Stefan Johansson
stefanj@cs.umu.se

UMEÅ UNIVERSITY
DEPARTMENT OF COMPUTING SCIENCE

3D Studio
The Project

Introduction and Objectives
During the course project you will write a limited 3D visualization software. This will make you familiar with 3D transformations, illumination, and shading, and how these are implemented in OpenGL. The project is divided into three parts and has the following requirements:

- This is an individual project.
- The project must be written in **C++** using the graphics standard API **OpenGL**. As window manager you will use **GLFW** and later on you will use **Dear ImGui** for creating a GUI.
- It is allowed to use any external libraries and the code handed out on the course (e.g. the code from the workshops).
- The complete project (including the first two parts) must be orally presented and demonstrated in the end of the course. More info about this later on in the course.
- Only the final code is submitted in the end of the course (no written report).

If the first two parts are **completed and demonstrated** no later than the due date, bonus points are available for the third part. Bonus points can be used to get a higher grade on the first written exam, but not to pass. The demonstration of the first two parts will be in the CS department's Linux labs MIT.A.146 and MIT.A.156.

Take some time in the beginning and think about the system design of your 3D Studio program. It can save a lot of headache later on. You will discuss suitable system designs during a Workshop.

Note! The code in the first workshops can be used as a starting point for the project, but it may not be suitable for your choice of system design.

Overview of the project
During the project you will step-by-step build a fully functional visualization tool for 3D objects. You will do this in three parts which are outlined below. See respective specification for details.

Part 1
In the first part, you will implement the reading of OBJ files and visualizing them as simple wire-frames or filled polygons without any projections or camera. It should be possible to rotate and translate the object using the keyboard. You only need to show one object at a time, but it should be possible to exchange the object with a new one.

Stefan Johansson
stefanj@cs.umu.se

UMEÅ UNIVERSITY
DEPARTMENT OF COMPUTING SCIENCE

Part 2

In the second part, we introduce the camera and projections. These two components must be implemented at the same time.

The projections you will implement are parallel (orthographic and oblique) and perspective projections, where the user can change the projection parameters. To change these parameters, we now also introduce a GUI. This GUI will in part 3 be extended for setting material, light, and texture parameters.

The camera is controlled by using the keyboard and the mouse (as in 3D drawing tools or FPS games).

A possible extension now is to extend the project with the possibility to show several objects at the same time. This is, however, optional for the project. To be able to do this the system design of the project must be reconsidered.

Part 3

In the third and final part, we add lights into the scene and shading of the object. It should now also be possible to add a texture to the object. New parameters for each object are the material properties and the texture. You now also need the vertex normals for shading the object.

During this part you will have the possibility to add extra extensions that may give you bonus points. Examples of that could be: several shading models, more than one object, add a ground floor, and reading texture coordinates and material properties from the OBJ file.

Use of Libraries and Licenses

If you use any existing libraries then it is important that you read the license for each library and check in which circumstances you are allowed to use it. An appropriate disclaimer and any necessary license file(s) must be included in your final source bundle.

OpenGL Tips and Guides

The reference pages for all OpenGL functions and the useful Quick Reference Card can be found at

`https://www.khronos.org/registry/OpenGL-Refpages/`

and

`https://registry.khronos.org/OpenGL-Refpages/gl4/`

For each command and function the supported version of OpenGL is listed at the corresponding reference page. For this course OpenGL 3.3 is sufficient, however, you may use up to version 4.4 (note that the reference pages are for OpenGL 4.5+, so some functions cannot be used).

Stefan Johansson
stefanj@cs.umu.se

UMEÅ UNIVERSITY
DEPARTMENT OF COMPUTING SCIENCE

The glsl-files follows the specification of the *OpenGL Shader Language* (GLSL). If you have problems with data types or any compilation errors, please take a look at the following links:

`https://www.khronos.org/opengl/wiki/OpenGL_Shading_Language`
(Introduction)

`https://www.khronos.org/opengl/wiki/Data_Type_(GLSL)`
(GLSL data types)

(The GLSL specification and Quick Reference Guide are also included in OpenGL's reference pages)

OpenGL on your own computer

We expect you to use the Linux computers in the department's computer labs to do the project. If you like to use your own computer anyway, we are unable to provide any help for installing and configuring the development environment on your personal computer. You must also consider the following:

- The time you need to spend on installing and configuring the OpenGL environment must be taken from your free time.
- If you are planning to bring your own computer to the final oral presentation, test your code in good time before the deadline on one of the computers in the Linux labs! You may not get the same results.
  - You are only allowed to bring your own computer on the final oral presentation in January.
- Make sure your graphics card supports at least OpenGL 3.3 and GLSL 3.30.
- OpenGL does not normally work over an X-tunnel. It must be run locally.
- If you use a virtualization environment, make sure that it supports OpenGL 3.3.
- Linux is preferable, but Windows and Mac should work fine (Mac with some tweaking).
- Usually a default graphics driver is installed by default, but it may not support hardware acceleration. Install the right graphics driver. On Linux this can be nothing but tricky. See if you find instructions and follow it. Also install mesa-utils. You can verify that it works with 'glxinfo | grep OpenGL'. It should confirm the versions supported. You can also try 'glxgears'.
- Install the required libraries.

GLFW Tips and Guides

GLFW is a windows and OpenGL contexts manager and the reference pages can be found at `http://www.glfw.org`. The installed version of GLFW in the computer labs is 3.3.8.

Since GLFW does not include any user interface library you will use Dear ImGui for the GUI. The GUI will be introduced from part 2 of the project and the code for that will be provided.