Philipp Koegel, Bastian Mattes, Jan Doerntlein *Erlangen, 07.01.2015*

GraPro: Graphics Project (Aufgabenblatt 5)

Aufgabe 1 [4 Punkte] OpenGL Framework. The following assignments are based on a OpenGL 4.4 engine that is written from scratch.

1. Basic Setup (1 Punkte). Wrapper classes for easy use of shaders, materials, debugging etc.
2. Object loading (1 Punkte). Use Assimp to load Wavefront obj/mtl files.
3. Texturing (1 Punkte). Implement Blinn/Phong shading and load a scene textured.
4. GUI (1 Punkte). Integrate a GUI library.

Abbildung 1: Possible result of assignment 1.

Aufgabe 2 [10 Punkte] Voxel Cone Tracing. The indirect illumination algorithms implemented in the previous assignments are computationally expensive. Using accelerating data structures like BVHs or kd-trees results in greats speed ups but still the time to render an image is far from real time. While this is almost impossible to accomplish with CPU rendering there are still ways to gain some more speed without a perceivable loss in quality.

The algorithm of Crassin, Cyril, et al. encapsulates a number of rays with approximately the same direction in a ray cluster. This cluster is represented as a cone. In a preprocessing step the scene is rendered from the light sources and the radiance and light direction is saved in an octree. The visibility and the energy of the cone can then be estimated very fast by testing against the voxels of the octree. If the result is not relevant for the upcoming computations all rays in one cluster can be discarded.

1. Voxelization (2 Punkte). Sort fragment info into a voxel grid.
2. Octree (3 Punkte). Build an octree using Compute Shaders and sort the voxel fragments into the leafs.
3. Filtering (2 Punkte). Filter values from the octree leafs and save them into a 3D brick texture.
4. Rendering (3 Punkte). Render from camera and sample indirect light (diffuse and specular) using voxel based cone tracing.

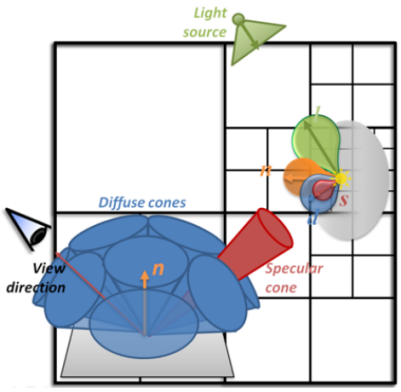

Abbildung 2: Voxel Cone Tracing.

Aufgabe 3 [6 Punkte] Extensions for assignment 2. To create a more realistic look of the voxel cone tracing algorithm it is extended with an ambient occlusion technique that makes use of cone tracing here. Furthermore the meshes should support emissive material. Also the algorithm should be fast enough that a live demo can be presented.

1. Ambient Occlusion (2 Punkte). Cone Tracing Ambient Occlusion.
2. Area Lights (2 Punkte). Emissive Material.
3. Performance (2 Punkt). Live Demo.

Abbildung 3: Possible result after assignment 3.

Sources:

1. Crassin, Cyril, et al. “Interactive indirect illumination using voxel cone tracing.” Computer Graphics Forum. Vol. 30. No. 7. Blackwell Publishing Ltd, 2011.