

CANDYBREAK

RTR 2019 - Submission 2

Group 3 - David Ammer, Simon Pointner

19th January 2020

Implementation Description

The two mayor effects were Volumetric Lighting and GPU Particle Systems with Compute Shader. The Volumetric Lighting was implemented using the light Source Depth map which is also used for Shadow calculations and the depth map of the rendered scene. Using those two Depth maps a ray marching was performed along the sight rays of the user. The amount of steps which were visible from the light source were summed up which creates the volumetric lighting effect. For the GPU Particle Shader a compute shader updates the position, velocities and rotations for each particle in a particle system. Further a geometry shader outputs a quad for each particle corresponding to the position and rotation of each particle. For the automatic camera the catmull rom spline from the glm library was used in order to interpolate between control points. These control points were created with the help of a debug camera and are stored in a vector of curves where each curve consists of multiple control points. The scene was created using Blender, all models excepts the muffin are our creations inspired by Figure 1. Fortunately, we had the same graphics card as the lab computers, which is the GTX1060 with 6GB memory.

Libraries

The following libraries were used in this project:

- Irrklang [7]
- GLM [12]
- glad.c [6]
- stb_image [2]
- GLFW [4]
- Assimp [5]
- GLEW [3]

Development Status

The following Effects where implemented, further sources are referenced.

- Volumetric Lighting [11] [13] [10] [1]
- GPU-Particle System (for this effect the lecture slides were sufficient)
- Bloom [9]
- Ripple-Effect [8]
- Void-Effect (inspired by Minecraft, very simple effect)



Figure 1: Inspiration

Controls

- If Debug Camera is enabled in config it can be controlled using WASD.
- ESC to close the application.
- In assets/settings.ini the settings can be edited, like turning the music on and off or toggling the fullscreen mode.

References

- [1] Github - metzzo/ezg17-transition: Repository for the project at the realtime rendering course at university of technology vienna. <https://github.com/metzzo/ezg17-transition>. (Accessed on 01/19/2020).
- [2] Github - nothings/stb: stb single-file public domain libraries for c/c++. <https://github.com/nothings/stb>. (Accessed on 01/19/2020).
- [3] Glew: The opengl extension wrangler library. <http://glew.sourceforge.net/>. (Accessed on 01/19/2020).
- [4] Glfw - an opengl library. <https://www.glfw.org/>. (Accessed on 01/19/2020).
- [5] Home. <http://www.assimp.org/>. (Accessed on 01/19/2020).
- [6] <https://glad.dav1d.de>. <https://glad.dav1d.de/>. (Accessed on 01/19/2020).
- [7] irrklang - an audio library for c++, c# and .net and high level 3d and 2d sound engine. <https://www.ambiera.com/irrklang/>. (Accessed on 01/19/2020).
- [8] Kyle halladay - screen space distortion and a sci-fi shield effect. <http://kylehalladay.com/blog/tutorial/2016/01/15/Screen-Space-Distortion.html>. (Accessed on 01/19/2020).
- [9] Learnopengl - bloom. <https://learnopengl.com/Advanced-Lighting/Bloom>. (Accessed on 01/19/2020).

- [10] Light scattering with opengl shader. <https://www.fabiensanglard.net/lightScattering/>. (Accessed on 01/19/2020).
- [11] Nvidia volumetric lighting | nvidia developer. <https://developer.nvidia.com/VolumetricLighting>. (Accessed on 01/19/2020).
- [12] Opengl mathematics. <https://glm.g-truc.net/0.9.9/index.html>. (Accessed on 01/19/2020).
- [13] Volumetric lights - alexandre pestana. <https://www.alexandre-pestana.com/volumetric-lights/>. (Accessed on 01/19/2020).