Roaming Blocks

Dorin Postolache, 0926765
Florian Egger, 0508085

Vorwort:
Für die rendering-pipeline haben wir die Tutorials von http://www.learnopengl.com/ benutzt, das heißt es werden sich Code-Stücke dieser Seite auch in unserer Applikation finden.

Gameplay:
Das Spiel ist ein First-Person Rätsel-Spiel. Das Ziel des Spiels ist es den Weg zum Pokal zu finden. Dabei stehen einem jedoch Hindernisse im Weg die anhand der Bewegung des Spielers zur Seite gebracht werden müssen. Dabei gibt es 2 Arten von Hindernissen: fixe und bewegliche. Die beweglichen Teilen sich wieder in 2 Kategorien: die, die sich mit dem Spieler bewegen und die, die sich entgegengesetzt des Spielers bewegen. Die Idee des Spiels ist es die fixen Blöcke zu verwenden um die beweglichen aus dem Weg zu bekommen. Viel Spaß! ☺

Complex objects:
Der rotierende Pokal, der das Ziel anzeigt, sowie die 2 Sterne die um den Pokal und um die eigene Achse rotieren, werden mit ASSIMP geladen.

Animated objects:
Der Pokal rotiert um die eigene Achse und 2 Sterne rotieren sowohl um die eigene Achse, als auch mit dem Pokal.

Transparency:
Transparency is used within the Particle System.

Experimenting with OpenGL:
We use VBOs and VAOs for our block-Meshes and for the models loaded with ASSIMP.
We have FBOs in use, one for Shadow Mapping, and two for Bloom.

F-Tasten:
Frametime und Wire Frame werden angezeigt.

Illuminated Objects, Light Sources & Textured Objects:
Es gibt für die Blöcke orthogonales Licht (oben mittig im Raum) mit Schattenwurf und Spotlight („Taschenlampe-Effekt“).
Alle 3 Lichtanteile (ambient, diffuse und specular) werden mit der Entfernung immer schwächer (attenuation).
Das ambiente Licht wird aus der Textur-Farbe und dem ambienten Lichtanteil erzeugt.
Das diffuse Licht wird mit dem Phong-Model berechnet und von den Cones des Spotlights begrenzt.
Der spekulare Anteil wird mit dem Blinn- Phong-Model berechnet. Außerdem haben die Würfel eine specularMap, die den spekularen Anteil am „Metallrand“ der Würfel hervorheben.
Dort wo das Spotlight gerade hinzeigt wird auch der Schatten abgeschwächt.

Additional Libraries used:
- GLM für Vektoren, Matrizen und anderen Berechnungen
- GLFW für das Fenster und die Steuerung
- SOIL für die Texturen (http://www.lonesock.net/soil.html)
- ASSIMP für Model-Loading (http://www.assimp.org/)

Effects:

Shadow Mapping:
(in basic.vert – Methode shadowCalculation)
Es wird das übliche Shadow-Mapping-Verfahren angewendet. In ein FBO von der Lichtquelle aus die Tiefeninformation rendern und dann diese „shadowMap“ im zweiten Durchlauf als Textur für den Vergleich verwenden. Um ShadowAcne entgegen zu kommen wird ein Bias verwendet. Weiters wird PCF mit einem 5x5-Kern angewendet.
http://www.learnopengl.com/#!Advanced-Lighting/Shadows/Shadow-Mapping

Spotlight:
Das Spotlight hat einen inner und einen outer Cone, wobei die Intensität des Lichts (diffuse+spekular) außerhalb 0, innerhalb 1 und zwischen den Cones zwischen 0 und 1 geclampt wird.
Die Cone-Weiten werden als Cosinus-Werte der jeweiligen Winkel an den Shader übergeben.
http://www.learnopengl.com/#!Lighting/Light-casters

Bloom:
http://www.learnopengl.com/#!Advanced-Lighting/Bloom

Particles:

Toon:

(Submission 1)

Illuminated Objects, Light Sources & Textured Objects:
Die Blöcke sind sowohl texturiert, als auch beleuchtet.
Spekular Map:
Die Würfel haben ein „Material“, das durch ein spezifisches diffuses und spekulares Licht definiert wird. Man kann das am „Metall-Rahmen“ der Blocks erkennen: Diese reflektieren das Licht im Gegensatz zum Holzteil des Blocks. Das wurde mit Texturen gelöst (container2_specular.png).

Freely moveable camera:
Es gibt eine Kamera-Position (ist am Anfang auf (0,0,5) gesetzt, einen Front-Vektor, der zeigt die Richtung der Kamera (-z), und einen “Hinauf”-Vektor, der dafür sorgt, dass sich die Kamera nicht kopfüber drehen kann. Mit der glm-lookAt Funktion und diesen 3 Vektoren berechnen wir view-Matrix, die für die Darstellung der Szene gebraucht wird.

Moving objects:
Die Blocks bewegen sich abhängig von der Spieler-Bewegung, außer auf Bereiche die von anderen Entitäten eingenommen werden (andere Blöcke, der Spieler oder Wände/Böden).

Texture Mapping:
Wir importieren png/jpg-Dateien mit SOIL (SOIL_load_image). Dieses gibt uns die Breite, Höhe und das Bild als char zurück. Mit glTexImage2D laden wir dann die Texturen hoch, um sie für später – in der game-loop - zur Verfügung zu stellen. Mit glTexParameter setzen wir einige Textur-Parameter, z.B. wie die Texturen „gewrapt“/ausgerollt werden, oder was für Filter bei näherer bzw. entfernterer Ansicht verwendet werden sollen (mipmaps).
In dem Shader sprechen wir die Texturen mit sampler2D an und verwenden sie mit dem Befehl texture. Die Texturkoordinaten kommen über das VBO.

Simple lighting and materials:
Wir haben das Blinn-Phong-Model implementiert. Dazu brauchen wir eine Lichtquelle mit Position, ambienten, diffusem und spekularem Lichtanteil.

Ambientes Licht wird als Licht das aus der Umwelt kommt definiert. Um diffuses Licht zu bekommen nehmen wir das Skalarprodukt von Normalvektor und Vektor der Richtung zur Lichtquelle. Für den spekularen Anteil benötigen wir zusätzlich die Kamera-Position, um die Blickrichtung (vom Fragment zur Kamera) zu berechnen. Diese wird dann verwendet um mit der Reflektion des Lichtvektors an der Normalen das Skalarprodukt zu bilden, das zusätzlich mit einem gewissen Faktor (shininess) potenziert wird.
Alle 3 Anteile ergeben zusammen-addiert die Farbe des Fragments.

Controls:
WASD + Maus + Zoom
für debug-Zwecke auch Q & E für rauf und runter
All diese Funktionen basieren auf den _callback-Funktionen von GLFW.
Die Maus-Bewegung basiert auf den Berechnungen der Euler-Winkel (Pitch und Yaw).

Basic Gameplay:
Es gibt noch keine win-condition, aber zwei Blöcke bewegen sich bereits so, wie sie es sollen, abhängig von der Bewegung des Spielers. Der Spieler kann sich dabei frei bewegen während die Blöcke an ein 3D Raster gebunden sind und sich nur bewegen, wenn der Spieler ein neues Feld des Rasters betritt. Der Spieler hat noch keine Collision Detection, aber die Blöcke beachten bereits dass sie kein blockiertes Feld betreten.