OPENGL PROJECT PRESENTATION SLIDES INTERFACE AND A CASE STUDY

Serguei A. Mokhov and Miao Song
Concordia University, Montreal, Quebec, Canada

Keywords: Slide presentations, 3D, OpenGL, Real-time, Frameworks, Shaders, Tidgets, GPU, Human-computer interfaces, Interaction, Computer graphics education.

Abstract: We present a concept of a 3D interface to power-point-like presentations using OpenGL. We argue the presentations of such kind are a lot more useful to demonstrate projects, conference talks, or computer graphics education tutorials and lectures with the results of 3D animation, effects, and others alongside with the presentation in place instead of switching between a regular presentation software to the demonstration and back – the demo and the presentation are combined together in one unit and can serve as an educational piece on its own.

1 INTRODUCTION

We present a re-usable open-source framework designed and implemented in C++ for OpenGL-based presentation slides. We position ourselves and argue that such presentation tools are better suited for computer graphics presentations be it a lecture, tutorial, a project, or a conference talk on computer graphics techniques than a generic power-point presentation done say in Impress (Sun Microsystems, Inc., 2008) or PowerPoint (Microsoft, Inc., 2008) followed by the demo. There are several reasons for this argument: one can show progressively a topic on some computer graphic techniques with the immediate effects and the required animation, rendering, texture mapping, shading, and other techniques while retaining the expressiveness of power points. We briefly go over the visual design through an actual case study and show some examples and argue why on may want such a framework for their OpenGL (OpenGL Architecture Review Board, 2008; Angel, 2003; Woo et al., 1999) project as opposed to the standard widespread generic presentation software despite the fact it takes (sometimes a lot) more effort to prepare such OpenGL slides.

2 DESIGN AND CASE STUDY

The design of this OpenGL slide presentation framework consists of two parts – the visual and the software architecture. Further in this work we discuss only the visual aspects of the design in some detail and to save space omit the software design, which is described in more detail in the related work (Mokhov and Song, 2008) and the project's documentation. We discuss a use case the framework was applied to along the way.

The actual final "production" visual design aspects of the OpenGL-based slides are up to the particular programmer or artist that prepares the demo together with the main concept. The presenter can set up the main colors of the fonts, the text of the items and the title bar, as well as the display of frame rate, elapsed time, etc. as they see fit in their overall theme.

The main body is the scene in each slide, potentially with the animation, is completely determined by the nature of the project's scene itself and what is being portrayed. The colors of the power points and other textual widgets, which we call *tidgets*, can be adjusted to match the main theme. All tidgets, can be disabled from being displayed such that they don't clutter the slide screen if there is something happening at the main scene of that slide.

The framework also maintains support for the optional vertex and fragment shaders (Rost, 2004; Everitt, 2003; 3D Labs, 2004a; 3D Labs, 2004b), which are popular among graphics community of

A. Mokhov S. and Song M.
OPENGL PROJECT PRESENTATION SLIDES INTERFACE AND A CASE STUDY.
DOI: 10.5220/0001820604090412
In *Proceedings of the Fourth International Conference on Computer Graphics Theory and Applications (VISIGRAPP 2009)*, page
ISBN: 978-989-8111-67-8

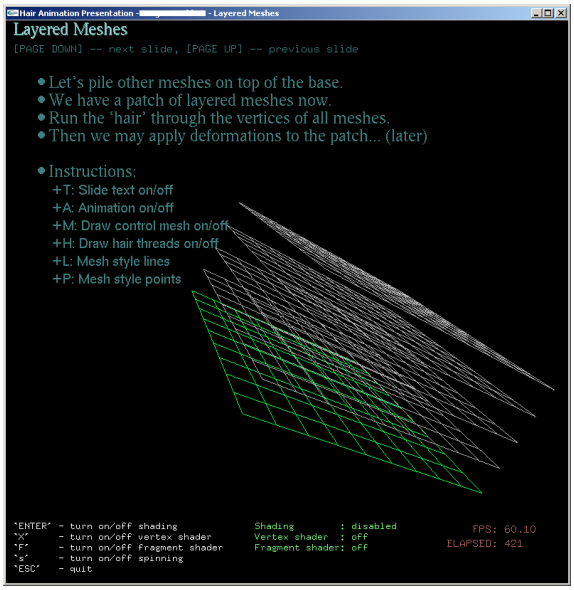

Figure 1: OpenGL Slide Example 1.

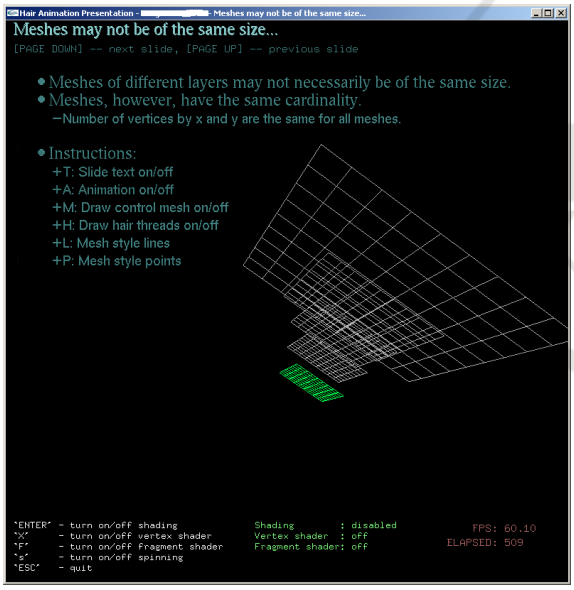

Figure 2: OpenGL Slide Example 2.

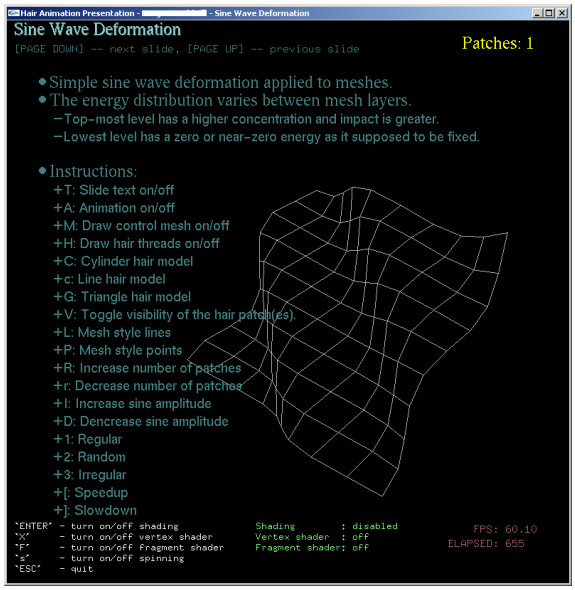

Figure 3: OpenGL Slide Example 3.

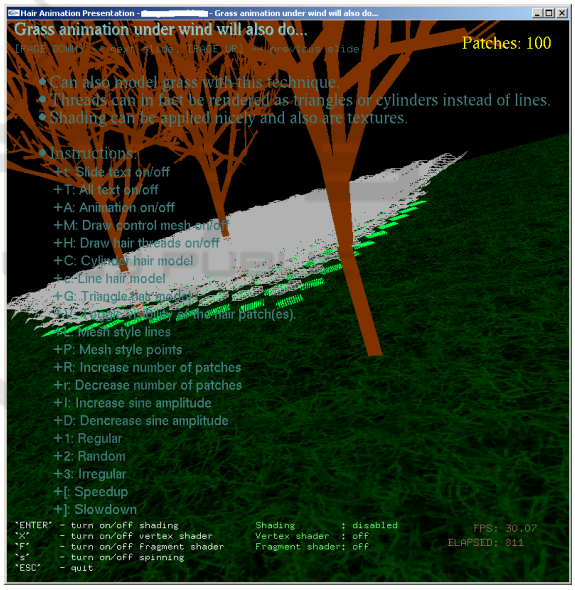

Figure 4: OpenGL Slide Example 4.

GPU programmers and beyond.

One of the use-case is illustrated further through some selected slides of an actual presentation. The example project that uses the framework is the preliminary real-time hair animation modeling done by students (Mokhov, 2004) at the early stages of the development as a progress report. The figures in this work illustrate the examples of the framework's usage from that project (some select slides). The authors are currently designing the equivalent presentation slides for the Softbody Simulation System (Song and Grogono, 2008) and the Open Stereoscopic Plug-in collection (Loader et al., 2008) to further strengthen the concept of the OpenGL slides and make it more usable.

Every slide comprises a scene that can feature its own geometry hierarchy (usually in accordance with the procedural modeling techniques (Wikipedia, 2007)), lighting setup (if necessary), texture mapping, pixel and vertex shaders (Various contributors, NVIDIA Corporation, 2003; Various contributors, Microsoft Corporation, 2003; Kilgard, 1999), or cus-

tom navigation keys. One can see the basic structure of the slides: the slide title on the top left corner in bold, followed by navigational help line how to move between the slides, and some help on the default slide keys at the bottom left, then some state information about the status of the vertex and fragment shaders, and the FPS (frames per second) and the elapsed time metrics tidgets that are updated at run-time, and often are asked about to be shown while doing demo to show how efficient or inefficient some real-time graphics algorithms are.

In Figure 1 is an example of a simple scene with a set of 3D meshes (that can be interacted with as instructed on the slide and to turn on the animation). The slides in Figure 2, Figure 3, and Figure 4 show progressively the increase the level of detail of the graphical information shown on the slide scene from that particular project (Mokhov, 2004).

The framework is designed to be source-code portable, i.e. it can compile and produce a runnable executable in Windows, Linux, and MacOS X platforms, wherever OpenGL and C++ compiler are available.

3 LIMITATIONS

One of the most obvious limitations of the presented framework is the amount effort required to program the slides and as a consequence the presenter has to have some programming experience. Additionally, in the current state the programming language restriction is that of C++, so if the programmer designs the project using another language, they currently have do the required integration support themselves.

Some of these are mitigated as most of such presenters are computer science professors and students and teach an learn the graphics. Once a presentation set of slides created – it can serve both teaching and learning purposes – to present it in class and if the source code is made available to students to learn gradually the techniques from the actual *working* examples, done in the presentations. Some other limitations are to be resolved during the future work on this project.

Another potential limitation of this work is that it requires to be compiled into an executable that cannot be edited on the spot without recompilation. It also places the limitation on the presenter's hardware used in the classroom or a conference that should have all the necessary libraries an drivers to run the executables as well as perhaps a reasonable graphics card. The software requirements and the corresponding space here are however less stringent than the complete office installation or even PowerPoint viewer and does not absolutely require administrative privileges to install. It is also fair to assume at the computer graphics events and classes the computers have the necessary graphics card to enable proper presentations.

4 CONCLUSIONS

We have presented a framework for OpenGL-based programs to make power-point like presentation interfaces to projects as well as teaching and education materials in computer graphics (Tenneson et al., 2008; Talton, 2007) and beyond, where each slide has a navigation capabilities and power points, and things alike. It additionally includes an OpenGL-based scene that can be interacted with within that slide, with the GPU shader support and so on. The framework allows the presenters to integrate their computer graphics project with the presentation saving demo and presentation time to illustrate various stages of the project evolution as well as demo them. This idea can be further extended in providing embedded manuals to the games and projects. While the slides development may require more effort to produce as they are required to be programmed, they integrate much better and can express much better the animation and modeling aspects, even with movies and audio that OpenGL and its extensions allow with the demo that other presentation software tools, such as Microsoft Power Point (Microsoft, Inc., 2008) and OpenOffice Impress (Sun Microsystems, Inc., 2008) cannot allow seamless integration with one's own projects while giving the presenter all of the control and interaction they need to present their work.

Future Work. There are a number of items to improve on in the framework to make it more usable and adopted:

- Release the framework at CGEMS (Jorge et al., 2008) and SIGGRAPH.
- Integrate with the Softbody Simulation System (Song and Grogono, 2008).
- Integrate with the OpenGL stereoscopic plugin-framework (Loader et al., 2008).
- Add an optional GLUI interface for navigation and control when not in the full-screen mode.
- Extend to other languages that support OpenGL, e.g. Java, Python, and others.
- Export functions to export the presentation as a sequence of images automatically in case it is ab-

solutely necessary to produce static slides for incorporation into the regular presentation software.

- Allow for XML-based or `.properties` based configuration files to be imported and exported to generate the slide content and instantiate the appropriate scene, i.e. to reduce the programming effort required thereby making the OpenGL Slide Framework more accessible to non-programmers (e.g. Maya (Autodesk, 2008) users, etc.)

ACKNOWLEDGEMENTS

We acknowledge the reviewers of this work and their constructive feedback. This work was sponsored in part by the Faculty of Engineering and Computer Science (ENCS), Concordia University, Montreal, Quebec, Canada. We also thank Drs. Peter Grogono and Sudhir Mudur.

REFERENCES

3D Labs (2004a). OpenGL Shading Language demo and documentation. 3D Labs, Inc. *http://developer.3dlabs.com/openGL2/downloads/index.htm*.

3D Labs (2004b). OpenGL Shading Language shader examples and source code. 3D Labs, Inc. *http://3dshaders.com/shaderSource.html*.

Angel, E. (2003). *Interactive Computer Graphics: A Top-Down Approach Using OpenGL*. Addison-Wesley.

Autodesk (2008). Maya. [digital]. *autodesk.com*.

Everitt, C. (2003). OpenGL ARB vertex program. NVIDIA Corporation.

Jorge, J., Hanisch, F., Figueiredo, F., and Schauer, R. (2008). CG Educational Materials Source (CGEMS). [online]. *http://cgems.inesc.pt/*.

Kilgard, M. J. (1998–1999). All about OpenGL extensions. OpenGL.org. *http://www.opengl.org/resources/features/OGLextensions/*.

Loader, A. R., Mokhov, S. A., and Song, M. (2008). Open Streoscopic 3D Plugin Collection. SourceForge.net. *http://sf.net/projects/stereo3d*, last viewed April 2008.

Microsoft, Inc. (2008). Microsoft Office Power Point. [digital]. *microsoft.com*.

Mokhov, S. A. (2004). Real-time animation of hair and thread-like objects via deformation of layered meshes. Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada. Project and report.

Mokhov, S. A. and Song, M. (2008). An OpenGL-based interface to 3D PowerPoint-like presentations of OpenGL projects. In *Proceedings of CISSE'08*, University of Bridgeport, CT, USA. Springer. To appear.

OpenGL Architecture Review Board (1998–2008). OpenGL. [online]. *http://www.opengl.org*.

Rost, R. J. (2004). *OpenGL Shading Language*. Pearson Education, Inc. ISBN: 0-321-19789-5.

Song, M. and Grogono, P. (2008). A framework for dynamic deformation of uniform elastic two-layer 2D and 3D objects in OpenGL. In *Proceedings of C3S2E'08*, pages 145–158, Montreal, Quebec, Canada. ACM. ISBN 978-1-60558-101-9.

Sun Microsystems, Inc. (2008). OpenOffice Impress. [online]. *openoffice.org*.

Talton, J. O. (2007). Teaching graphics with the OpenGL Shading Language. *ACM SIGCSE Bulletin archive*, 39.

Tenneson, D., Spalter, A. M., Kumar, J., Medvedev, I., and van Dam, A. (2008). The Graphics Teaching Tool (GTT). [online], Brown University. *http://graphics.cs.brown.edu/research/gtt/*.

Various contributors, Microsoft Corporation (2002–2003). ARB_fragment_program, Revision 26. Microsoft Corporation.

Various contributors, NVIDIA Corporation (2002–2003). ARB_vertex_program, Revision 46. NVIDIA Corporation.

Wikipedia (2007). *Procedural Modeling*. *http://en.wikipedia.org/wiki/*.

Woo, M., Neider, J., Davis, T., Shreiner, D., and OpenGL Architecture Review Board (1999). *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 1.2*. Addison-Wesley, 3 edition. ISBN 0201604582.