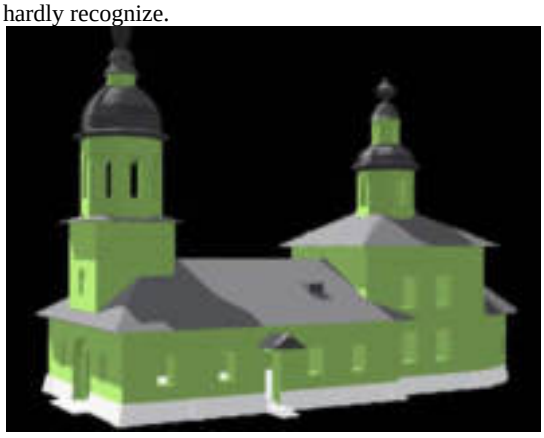

Figure 1. Low Level of Detail

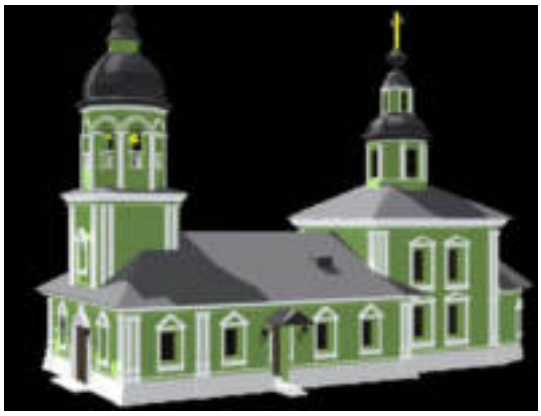

Figure 2. High Level of Detail

3.4 Field Container

One major feature that distinguishes OpenSG from other scenegraph-systems is its usage of Field Containers. The Field Container concept was invented by the need for assigning objects of multi-threading applications exclusively to a single user. This procedure prevents two calculating-processes to alternate the same data-object at the same time which frequently causes a system-crash. Almost every class in OpenSG, so are nodes and cores, are produced out of Field Containers. Objects produced with the assistance of Field Containers are accessed by pointers with a "Ptr" suffix. For this reason nodes and cores are accessed with pointers too. Field Containers are able to provide information about themselves or, as the case may be, the data stored inside of them, e.g. they are self-reflective. The ability to reflect was one of the ambitions of OpenSG. Every class which is produced by Field Containers is declared in the following format:

Static::create

Static describes the kind of container, for example transformation, geometry, node etc. Create is the command to create the container. The command to create a node named Rootnode is:

NodePtr Rootnode = Node::create();

The type of the pointer has to be conform to the type of the created object. The creation of the pointer and the assignment to the created node can be accomplished in two separate steps. In this case both was condensed to one in order to shorten the program-code. Field Container store data in single- and multifields. For example, the children of a node are stored in a multifield because it is possible to assign more than one child per node. This field is called childrenfield. The parent of a node is stored inside of a singlefield because there cannot be more than one parent for each node. Similar to arrays in C++ Multifields have an index to be accessed too. These indexes are, as common in C++, defined in brackets.

4. OPENSG- CLUSTERS

A computer cluster is a group of locally connected computers that work together as one unit. OpenSG divides a computer-cluster in one client and several servers. This contradicts the common perception of a client-server-setup but is explained by the OpenSG-philosophy that servers are serving the screen while the client performs the task of being the interface between cluster and user and providing the model-data. A cluster can be used to operate a powerwall or to realize a CAVE or to generate stereo-projections. Powerwalls are clusters of several screens which are projected from behind like TFT-Displays or backprojectionscreens, to build a large formatted imaging area. OpenSG provides a fairly simple possibility to realize a powerwall that is rectangularly formatted (a matrix-formed arrangement of screens). A Cave Automatic Virtual Environment (better known by the recursive acronym CAVE) is an immersive virtual reality environment where projectors are directed to four, five or six of the walls of a room-sized cube. The name is also a reference to "The Simile of the Cave" in Plato's Republic where a philosopher contemplates perception, reality and illusion. A lifelike visual display is created by projectors positioned outside of the cube and controlled by physical movements from a user inside the CAVE. The first CAVE was developed in the Electronic Visualization Lab at University of Illinois and was announced and demonstrated at the 1992 SIGGRAPH. It has been used and developed in cooperation with the NCSA, to conduct research in various virtual reality and scientific visualization fields.

Figure 3. OpenSG-cluster contending one client and three servers

In this kind of visualization-technique each server controls one of the output devices e.g. a TFT-display. The client provides model-data, handles the synchronization of calculating-processes and allows user-navigation inside of the scene. Each server renders only those parts of the whole scene which is displayed in the servers field of view.
OpenSGs clustering-ability is very similar to the multi-threading functions because clustering feeds the calculating processes partially to several computers. For clusters the type of computer does not matter. It is possible to build a multimedia-cluster with common customary computers. Even operating systems, CPUs and graphic cards do not have to be of the same type. Nevertheless, a network connection between client and all servers is recommended. Notionally it is possible to initialize more than one server on one computer or even to start a client and a server on the same computer. However, this is contradicting to the cluster-philosophy which is to take advantage of the performance of multiple computers in order to reach the best possible result in calculating-power. To operate with just one computer for server and client may be useful to test a cluster-program.
This is the schemata of clusters consisting one client which is controlling two servers that serve four projectors:

5. STEREO-APPLICATIONS WITH OPENSG

Stereo-applications with OpenSG need a cluster to serve at least two projectors which are swapping two pictures on the same screen. To achieve this both projectors have to be adjusted to use exactly the same projection-screen. In this case Projector-racks with adjustable mountings are the best choice. OpenSG supports stereo-applications with polarised filters in an easy way. The OpenSG tutorial contains one of these applications. This is a passive stereo-technique that uses filters in front of the beamer-lenses. The viewer wears filter-glasses and each glass corresponds to one of the beamer-filters. In this way each eye sees just the part that is projected by the corresponding beamer. OpenSG supports StereoCameraDecorators. Using this decorator it is possible to use just one camera position and assign two different viewports to this camera. These viewports form the two pictures necessary for stereo-viewing. Each viewport represents the field of view of one eye. It's necessary to input the eye-separation of the viewer. The larger the eye separation the better the stereo-effect is realized. Too large eye-separations cause the impression of a strong cross-eye effect.
Besides the polarised filters there is the anaglyphic-illustration. This technique applies to the following: The polarised filters are replaced by red-green illustration of both pictures. The viewer wears glasses of the same colour. This is a simple stereo-technique that does not need clusters or projectors because both pictures are displayed on the same screen. The superposition is done by the computer.
The DAVE of the Universität Braunschweig uses OpenSG and shutter-glasses to realize 3D-visualization. Shutter-technique is an active stereo-method and recommends high programming skills and large technical equipment. In this method glasses are worn that are shutting both eyes alternating, so there's just one eye looking on the screen at the same time. The projection changes in the same frequency as the shutter glasses so that both eyes are looking on different pictures while the picture remains the same for the single eye. This kind of 3D projection delivers very realistic results. The grade of realism of the visualization is referred to in "level of immersion". Immersion describes the effect which the viewer realizes when the displayed scene gets more and more real. The more realistic the displayed scene the higher is the level of immersion. To identify the correct perspective onto the scene a sensor is placed at the back of the head of the viewer that detects the position inside of the CAVE and the perspective. A mouse like pointing device serves as navigation interface which transmits position and movement to a receiver. Like a mouse this 3D-navigator offers several buttons to activate the movement of the avatar or the rotation around several axes.

6. CONCLUSIONS

OpenSG provides much potential to visualize large scenes in different ways. For architectural purposes this processes are more and more wanted. The advantages of OpenSG are low costs for soft- and hardware, an open system structure, and miscellaneous functions. Prepared stereo-applications and high performance are further benefits. Contra arguments are incomplete and rather brief documentations. This complicates the access to program own applications. Interested users should work through the OpenSG-Tutorial written by Oliver Abert and use the OpenSG-forum for further questions. Furthermore programming-examples may at times provide help with unanswered questions. Scenegraph-systems are state of the art at moment and provide best results in this field. Therefore it is very likely that scenegraphs gain more and more influence for visualization-purposes. OpenSG supports trendsetting techniques and is permanently updated. OpenSG is freeware and provides a complete scenegraph-system with good expansibility.

REFERENCES

OpenSG Homepage

http://www.opensg.org

Pomaska Günter Between photo-realism and non-photo realistic rendering - modeling urban areas for real time VR International Workshop on Vision Techniques applied to the Rehabilitation of City Centres, Oktober 2004, Lisbon, Portugal

Pomaska, Günter Implementation of web 3D tools for creating interactive walkthrough environments from building documentations ISPRS WG V/4 and IC WG III International Workshop on Vision Techniques for Digital Architectural and Archaeological Archives 2003, Ancona, Italy

Büschenfeld, Arne

Installation einer OpenSG Entwicklungsumgebung und Transformation eines komplexen Cad-Modells in eine Scenegraph-Datenstruktur

Fachhochschule Bielefeld, 2005, Minden, Germany