



CGUE 2018

## Submission 2 – final Game

Max Irendorfer - 01634015

Dominik Eitler - 01633008

# Dokumentation – Submission 2 – Final Game

## Implementierung

Die Implementierung basiert auf dem von der LVA Einführung in die Computergraphik zur Verfügung gestellten Framework. Darauf aufbauend haben wir eine grundsätzliche Struktur rund um die Physik-Engine und unsere Levelobjekte aufgebaut, um danach die einzelnen Features einzubauen.

## Features

Die Features des Spieles sind die Cell-Shading Optik, welche durch Konturen noch verstärkt wird, ein ebenfalls im Cell-Shading look gehaltenes Water-mesh, ein Partikelsystem und simple normal mapping.

## Illumination

Zur Beleuchtung verwenden wir ein Directional-Light für generelles Licht und ein Point-Light für eine atmosphärische Zusatzbeleuchtung. Die Texturen beinhalten Lightmaps, welche für Schatten sorgen.

## Additional Libraries

Für die Physik wurde Nvidia PhysX[1] verwendet, für das Einlesen von .obj Dateien assimp[2]. Zusätzlich dazu wurden noch FMOD[3] für Sound und freetype[4] für Schrift-rendering eingebunden.

## Effects

Cel-Shading + Contours (edge detection)

Water Mesh

Simple normal mapping

GPU-Particle System (Transform Feedback)

Der Celshading Effekt wird durch Konturen verstärkt. Diese werden mit einem Sobel Filter in einem screen-shader gezeichnet. Dieser läuft über einen depth-buffer und color-buffer, welcher die Oberflächennormalen farblich kodiert, um eine bessere edge-detection zu erzeugen.

Für das Water Mesh werden im Vertex Shader die Vertex-Positionen und Normalen mit einer Sinus-Funktion manipuliert, um einen Wellen-Effekt zu erzeugen. Zusätzlich ist die Plane mit ein paar Kanten versehen, um die Sobel-edge detection zu triggern, was den Cel-Shading Effekt auf dem Wasser verstärkt.

Das Simple normal mapping findet bei jeder Textur Anwendung, was eine zusätzliche räumliche Tiefe auf dieser erzeugt. Dazu wurde für jede Textur eine normal map erstellt, die dem shader mitgegeben wird um die Normalen anhand dieser Textur neu zu setzen.

Das Partikel-System wird auf der GPU berechnet und gezeichnet. Es wurde mittels Transform Feedback und daher ohne Compute Shader, dafür mit einem 2. Geometry Shader implementiert.

## Special Features

Da sich unser Spiel um Sound dreht (vgl: The Sound of Light), beinhaltet unser Spiel auch Sound, welcher ein sehr zentrales Element des Gameplays darstellt.

## Tools to create Models

Für das Erstellen der Objekte wurde Blender[5] verwendet. Es wurden uv-maps erstellt und die Objekte wurden als .obj-Dateien exportiert, welche vom Programm eingelesen werden.

## Gameplay

Das Ziel des Spiels ist es, immer weiter nach unten zu gelangen. Am Anfang eines Levels wird eine Melodie abgespielt, welche der/die Spieler\_in durch das Betätigen von Druckplatten in einer bestimmten Reihenfolge rekreieren muss. Die Melodien sind sehr einfach und kurz gehalten, da es im Vorhinein keinerlei Hinweise auf die Reihenfolge gibt. Wenn dies passiert ist, öffnet sich in der Mitte des Levels die Luke zum nächsten. Diese kann auch mit der Taste "O" (null) geöffnet werden. Alle anderen Controls können mit "F11" abgerufen werden.

[1]: <http://www.nvidia.de/object/nvidia-physics-de.html>

[2]: <http://www.assimp.org/>"<http://www.assimp.org/>

[3]: <https://www.fmod.com/>

[4]: <https://www.freetype.org/>

[5]: <https://www.blender.org/>