Introduction to graphics programming Project

Mickaël Sereno - mickael.sereno@inria.fr

27 novembre 2018

1 Description

The purpose of this project is to see if you can apply the graphics concepts seen in class in a bigger project. The project can, but is not restricted to, be :

- Visualization of a physical concept (fluid, vector field, electron diffraction animated, galaxy system (more complexe than the solar system), etc.). This has to be somehow animated, for example by using a character around visualizing it, by moving your components, etc. You have the choice.
- Create an artistic scene.
- Create a 3D simulation of a concept (crowd in emergency fire for example).
- Anything that you think is either *cool*, *creative*, *funny* or require higher computer graphics techniques.

For higher computer graphics techniques (shadows, mirrors, etc.), you can check online tutorials (e.g https://learnopengl.com/). The keyboard / mouse / joystick (if you have one) can be used and useful. The SDL can handle them correctly.

For this project you are gathered from *three to four students*. The project must uses OpenGL 3.0 **at least**, C++ and must compiled on Windows (Visual Studio) and/or Linux. Your grades however will not be based on your C++ skills, but more in your general programmation skills and in the use of computer graphics techniques. Remember to **document** your code : it will be read carefully. A code not documented can be severed.

Please, select a project you are almost sure to finish. A modest but finished/almost finished project is way better than a hard but far from finished project.

The scene has not to be as incredible as it sounds by reading the suggested topics !

2 Add another library

If you add another library, add it to your project for Windows (in the folders *libs*. Follow the hierarchy please : the includes in the folder *include* and the libs for the correct plateform in the correct directories).

The file *CMakeLists.txt* can be modified to add another libraries at compile time. See the TODO flags (search for TODO). It is the same principle than for the SDL library (search for SDL in the CMakeLists.txt). Normally you have to :

- Add another options (include and libs paths) for us to modify the directory paths if needed. Look at what is already written about this (see the SET from line 20 to 47).

— Change the *link_ directories* in order to add another library directory.
— Add compiler options in *target_ compile_ options* and *target_ link_ libraries* for all the platforms (MINGW, MSVC and Linux).

3 Multiple Shaders

You can have multiple Shaders. Add your shaders in the Shaders folder. When compiling with the CMakeLists.txt furnished, this folder will be automatically copied in your binary directory (bin). Only your hardware restricts at how much Shaders and VBOs you can use.

Take however account that more than 1024 VBOs are not always supported. Use instead a VBO for more than one object data, and when doing glDrawArrays / glDrawElements (when using EBOs) select what you want to draw using the *offsets* and *number of vertices* parameters.