A Virtual Laboratory for Computer Graphics Education

L. W. Pettersson

Department of Information Technology, Uppsala University, Uppsala, Sweden

N. Jensen

Learning Lab Lower Saxony, Hanover, Germany

S. Seipel

Department of Information Technology, Uppsala University, Uppsala, Sweden
Department of Mathematics Natural and Computer Sciences, Gävle University, Sweden

Abstract

We specify a 3D network-distributed virtual environment (DVE) where students learn computer graphics programming. The client/server software consists of a programming environment, the 3D DVE that is used to develop dynamic link libraries (DLLs) to generate 3D graphics. The software distributes a DLL to remote clients in an automatic way, and allows students to view the output of the DLL together in the interactive 3D DVE. A field study will evaluate the use, usability, and usefulness of the 3D DVE by an analysis of the discourse between students. We hypothesize that the software supports autonomous and collaborative learning of computer graphics principles and are valuable for readers that teach 3D computer graphics to students at remote universities.

Categories and Subject Descriptors (according to ACM CCS): I.3.7, K.3.1 [Computer Graphics, Computers and Education]: Virtual Reality, Computer Uses in Education

1. Introduction

Distance learning becomes more important[16]. Students scattered geographically access data and tools that are expensive to replicate from remotely located educational institutions. Virtual universities[2] encourage students to participate in their studies from home, at work and from other education institutions than their primary one. Also for pedagogical reasons, the value of collaborative work for learners in small teams becomes evident[6]. Further, collaborative work eases teachers work effort because students learn in self-directed, autonomous ways[7].

The paper specifies a taylored 3D DVE that resembles a virtual laboratory for teaching 3D computer graphics programming at universities. See the web page on the VASE plugin framework[15] for an overview of the base technology. The 3D DVE introduces new aspects to computer graphics education (CGE) by the integration of formerly decoupled applications for group-learning in a homogenous and flexible to rearrange 3D user interface. With the virtual laboratory we aim to provide a learning environment which is as good as a traditional setup where students cooperate in pairs and share the same computer. The virtual laboratory is specified in the paper.

The virtual laboratory helps students to learn OpenGL programming by use of the following features: (see Figure 1):

- immediate distribution and execution of students' OpenGL code,
- shared viewing of the result,
- scaling, rotating and translating the result,
- viewing and editing the same code through any compiler creating a Win32 DLL and
- peer to peer learning by means of integrated chat tools and computer-supported collaborative learning (CSCL).

Previous work in the area of educational virtual learning environments is discussed in Section 2. The design and implementation of the virtual laboratory is specified in Section 3. Section 4 describes a preliminary test setup used for

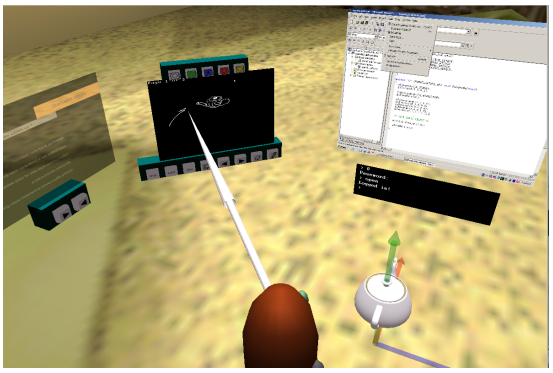

Figure 1: *Avatar and collaboration tools, presented clockwise starting from left: slide show tool, whiteboard, VNC, teapot rendered with OpenGL and an Avatar.*

preparing a field study where students performance in the virtual laboratory will be compared to students collaborating in a traditional computer laboratory. Section 5 discusses the results. A conclusion is given in Section 6.

2. Previous and Related Work

2.1. The Relevance of 3D Programming for Computer Graphics Education

Several authors motivate the use of 3D graphics libraries to start teaching computer graphics programming because 3D graphics appeals students due to it's ease of use, flexibility, and power of expression[13, 14]. 3D graphics can be used to teach 2D graphics as a special case. Critics argue that the approach supports the use of computer graphics, but not the apprehension of its underlying principles[5]. However, field studies indicate that 3D computer graphics programming supports an understanding of geometry and rendering principles once students have become attracted and familiar with the programming environment[1]. This does not mean that programming should be the only way to teach computer graphics principles[12], but it is an essential part of the curriculum. Therefore, we must use tools to teach 3D graphics programming.

OpenGL is not the only mean to teach 3D computer graphics programming[9, 19]. We use it because OpenGL is what many students will use in practice. Moreover, it is flexible to use, widely disseminated, and platform-independent. Compare for example [1] for further discussion.

2.2. Tools for Computer Graphics Education

Systems that are comparable to our work but have 2D GUIs are available [3, 6, 10] (see [18, 8] for potential usability shortcomings in 2D compared to 3D GUIs and between different metaphors of use for interfaces). They indicate that several communication media should be integrated to enhance the learning experience for users.

Our system is most comparable to MAVERIK[11] but with a focus on small learning environments and moderately populated, synchronous learning groups (ca. 2–10). The major difference is that our system components can be re-arranged in an easy way by the specification of component instance names and event flows between them in an XML file. For example to add an extra VNC window to the environment the component tag for the VNC in the XML configuration file should only be duplicated and repositioned as shown below:

```
<!- original VNC plugin ->
 <component type="vpiVNC" name="vnc">
  <terminal>login</terminal>
  <translate>2 1.3 0</translate>
  <rotate>0 -90 0</rotate>
 </component>
<!- added VNC plugin ->
 <component type="vpiVNC" name="vncextra">
  <terminal>login</terminal>
  <translate>2 -1.3 0</translate>
  <rotate>0 90 0</rotate>
 </component>
```

Churchill et al. survey[4] other relevant systems, for example MASSIVE-3. Those systems either do not support computer graphics teaching per se, are much more complex to learn and customize, or do not integrate arbitrary tools seamlessly in an immersive 3D environment.

3. A Distributed Virtual Laboratory

3.1. Design Objectives

CGE becomes more relevant because visualization has been recognized as *the* means to make complex data understandable to human beings[17]. However, the computer science curriculum needs revision to take into account the change of available technology and educational environments in recent time – 3D graphic workstations and networks are widely disseminated. What once was designed to adhere to technical constraints may need revision. For example, students can be expected to use networked computers with affordable 3D graphics cards, even at home.
Another issue is the way how students should learn – collaboration and peer-programming are examples of two modes of interaction that have attracted considerable interest from researchers and industry. A CGE tool must support students to prepare and act out authentic work patterns, for example as specified in eXtreme Programming[20].

When computer graphics is taught today it is more often with a focus on 3D graphics than 2D graphics[1, 5], it is therefore natural to provide a seamless environment in which 3D

computer graphics can be developed and investigated. The distributed virtual laboratory helps remote students to learn together by rapid prototyping and eXtreme programming[20]. In contrast to other software (stand-alone instances of Virtual Network Computer[21] (VNC)) students make use of shared viewpoints and also different viewpoints to investigate the result in 3D. Students develop programs that generate three dimensional graphics in the distributed virtual laboratory by use of OpenGL.

3.2. Base System

Our system helps students to rapidly prototype an OpenGL application, build their application on a server, and automatically run the application on their workstations with interactive graphical feedback. The virtual laboratory for CGE is based on a scene graph library (VRT), a shared state repository for network communication[22] (Streep) and an environment for visualization and simulation (VASE). VASE is a framework for modular development of networked virtual environments. The framework is configured using an XML document describing which components should be loaded and, specifying their initial position and configuration in the DVE.

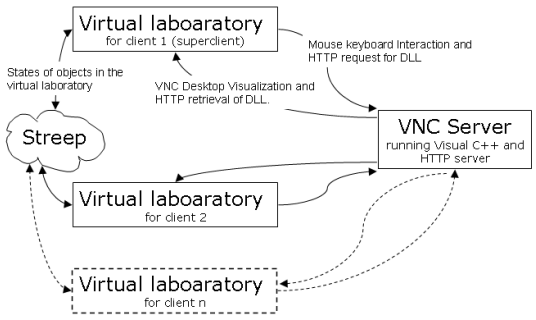

Figure 2: *Design and data flow in the virtual laboratory.*

The main idea is to make it easy to develop and integrate new plugins in the DVE and to enable the reuse of plugins developed by other parties. The VASE framework, see Figure 3, designed for creating virtual teaching environments includes many plugins that support collaboration. The virtual laboratory reuses plugins such as the whiteboard plugin for realtime sharing of handwritten notes and the VNC plugin for sharing a computer desktop. Real time conference voice communication is an important collaboration tool which has not been implemented as a plugin. We have found that both Microsoft Game Voice† and SpeakFreely‡ are easy to setup and work without noticeable latency for communication in small teams (2-4). To distribute OpenGL code to

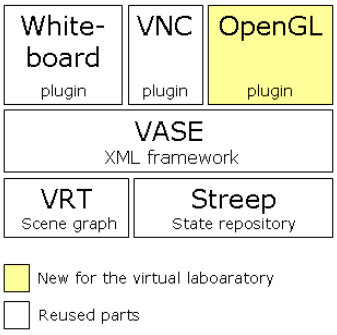

Figure 3: *Structural overview of the virtual laboratory.*

connected clients that is compiled on the VNC server, the VASE framework is extended by the OpenGL plugin, see Figure 2 and 3. The OpenGL plugin uses Hypertext Transfer Protocol (HTTP) to request and download the DLL from the VNC server. The DLL is reloaded into memory and the OpenGL code is rendered. To make this transparent for the user a Common Gateway Interface (CGI) script is called periodically. The CGI script returns the DLL only if the creation time of the DLL is greater than the recorded time of the last download.

3.3. Collaboration in the Distributed Virtual Laboratory

Users configure VASE to adapt and change the design of the virtual laboratory. The design of the virtual laboratory depends on the number of students, their preferences and the OpenGL visualization tasks. Avatars are navigated with a mouse in the horizontal plane. A mouse is also used for pointing with the telepointer, for interaction with collaboration tools, see Figure 1. Avatars' perspectives can be moved swiftly from one collaboration tool to another using shortcut perspective changes. When a *shortcut perspective* is chosen (by hotkeys or by clicking the collaboration tool with Ctrl-LeftMouseButton) the Avatars' movement is animated between the current position to a position and perspective where the Avatar is facing a tool for collaboration. This allows for quick context switching without cumbersome mouse navigation of the Avatar.

We suggest that work in the virtual laboratory should be carried out in pairs using one shared desktop. The two students switch between using the desktop and evalutating the result of the source code compilation. With only two Avatars in the environment the risk of covering each other views will be limited. The design of the virtual laboratory can easily be modified by the students by adding and removing tags of the

† http://www.gamevoice.com
‡ http://www.speakfreely.org

XML file describing the structure of the environment. Assistance can be given by a supervisor, joining the environment as a third Avatar. The supervisor can for example answer questions about laboratory tasks using voice communication and a whiteboard or teach in the VNC development environment.

4. Testbed and Evaluation

Assessment of the effectiviness of the virtual laboratory for CGE will be evaluated by comparing traditional laboratory work to the virtual laboratory. A study based on within subject design is planned. A number of users, divisible by four, with similar knowledge of OpenGL will be split in two groups, G_1 and G_2. The members of each group will be teamed up in pairs. Each pair of users will solve two minor visualization problems, P_1 and P_2, based on modifying available OpenGL code to comply with the result asked for in the laboratory instructions. Pairs in group G_1 start the study by working on a shared physical computer and then by each pair being split to work through remotely located setups of the virtual laboratory. The pairs in group G_2 conduct the study in opposite order with the virtual laboratory first and then continue with the physical computer. Half of all the pairs in both groups are first presented with problem P_1 and then with problem P_2, the other half with problem P_2 and then P_1. To be able to record data efficiently one pair of users will be tested at a time.

Our hypothesis is that there will be no significant difference between solving the visualization problems using a physical computer and using the virtual laboratory.

5. Results and Discussion

Preliminary testing has shown that users (approximately 20 kids, 8-14 years old) after a short instruction quickly (approximately 2-3 minutes) learn both how to navigate the Avatar, surf webpages in the VNC and draw using the whiteboard. The target group for the virtual laboratory are college students that learn computer graphics using OpenGL. The main issue with the virtual laboratory is not ease of use or usability but rather performance problems due to the bandwidth intensive Remote Frame Buffering[21] (RFB) protocol used by VNC. Interaction with a shared VNC desktop can quickly congest a low bandwidth network. Network congestion leads to network latency. VNC need at least bandwidth similar to low profile ADSL connections and response times below 100ms to work efficiently. It remains to be seen if the slow down off the virtual laboratory in the interaction with VNC inhibits the students more than they benefit from the virtual laboratory, see Section 1.

6. Conclusions

The DVE run time environment is new because it provides most features that more complex systems possess, but is easy to customize by the specification of XML code in a text file. The novel virtual laboratory that we will test is usable and extensible. We will evaluate the system and specify the results in a new publication. We will report preliminary results at the conference.

References

1. E. Angel. Teaching a three-dimensional computer graphics class using openGL. *Computer Graphics*, **August**:pp. 54–55, 1997. 2
2. C. Bradley, The evolution of pedagogic models for work-based learning within a virtual university, *Computers and Education*, **38**:pp. 37–52, 2002. 1
3. N. Catenazzi, and L. Sommaruga. The evaluation of the hyper apuntes interactive learning environment. *Computers and Education*, **1999**:pp. 35–49, 1999. 2
4. E. Churchill, D. Snowdon, and A. Munro (Eds.). Collaborative virtual environments. Springer, 336 p., 2001. 2
5. S. Cunningham. Powers of 10: the case for changing the first course in computer graphics. *SIGCSE 3/00*, **3**:pp. 46–49, 2000. 2
6. K. Curran. An online collaboration environment. *Education and information technologies*, **7**:1 pp. 41–53, 2002. 1, 2
7. B. Dalgarno. The potential of 3D virtual learning environments: A constructivist analysis. *Electronic Journal of Instructional Science and Technology*, **3**:19 p., 2002. 1
8. A. Gulz. Spatially oriented and person oriented thinking – implications for user interface design. *Education and information technologies*, **7**:1 pp. 67–80, 2002. 2
9. S. Grissom. uisGL: a C++ library to support graphics education. *Computer Graphics*, **30**:3 pp. 36–37, 1996. 2
10. H. Haga. Combining video and bulletin board systems in distance education systems. *Internet and Higher Education*, **5**:pp. 119–129, 2002. 2
11. T. Howard, R. Hubbold, and A. Murta. Maverik: a virtual reality system for research and teaching. *Proceedings GVE99 Workshop*, **1999**:7 p. 2
12. J. L. Lowther, and C. Shene-Kuang. Rendering + Modeling + Animation + Postprocessing = Computer Graphics. *Proceedings of the seventh annual consortium on Computing in small colleges midwestern conference*, pp.20–28, 2000. 2
13. J. McConnell. United states: computer graphics education in computer science undergoing a transformation. *Computer Graphics*, **30**:3 pp. 31–32, 1996. 2

14. B. Menousek, and T. Wolfe. Virtual reality the modular way. *Computer Graphics*, **31**:pp. 66–68, 1997. 2

15. B. Andersson. VASE Plugin Framework. Online at http://www.hci.uu.se/projects/vase, accessed on 22. June 2003. 1

16. G. Owen. Web based multi-user interactive graphics worlds for education. *Proceedings of Graphics and Visualization Education 99.*, 1999. 1

17. Á. Sánchez, J. Barreiro, and V. Maojo. Design of a virtual reality system for education: a cognitive approach. *Education and information technologies*, **5**:4 pp. 345–362, 2000. 2

18. M. Tavanti, and M. Lind. 2D vs 3D, implications on spatial memory. *Proc. IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, **2001**:6 p., 2001. 2

19. R. Wolfe. Open GL: agent of change or sign of the times?. *Computer Graphics*, **August**:pp. 29–32, 1998. 2

20. D. Wells. eXtremeProgramming. Online at http://extremeprogramming.org, accessed on 22. June 2003 2, 3

21. T. Richardson et al. Virtual Network Computing. *IEEE Internet Computing*, **2**:1 pp. 33–38, 1998. 3, 4

22. M. Lindkvist. A State Sharing Toolkit for Interactive Applications. *Master thesis at KTH Royal Institute of Technology*, **April**, 2002. 3