Submission2 - Documentation

Demoname: Hot Island
Group: Group 4
Member(s): Alexander Heinz, 01426648
Repository: https://github.com/Schandor2008/ezg18-hotisland.git

Recap from submission0:
The scene shows a lonely island and the starting point is a sandy beach that is meeting the sea. We are staring at the sea as suddenly the earth starts to shake. In the middle of the island there is an active volcano. Of course we are climbing up the volcano, take a look inside the mountain and see that there is hot lava in it. So we start to climb down the mountain and at the same time the volcano starts to explode. Fireballs fly past us and set plants on fire. When arriving at the sandy beach again we can see a very big smoke cloud over the volcano.

Implementation:
The scene was not implemented exactly like mentioned in submission0. The following things are missing or different: Earth is not shaking; Volcano is giving much smoke from the beginning and not at the end; Plants are not getting on fire. Except these everything was implemented like stated.

The whole terrain (beach, grass and volcano) is created on the GPU using a computeshader with a heightmap as source. Positions, Normals, UVs and Tangents were computed on my own on the GPU. With a mask texture (self created in GIMP) the different regions are determined. Only the top part (volcano) has normalmapping, tesselation and displacementmapping enabled. Between regions everything is interpolated. Tesselation is adaptive according to camera distance and patch culling (backface and view frustum) is enabled in the TesselationControlShader. Also the displacement strength is interpolated according the camera distance in order to decrease popping a little bit. Pn-Triangles were not implemented, because there was an issue with that and not enough time to fix it.

Particles are stored in systems laying on the GPU in buffers. They are simulated with Transformfeedback and a GPU shader program. Of course computeshader could be used also here, but in my opinion Transformfeedback was much easier to setup for this. The fire particles are bloomed (the red channel) and additive blended. The smoke particles are not bloomed and alpha blended (NOT SORTED). Both systems are faded in and out accordingly.

The water is just a plane and illuminated by a normal- and dudvmap. When rendering the Reflection and Refraction maps, only the lower part of the terrain, the skybox and one entity are processed. The refraction part should be more visible at steeper looking directions. (It is not very noticeable, but some may pause the playback and take a closer look).
Drawback: At the beach there are ugly edges visible. A solution would be to take in the underlying depth and fade the water accordingly, but I had not enough time …) << DONE

The shadows of the entities are only visible on the terrain (no self shadowing). Due to the fact that I used one directional light source as the sun the projection matrix for the shadowmap was orthographic. Additionally it is updated every frame that the sun camera is always looking at the demo camera position.

Therefore some shadows may appear or disappear very roughly when moving. A solution would be to fade the shadow in or out according to the camera distance. << DONE
PCF was implemented and 5 x 5 = 25 samples are taken from the shadowMap in order to decrease hard edges.

Libraries:
I used the following (standard) libraries:

- GLFW
- GLEW
- ASSIMP (model loading)
- DevIL (texture loading)

Effects:

- NormalMapping in the whole scene (water, terrain, entities)
- Water with Refraction and Reflection
- Terrain with Hardware Tesselation and DisplacementMapping
- Smoke and Fire particles with soft particles stored and simulated on the GPU using transform feedback buffers
- Bloom with fire particles
- Skybox with Cubemap
- Fog in the horizon
- Shadow mapping + PCF (Shadows from entities on terrain)

References:

- Slides from Algorithms for Realtime-Rendering Lecture 2018
- Slides from Realtime-Rendering Lecture 2017
- http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/
- https://developer.nvidia.com/gpugems/GPUGems/gpugems_ch01.html
- https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter18.html
- http://ogldev.atspace.co.uk/www/tutorial30/tutorial30.html
- http://ogldev.atspace.co.uk/www/tutorial31/tutorial31.html
- http://www.opengl-tutorial.org/intermediate-tutorials/billboards-particles/
- https://learnopengl.com/Advanced-Lighting/Bloom
- https://learnopengl.com/Advanced-Lighting/Normal-Mapping
- http://ogldev.atspace.co.uk/www/tutorial26/tutorial26.html

Tools or Sources:

- https://free3d.com/
- https://archive3d.net/?tag=palm
- CG Textures
- ARTR course from last semester
- Basically google pictures search

- Terrain is self created from googled heightmap
- GIMP

Controls:
Demo playback can be stopped everytime by typing F1. Typing F1 again continues with demo playback. Additionally the following controls have been implemented:

F1	Playback toggle
F2	Toggle hardware tesselation
F3	Toggle displacement mapping
F4	Toggle normal mapping in scene
P	(Debug: prints current camera position and direction)
V	Toggle volcano
F	Toggles flying with manual camera
RIGHT	Fast forward playback
LEFT	Slow down playback
ESCAPE	Exit
Movement	WASD, E (up), Q (down), SHIFT (Running), Mouse

Tested on NVIDIA!!