

The Last Chapter – Submission 2

Gameplay

The player is controlling a spaceship in first person perspective. The objective is to destroy as many asteroids as possible to save planet earth and not to get hit by any of the objects, or the player loses.

Effects

Depth of Field (1.5 Points)

<https://entwickler.de/online/opengl-weltraumeffekte-mit-motion-blur-lens-flare-co-how-to-114796.html>

Motion Blur (1.5 Points)

<https://entwickler.de/online/opengl-weltraumeffekte-mit-motion-blur-lens-flare-co-how-to-114796.html>
<http://john-chapman-graphics.blogspot.co.at/2013/01/what-is-motion-blur-motion-pictures-are.html>

HDR (1 Point)

<http://learnopengl.com/#!Advanced-Lighting/HDR>

Bloom (1 Point)

<http://learnopengl.com/#!Advanced-Lighting/Bloom>

Effects are visible in a subtle way. If necessary, they can be separately activated, deactivated and modified by adjusting the “*color = ...*” part (manipulation of coefficients) in “*main()*” in “*fboFragmentShader.glsl*”.

Complex Objects

Apart from a flat-shaded sphere, a smooth-shaded sphere and a cube, there is a “space station” model and a “rock”, which is used as the projectile. The space station also has concave parts (on the upper side).

Animated Objects

A hierarchical animation is done while shooting. The space station object is rotating around the projectile and is influenced by its movements.

View-Frustum-Culling

A view frustum is implemented using the geometric approach.

Transparency

Transparency is implemented by modifying the alpha-channel.

<http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-10-transparency/>

Experimenting with OpenGL

Vertex-Buffer-Objects (VBOs), Vertex-Array-Objects (VAOs) and Frame-Buffer-Objects (FBOs) are implemented and being used. Mip Mapping is implemented too and can be turned on and off, as well as the Texture-Sampling-Quality.

Lighting

Specular lighting, diffuse lighting, ambient lighting.

Function Key Mapping

All requested function keys are usable, and a feedback message is displayed in the console if any of them are pressed.

Additional Libraries

- Object loader: **ASSIMP**
<http://www.opengl-tutorial.org/beginners-tutorials/tutorial-7-model-loading/>
- Collision Detection / Physic: **PhysX**
<http://docs.nvidia.com/gameworks/content/gameworkslibrary/physx/guide/Manual/Index.html>
- Math: **GLM**
- **GLEW**
- **GLFW**

Tool for Modeling

Blender

Win Condition

Enough asteroids have been destroyed.

Lose Condition

The player gets hit by any object or earth has been hit by incoming asteroids.