

FAT HAMSTER
Projekt-Statusbericht
(2nd Submission)

Steuerung

| Key | Effect |
|---------|-------------------|
| W,A,S,D | Hamster bewegen |
| Space | springen |
| Maus | Kamera schwenken |
| ESC | Spiel beenden |
| R | Reset |
| G | Kamera global |
| F | Kamera 3rd Person |

Walkthrough

Man startet in der Mitte der Map. Ziel des Spiels ist es die verführerisch nach Gänseblümchen duftende grüne Box (in der linken oberen Ecke) zu erreichen.

Geht man ein Stück nach vorne kann man rechts in einem schmalen Spalt ein Stück Kuchen entdecken. Angelockt davon bemerkt man schnell, dass man nach Verschlucken dieses Kuchens ein wenig zu dick für die Rückkehr ist. Zum Glück hat jemand daran gedacht eine Treppe zu installieren, die man sogleich erklimmen sollte. Oben angelangt hat man einen wundervollen Überblick über die Szene. Dabei sticht einem sofort die verführerisch grüne Box ins Auge, die einen geradezu magisch anzuziehen scheint. Gibt man dem Verlangen nach, und hüpf hinunter, sammelt man dabei eine Diättablette auf, die einen wieder schrumpfen lässt. Oje, die grüne Box ist wieder in unerreichbare Ferne gerückt. Dreht man sich zur Linken entdeckt man aber, dass in einer Nische unter der Treppe, wie auf einem Präsentierteller, ein saftiges Stück Kuchen herumliegt. Schnell schnappt man sich dieses und kann endlich die fantastische Box erreichen. Geschafft! Oje, ... die Forscher setzen einen zurück und Forschen munter weiter. Zum Glück warten wieder neue herrlich duftende Kuchenstücke auf unseren Hamster.

Implementierung der Anforderungen:

Gameplay

Wir haben mehrere Ebenen (bzw. Plattformen) eingefügt, auf denen der Hamster laufen kann. Die Collision Detection wurde mittels der Nvidia Physx Engine implementiert. Der Hamster kollidiert mit Wänden und Food-Objekten, letztere kann er sogar aufsammeln. Dabei wächst oder schrumpft er. Wird der Hamster kleiner so verringert sich auch seine Sprunghöhe. Wird er größer, kann er schneller laufen und auf höhere Ebenen springen.

Effects

| | | |
|------------------------|-----|--|
| Shadow Maps (with PCF) | 1.5 | As presented in the lectures, shadow maps calculate the shadows via rendering the scene from the light source. PCF samples the shadow map several times to improve quality. Make sure to counter all artifacts. |
| Spotlights | 0.5 | Limit the lit surface with cones (outer and inner) for a smooth transition. This also allows you to experiment with changing the light intensities. Goes well with any shadow technique. |
| Motion Blur | 1.5 | Motion blur is achieved by blurring along the moving direction of the objects. ATTENTION: Just using the Accumulationbuffer is not sufficient! |
| Projected Textures | 0.5 | A texture is projected onto geometry. This works similar to Shadow maps. |

- **Shadow Maps (with PCF)**

Alle Objekte werfen Schatten aufeinander. Dies wurde mittels 2-Pass-Shading implementiert. Zuerst wird im Depth Shader der Z-Buffer aus Sicht des Lichts in eine Textur gerendert und diese Textur im Shader ausgelesen und zur Berechnung der Sichtbarkeit der Fragments verwendet.

Mit Poisson Sampling wird die Textur mehrmals gesampled um die Schatten zu verfeinern.

Tutorial Link

www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/

- **Spotlights**

Wir haben ein Spotlight in der Scene platziert, welches auf einen Kuchen leuchtet. Dabei haben wir cones (inner und outer) verwendet.

Tutorial Link

http://www.ozone3d.net/tutorials/glsl_lighting_phong_p3.php

- **Motion Blur**

Wenn der Hamster 2 Mal gewachsen ist, ist er so schnell, dass Motion Blur aktiviert wird.

Tutorial Link

oglddev.atspace.co.uk/www/tutorial41/tutorial41.html

- **Projected Textures**

In die Mitte der Scene wird eine Textur auf unsere Ground Plane projiziert. Sie soll den Hamster an seinen natürlichen Lebensraum erinnern.

Tutorial Link

www.arcsynthesis.org/gltut/Texturing/Tut17%20Projective%20Texture.html

Complex Objects

Komplexe Model-Files können wie auch schon in der 1st Submission unseres Programms eingebunden werden. Mit einem Object Loader werden alle Objects aus .obj Dateien ins Programm geladen.

Sichtbar bei den Pillen. Die Objekte wurden alle in Blender selbst erstellt.

Animated Objects

Wenn der Hamster eine Pille aufsammelt, kreist diese um den Hamster und wandert mit ihm, wenn er sich bewegt. Es wird also eine hierarchische Animation verwendet.

View-Frustum-Culling

Transparency

Experimenting with OpenGL

Nicht Implementiert ... :/

Features

- Verschluckt der Hamster einen Stück Kuchen wächst er, bei einer Diätpillle schrumpft er.
- Die Objekte werden mittels .obj Dateien geladen und texturiert.
- Collision Detection mit Physx

Lichtquellen

1 Globale Beleuchtung

1 Spotlight

Texturierte Objekte

Hamster



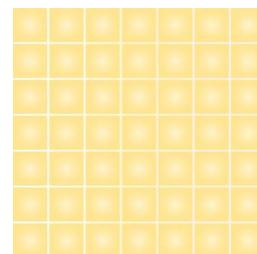
Kuchen



Diätpillen



Plane



Verwendete externe Bibliotheken

| | |
|-----------|---|
| GLFW | http://www.glfw.org/ |
| GLEW | http://glew.sourceforge.net/ |
| GLM | http://glm.g-truc.net/ |
| FreeImage | http://freeimage.sourceforge.net/ |
| Physx | https://developer.nvidia.com/physx-sdk |

Verwendete Tutorials

.obj Dateien laden:

<http://www.raywenderlich.com/48293/how-to-export-blender-models-to-opengl-es-part-1>