Guidelines for Project in EDAN35 High Performance Computer Graphics

Michael Doggett*and Second Author†

Lund University
Sweden

December 11, 2024

Abstract

In this paper, we describe the projects that can be done in the course EDAN35 High Performance Computer Graphics at the Department of Computer Science, Lund University. In order to pass this part of the course, the requirements for each type of project are described, and as well the rules for those of you who wish to participate in the competition. The project also needs to be documented, and you should use the style used in this paper for that, so there is a section on formatting as well.

1 Introduction

As part of the course EDAN35, each group of two[1] students need to create a project. The project should be a 3D Graphics applications based on a 3D engine, for example the framework used in the assignements. The project is worth 3 points.

Any group who wants to can participate in a competition where you present your project.

2 3D Graphics Project

In this project, you have a lot of freedom. You need to write an application using a 3D Graphics API, such as OpenGL, a rendering engine, such as the C++ framework used in the labs, which is a demonstration of 3D graphics algorithm. Examples of applications may include:

- 3D graphics algorithm. From recent paper or GPU programming text, such as GPU Gems
- Game
- Screensaver
- Demo
- Graphics tool
- something completely different

Remember though that as a group of two, you need to spend nearly 6 weeks on this.

The performance of your application is not the most important thing for the project. If you do optimize for performance, that is a good thing (especially for the competition).

You can use the code from the *Deferred Shading Assigment* when you start with this type of project.

2.1 Who wins the 3D Graphics competition?

A jury will decide at the last lecture.

The deadline is at 13:00 on the day before the last lecture and competition. The code should be delivered by then so that it can be demonstrated at the competition. However, note that modifications to the code can be done after that (in order to further impress the jury, or for final grading).

3 Written Report

The report should be handed in using a PDF. You can use any word processing software you like, but you need to generate a PDF for the final submission.

The typesetting for this paper was done using pdfLaTeX. It is recommended that you use this as well, and therefore the "source files" for this very document are available on the course website. If you are not familiar with pdfLaTeX, then you may use whatever other word processing program you like, as long as you mimic the general style in this paper (i.e., you paper should look similar to this paper).

The source files consists of two files: `project.tex` and `project.bib`. There is also a PNG image called `lugg.png` that is shown later in this paper. It is included into the document in `project.tex`, where you should delete this document's text, and instead write your own text.

You should write your report as a scientific paper. It should have (at least) the following sections:

- Abstract [brief summary of key results]
- Introduction [why do what we did? motivation]
- Algorithms or Application [description of what you did, what algorithms you implemented, references to how you found out about them]

*e-mail: michael.doggett@cs.lth.se
†second.author@cs.lth.se

[1] In some cases, groups with only three or one student have to be formed, but this should be avoided as much as possible.

Figure 1: This is the logotype for LUGG: Lund University Graphics Group.

- Results [e.g., performance, **screenshots**, usefulness etc]
- Discussion [what did not work, what worked well, what can be improved, optimizations you tried]
- Conclusion [any concluding remarks – skip if you do not have anything to say in addition to what you've already said]

Some groups may not have enough important material to write a "Discussion"-section, and in such cases, that section can be omitted. Also, look at scientific papers, e.g., [HG97, IEP98, RKLC+11, Dog12, GD15], and try to follow their general style. When in doubt, come and ask us. If you need references not found in the file `project.bib`, simply add your reference to that file. The report should be 2–4 pages long.

You can include screenshots as PNGs or illustrations in the PDF format. An example is shown in Figure 1. See the source file `project.tex` for how this is done in LaTeX.

4 Conclusion

Make a great project. You're clever. Surprise us (and the jury)!

References

[Dog12] M. Doggett. Texture caches. *Micro, IEEE*, 32(3):136–141, May 2012.

[GD15] Per Ganestam and Michael Doggett. Real-time multiply recursive reflections and refractions using hybrid rendering. *The Visual Computer*, 31(10):1395–1403, 2015.

[HG97] Ziyad S. Hakura and Anoop Gupta. The Design and Analysis of a Cache Architecture for Texture Mapping. In *24th International Symposium of Computer Architecture*, pages 108–120. ACM/IEEE, June 1997.

[IEP98] Homan Igehy, Matthew Eldridge, and Kekoa Proudfoot. Prefetching in a Texture Cache Architecture. In *Workshop on Graphics Hardware*. ACM SIGGRAPH/Eurographics, August 1998.

[RKLC+11] Jonathan Ragan-Kelley, Jaakko Lehtinen, Jiawen Chen, Michael Doggett, and Frédo Durand. Decoupled Sampling for Graphics Pipelines. *ACM Transactions on Graphics,*, 3(30):17:1–17:17, 2011.