

## **Spiel:**

Sobald alle Gegner abgeschossen wurden, hat man das Spiel gewonnen. Wenn der Spieler von einem Gegner berührt wurde startet das Spiel neu.

## **Implementierung:**

Die Collision Detection ist über eine einfache axis aligned bounding box implementiert. Die Kamerainputs werden über Callbackfunktionen geregelt. Da es in OpenGL kein Kamerakzept gibt verschieben wir hier die Umgebung angepasst nach unserem Input, um so das Gefühl der Bewegung zu vermitteln. Die Viewmatrix erstelle ich mit der LookAt Funktion der GLM-Library.

Der Gegner bewegt sich zum Spieler, indem mit seiner aktuelle Position und der Position der Kamera ein Richtungsvektor berechnet wird, dieser wird dann normalisiert und als Position für den Gegner weitergegeben. Daraus wird dann eine Modelmatrix durch die glm::translate Funktion erstellt.

Die Materialieninformationen sowie die Texturinformationen der Objekte werden, nachdem das Modell in eine ASSIMP Datenstruktur geladen wurde, aus der Datenstruktur in einer Model-Klasse gespeichert, von der aus sie den entsprechenden Shadern übergeben werden.

Der Penis wurde in Maya modelliert und rigged und animiert.

## **Features:**

1. Gameplay
2. Frei bewegbare Kamera
3. Umgebung (Skybox, Inseln)
4. Einfache Beleuchtung
5. Texturen (Skybox) und Materialien (Inseln, Gegner)

## **Beleuchtung:**

Alle Objekte in der Szene sind mit einem einfachen Beleuchtungsmodell (Phong) ausgestattet. Die Inseln und die Gegner werden mit dem Phong-Model beleuchtet. Die Skybox besitzt keine Beleuchtung. Es wird generell directional light verwendet.

## **Texturen:**

Die Skybox besitzt eine Textur, die mit Cubemapping zu einer schöneren Umgebung beiträgt. Die Inseln sowie die Gegner besitzen keine Textur, haben dafür verschiedene Materialinformationen die an die Shader übergeben werden.

## **Effekte:**

**Shadow Mapping** (mit PCF) wurde auf die Inseln angewendet (Quelle: Learn OpenGL). (1.5) Die Schatten sind gut unter den Baumkronen erkennbar.

Model Rigging, Animation und **Vertex Skinning auf GPU** wurden für den Gegner verwendet (Quelle: OGLdev). (2.0)

**HDR** als Post Processing Effekt für die gesamte Szene (Quelle: Learn OpenGL). (1.0)

Tastenbelegungen:

**U** – Unverwundbarkeit aktivieren

**W-A-S-D** für Fortbewegung, **Leertaste** fürs Schießen und **Maus** für Kamerabewegung.

**Backspace** – Neustart des Spiels

F2 – FPS on/off (sichtbar oben neben dem Window Title des Fensters)

F3 – Wire Frame on/off

F4 – Texture Sampling an der Skybox (Nearest Neighbor und Bilinear)

F5 – Mip Mapping on/off Nearest Neighbor und Trilinear (kein Unterschied zu Bilinear)

F7 – HDR on/off

F8 – Frustum Culling on/off (funktioniert nicht ganz)

F9 – Blending on/off

O-Key – Verändert Exposure Wert von HDR um +0.1

P-Key – Verändert Exposure Wert von HDR um – 0.1

K- Key und L-Key – (Exposure = 2 Wert und Exposure = 5)

Rückmeldungen über Aktivierung und Deaktivierung bei den F1 – F9 Tasten werden in der Konsole ausgegeben!

## Libraries:

- SOIL - <http://www.lonesock.net/soil.html>
- GLEW - <http://glew.sourceforge.net/>
- GLFW - <http://www.glfw.org>
- Assimp - <http://assimp.sourceforge.net/>
- GLM - <http://glm.g-truc.net/0.9.8/index.html>
- IrrKlang - <http://www.ambiera.com/irrklang/downloads.html>

## Tutorials:

- Learn OpenGL - <https://learnopengl.com/>
- OGLdev - <http://ogldev.atSPACE.co.uk/> (für Animation und Skinning)
- Lighthouse3D - <http://www.lighthouse3d.com> (Versuch des Frustum Cullings)
- miguelcasillas - <http://www.miguelcasillas.com/?mcportfolio=collision-detection-c> (collision detection)