ShaderSchool – A tutorial for shader programming

Malte Thiesen, Ulf Reimers, Kristopher J. Blom, Steffi Beckhaus
interactive media / virtual environments
University of Hamburg, Germany
malte.thiesen@informatik.uni-hamburg.de
ulf.reimers@informatik.uni-hamburg.de
blom@informatik.uni-hamburg.de
steffi.beckhaus@uni-hamburg.de

Abstract: *We present a tool for in-class and self-study learning that provides a convenient introduction into GLSL shader programming. The tool presents shaders in an interactive manner, and can be present in-class in a group interactive manner or used as an individual tutorial. In ShaderSchool the materials are presented in sections with interactive assignments integrated into the tool, which help reinforce the students learning. It was created and applied successfully in an university computer graphics class. Additionally, the ShaderSchool tool is extensible to easily incorporate further lessons.*

1 Introduction

This course module was created by as part of the “advanced computer graphics” class in winter 2005/2006 [Beckhaus and Blom 06].

Since the advent of programmable graphics hardware real time graphics programming has changed dramatically. The introduction of so called “shaders” enabled programmers to create visual effects that were previously impossible to achieve. Furthermore, a lot of calculations regarding graphics can now be done by the graphics hardware. Nowadays knowledge of shader programming is essential to obtain top performance and state-of-the-art visual appearance.

However shader programming is fundamentally different than traditional graphics programming. Shaders are written in domain specific languages like HLSL or GLSL that have very special characteristics. In addition advanced knowledge of the rendering pipeline, and how shaders fit into it, is needed.

2 Educational Goals

The goal was to develop an interactive tutorial that provides programmers that are familiar with traditional OpenGL programming with a convenient introduction into GLSL shader programming. Such a tutorial has to impart the required knowledge. The student has to be aware of the basics of the shader language and the functionality of shaders in general before she can do any practical programming. Furthermore, the student should be able to apply the learned information almost immediately by programming simple shaders.

After a successful completion of the tutorial the student should have basic knowledge of GLSL shader programming and be able to apply it by writing shaders or consulting more advanced tutorials and books about the same subject.

3 Methodology

This teaching gem was created as part of a course at the University of Hamburg, in which a new teaching method, Teachlet Tutorials was used [Beckhaus and Blom 06]. As such, this module can be used either as an interactive “Teachlet” for in-class learning or as a stand alone tutorial. More information to the setting in which it was originally held can be found in [Beckhaus and Blom 06] and in the original Teachlets paper [Schmolitzky 05].

The tutorial is implemented as a standalone program that provides the student with everything

she needs to start GLSL shader programming. It is separated in multiple lessons with an increasing level of difficulty. Each lesson introduces a topic of 3D programming by means of a Hypertext document (Figure 1), followed by a task to write a shader using the concepts presented (Figure 2).

Reading the lessons text and implementing the shaders happens inside the program. This gives the user the ability to concentrate solely on the shading language GLSL and not the underlying OpenGL system. Moreover, the student is able to directly apply his knowledge or even just try out ideas of his own without sticking to the task she was given.

4 User Instructions

The tutorial is distributed as a ZIP file. To install, a user has to unpack the content into a new directory. The program is started by executing “ShaderSchool.exe”. There is a German and an English version of the tutorial. Normally it automatically recognizes which language to use, by querying the systems language settings. However, it can be forced to use one or the other option by executing the files “ShaderSchool_Force_English.bat” or “ShaderSchool_Force_German.bat”.

ShaderSchool runs on Windows 2000/2003/XP systems and requires an OpenGL compatible graphics card supporting the following extensions:
ARB_multitexture
ARB_shader_objects
ARB_vertex_shader
ARB_fragment_shader
If a card does not support all extensions, a driver update may help. New driver versions often add support for previously unsupported extensions.

After ShaderSchool has started, the user chooses a lesson by using the "Lesson" menu item. This will load the lesson text and the shaders for the task, which is given at the end of the lesson. The lesson text gives an introduction to the topic discussed in the lesson and always ends with a task, where the user has to write his own shader. The shaders present when the lesson is loaded provide a starting point. They may actually be able to compile and do something, but normally they just help the user to get started.

The shader's source code is opened by clicking one of the tabs, "Vertex Shader" or "Pixel Shader". The source code editor acts like a normal text editor and highlights special GLSL keywords to provide better readability. If the user wants to see the results of her shaders, she has to compile them. This can be done by using "Compile Shaders" in the "File" menu or by hitting "F5." If the compile is successful, the shaders will automatically be applied to the object displayed in the OpenGL window on the upper right side of the ShaderSchool application.

If the user gets stuck and doesn't know how to solve the lesson's task, she can see the solution by selecting "Show solution" from the "File" menu. This will add two new tabs containing a possible solution for the task. By using copy'n'paste, this code can be inserted in the user's shaders and then be compiled to see the result in the OpenGL window.

An example for a ShaderSchool screen is shown in figure 1. It consists of the following elements:

- Tabs (1)

  let the user select the content of the main pane (2). She can choose between the lesson text, the shader source code editors and the solution views.

- OpenGL window (3)

  displays the object associated with the current lesson and has the current shaders applied to it. If a lesson was just started without adding anything to the basic shaders that where loaded with the lesson, there will likely be nothing but a blank screen (the object is there but because it has no shader assigned it won't be visible). By clicking and holding the left mouse button inside the window, the object can be rotated around its pivot point allowing one to see the shader's results from different viewpoints.

- Output window (4)

  this window contains the compile information generated by the OpenGL implementation. If there are syntax errors in your shaders, then it will likely be displayed here. Note that this output depends completely on the OpenGL implementation (e.g. if its from nVidia or ATI.)

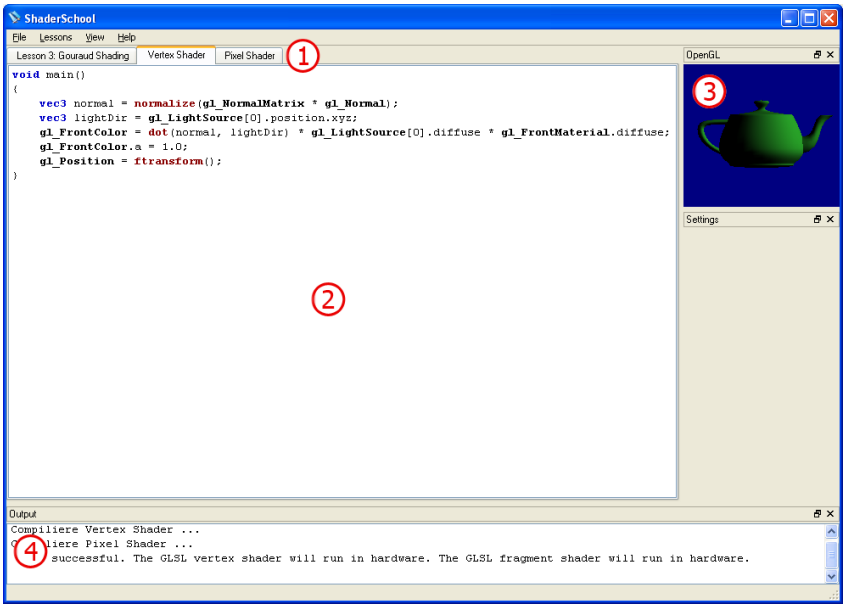

Figure 1: The main elements of the ShaderSchool UI

ShaderSchool was written with extendibility in mind, technical details on extensions can be found in the accompanying file “How to add lessons.pdf”.

4 Assessment

The tutorial was presented to a class of about 15 students. In a single session all lessons of ShaderSchool were worked trough. During this session a single computer was used in conjunction with a video projector. For each lessons a lecturer explained the lessons content using the text provided by ShaderSchool. Subsequently the students collaborated on solving the lessons task. The students dictated source code to the lecturer who entered it into the program.

Even though only few students had previous knowledge of GLSL shaders, they were able to follow the lessons and had no problem coming up with solutions to the tasks. On one occasion the solution provided by the students was even better than the one that ShaderSchool provided. However, we have yet to obtain information about ShaderSchool's effectiveness when it is used in self-study.

5 Conclusions

ShaderSchool is a nice way for beginners in GLSL shader programming to get started on the subject. It is very easy to set up, because it needs no extra software. It's greatest potential is possibly the freedom it grants the student: A student can play around with shaders and get comfortable with them in a very simple environment. ShaderSchool could be extended in numerous ways, the simplest being adding new lessons, but others like saving and loading custom textures and objects are also possible.

References

Beckhaus, S. and Blom, K. 2006. “Teaching, Exploring, Learning - Developing Tutorials for In-Class Teaching and Self-Learning” Eurographics 2006, Education Paper, September 2006, Vienna, Austria, The Eurographics Association

Schmolitzky, A. 2005. “A laboratory for teaching object-oriented language and design concepts with teachlets.” In OOPSLA ’05: Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications. New York, NY, USA. ACM Press, pp. 332–337.

Further Reading

OpenGL Architecture Review Board, et al 2005. “OpenGL® Programming Guide: The Official Guide to Learning OpenGL®, Version 2, 5th Edition”, Addison-Wesley

Randi J. Rost, 2006. “OpenGL® Shading Language, 2nd Edition”, Addison-Wesley

Wolfgang Engel, 2004. “ShaderX3: Advanced Rendering with DirectX and OpenGL”, Charles River Media