Demo: Alpenföhn

Real-Time Rendering WS22

Ole Siemers

January 2023

1 Scene

The scene features an alpine landscape, in a foggy morning setting. The camera will fly through the fog and above the landscape to showcase the implemented effects. A screenshot is shown in Figure 1.

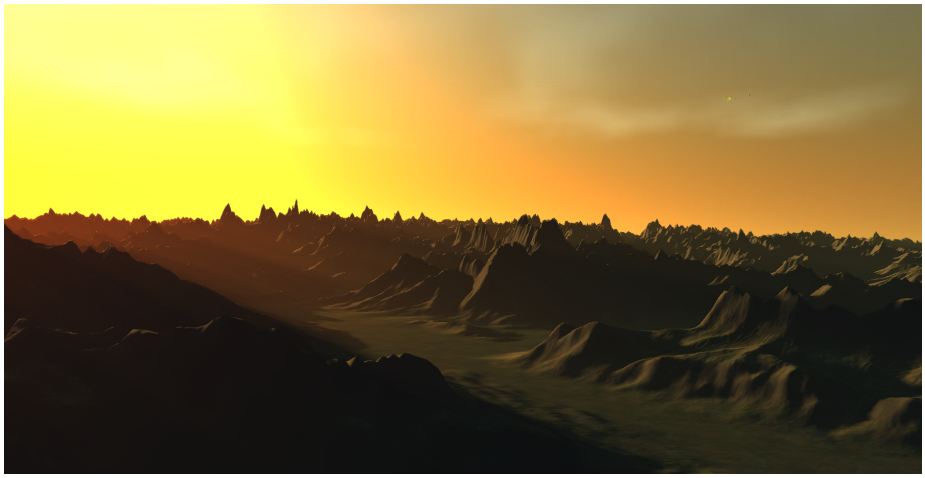

Figure 1: Screenshot of the Demo

2 Effects

The main effects are the heightmap-tessellation and volumetric light.

2.1 Heightmap-Tesselation

The terrain is modeled as vertex-displacement based on heightmap data. The mesh is adaptively tessellated for rendering the mountains with proper level of

detail. The amount of tessellation is depending on the distance from the vertex to the viewpoint. The heightmap was downloaded from [9].

2.2 Volumetric Light

The volumetric light [10] is rendered in half-resolution, depth-aware upsampled like in [6] and blended with the scene. The approach is similar to [11], except that no bilateral filtering is used to filter the low-resolution dither-pattern. Some stratus-clouds are modelled with fbm-noise from layers of 2D-simplex-noise [7].

2.3 Skylight

An analytical skylight model is used to render the color of the sky [8]. The sourcecode is adapted from [12]. A slight bloom effect is added.

2.4 Debug Controls

The camera will follow a predefined path. Use the control-keys explained in Table 1 to debug the effects. If the showcase is finished, one can move around manually with A, S, D, W and Mouse.

control	*action*
F2	toggle wireframe mode
F3	disable volumetric light
F4	disable downsampling
F4	disable bloom
Page Up	rewind daytime
Page Down	fast-forward daytime
ESC	enter manual mode / exit the application

Table 1: Controls to debug the application

2.5 Settings

Settings are defined int the file *config/config.ini*. Adjustable settings are: resolution, the number of raymarching steps, the field of view in degree, the time of day in the interval $[0, 1)$.

3 Used Libraries

The demo is written in C++ and OpenGL/GLSL. The library "GLAD" version 0.1.36 is used for function loading of the OpenGL-functions.[5] The library "GLFW" version 3.3.8 is used for creating and managing the OpenGL context and also the handling of user input.[3] The library "stbimage" version 2.26 is used for loading images for the use in textures.[2] The library "GLM" version

0.9.9.8 is used for vector- and matrix-operations.[4] The library "ASSIMP" is used for model-loading.[1]

References

[1] Kim Kulling et al. *ASSIMP*. https://github.com/assimp/assimp. 2022.

[2] Sean Barrett. *stbimage*. https://github.com/nothings/stb. 2022.

[3] *GLFW*. https://www.glfw.org/. 2022.

[4] *GLM*. https://github.com/g-truc/glm. 2022.

[5] David Herberth. *GLAD*. https://github.com/Dav1dde/glad. 2022.

[6] Louis Bavoil Jon Jansen. *Fast rendering of opacity-mapped particles using DirectX 11 tessellation and mixed resolutions*. 2011.

[7] Ian McEwan. *WebGL Noise*. https://github.com/ashima/webgl-noise/blob/master/src/noise2D.glsl. 2011.

[8] Smits Preetham Shirley. *A Practical Analytic Model for Daylight*. 1999.

[9] *Tangram Heightmapper*. https://tangrams.github.io/heightmapper/#9.4375/47.1459/9.7532.

[10] Umenhoffer Tóth. *Real-time Volumetric Lighting in Participating Media*. 2009.

[11] Nathan Vos. "Volumetric Light Effects in Killzone: Shadow Fall". In: *GPU Pro 5*. Ed. by Wolfgang Engel. A K Peters/CRC Press, 2014. Chap. 3.

[12] Dihara Wijetunga. *sky-models*. https://github.com/diharaw/sky-models. 2019.