Real-Time rendering

Final submission

Group 17

Dejan Belic, 12243268
Jakub Nawrocki, 12231223

Project implemented within custom C++ Vulkan rendering engine. No major changes in plans as compared to the initial proposal.

The engine uses the following libraries:

- Vulkan HPP (https://github.com/KhronosGroup/Vulkan-Hpp) for vulkan renderer.
- GLFW (https://github.com/glfw/glfw) for windowing and input handling
- GLM (https://github.com/g-truc/glm) for linear algebra
- tinyobjloader (https://github.com/tinyobjloader/tinyobjloader) for loading Obj models from disc
- STB (https://github.com/nothings/stb) for loading images from disc.
- TBB for multithreading
- JSON (https://github.com/nlohmann/json) for load of configuration files

Submitted build version was run with Nvidia's 3070Ti mobile, GTX 1080 and integrated Vega 8 card.

Effects present in the build:
- Directional lights (Diffuse, ambient, specular)
- Point lights
- Normal maps, specular maps
- 4xMSAA
- Mip-mapping
- Skybox
- Transparency
- Directional shadows
- Volumetric fog
- Screen space ambient occlusion (This one is only noticeable near corners. It may be easier to spot by flipping between render mods 4 & 0).
- Screen space reflections (Difference can be easier to spot by toggling them on and off with X key).
- Depth of filed (Most noticeable when close / clipping near an object. Focus is set pretty high, so its hard to spot).
- Tone mapping
- View frustum culling

Camera will move on its own along a predefined path, but the following controls can still be invoked:
- W, A, S, D -> Camera movement
- I, J, K, L -> Camera rotation
- E, Q -> Go up, down
- R -> Reset location
- Space -> rotate scene
- Z -> Reload shaders
- X -> Toggle reflections
- C -> Print out camera position

- Escape -> Close application
- B -> Move directional light origin
- F -> Print out fps metrics
- 0 -> Default render mode
- 1 -> Normal render mode
- 2 -> Lighting render mode
- 3 -> SSAO render mode
- 4 -> Default without SSAO render mode