Documentation

Beyond the Black Rainbow

In diesem Spiel ist der Player in einem Raum eingesperrt, das Ziel ist die Tür zu öffnen und frei zu werden.

Während der Implementierung haben wir versucht einen gewöhnlichen Zimmer zu modellieren, unsere Objekte stellen Gegenständer vor, die man in jeder üblichen Haushalt finden kann.

Im Spiel sind Objekte zu finden wie:

- Bett
- Sessel
- Blume
- usw.

Sie werden im Programm selbst als einzelne Objekte realisiert (MeshNode) die alle (um einfach rendern zu können) VBO und VAO besitzen. Sie sind natürlich auch texturiert (die Texturen werden mit FreeImage geladen) und sind die Shaderoptionen auch einstellbar. Wie ein Bild zu texturieren geladen wird, ist die Sampling- und Mipmapgrad per default als höchste Qualität eingestellt, die kann man aber auch während des Spieles umstellen (F4, F5).
Neben einfacheren Objekten (wie Tisch, Schrank) werden auch etwas kompliziertere Polygonen benutzt (Enten, Blume). Um sie einfacher zu importieren haben wir ein Objectloadingsystem implementiert (assimp). Damit das Spiel das Alltagsleben besser darstellen kann scheinte es nötig zu sein die Grundregeln der Physik zu modellieren, im Spiel gielt die Gravitation (schau Enten) und Kollisionen mit und zwischen verschiedenen Objekten werden behandelt (PhysX). Damit die Spielszene lebendiger wirkt kann man rollende Enten beobachten, die miteinander und auch mit ihrer Umgebung interagieren. Jede von unseren Enten hat auch ein Pärchen, die einander im Weg immer begleiten.
Die Objekte besser organisieren zu können benutzen wir einen hierarchischen Scenegraph, jeder im Spiel geladenen Gegenstand (Kamera, Objekte, Spieler, Lichtquelle) muss einen Transformationsknoten (um die Bewegungen zu steuern) und einen Knoten (um den Objekt selbst zu steuern) besitzen.
Die Interaktionen, wie "die Tür muss sich öffnen" oder "Schüsselobjekt wurde gefunden", werden von unserem Eventsystem behandelt.

Grundsätzlich haben wir im Spiel drei verschiedene Effekte implementiert, um sie zu realisieren brauchten wir die Hilfe von FBOs.

- Omnidirectional Shadow Mapping

  Der Zimmer hat eine Lichtquelle (Pointlight) gleich vor der Tür, so müssen die Objekte auch Schatten werfen. Am Anfang wie die Enten noch höher sind, kann man die perspektivische Projektion gut beobachten wie die Schatten ihre Größen verändern, weiters ist es auch zu sehen, dass das Schattenbild von weiter entfernten Objekten verzerrter ist als von denen die sich in der Nähe von der Lichtquelle befinden. Neben dem Bett werden die Schatten

plötzlich heller, der Grund dafür ist eine andere Lichtquelle (Spotlight) den Bereich zwischen dem Bett und dem Tisch beleuchtet.

- Bloom
  Falls der Player den Ausweg vom Raum gefunden hat ändert sich die Beleuchtung ein bisschen. Diesen Effekt kann man am besten gleich in der Nähe der Tür beobachten, wie die Lichter heller scheint. Im Zimmer kann man aber auch sehen dass die Kanten unschärfer geworden sind.
- Depth of Field

  Wie der Spieler durch die Tür geht verändert sich die Szene wieder ein bisschen. In dem Fall wird den gerenderten Bild für Depth of Field gefiltert. Falls man diesen Effekt besser beobachten möchte, gibt es immer noch die Möglichkeit in den Zimmer zurückzukehren. Da kann man gut sehen wie die Objekte (oder Bilder an der Wand) in der Nähe scharf sind, weiter entfernte Gegenstände sind aber kaum erkennbar.

Um Bloom implementieren zu können braucht muss man natürlich auch die Alpha-Kanal verändern. Die Einstellungen des Spieles sind so implementiert dass man die Transparency vom Bloom nicht verändern kann, sonst würde dieser Effekt seinen Sinn verlieren.Es wird aber auch Text mitgerendert, damit die Buchstaben das Spielerlebnis nicht mehr als nötig stören, haben wir ihren Hintergrund durchsichtig geschrieben. Es kann aber vorkommen, dass es jemandem doch lieber wäre (wegen besserer Lesbarkeit) einen Hintergrund angezeigt zu haben, diese Einstellung kann man jeder Zeit im Spiel ändern (F9).

WireFrame-Modus ist mit Hilfe OpenGL-Methoden implementiert, man kann zwischen normalen und Wireframe Ansicht jeder Zeit wechselcn bis auf den Schlüsselgegenstand nicht gefunden wurde.

Im Code haben wir eigentlich drei verschiedene Lichtquellen implementiert (Pointlight, Directionallight und Spotlight) später haben wir aber so überlegt, es ist besser nur Pointlight und Spotlight zu benutzen, weil sie in einem inneren Raum natürlicher wirken. Dise Lichter schaten sich automatisch ein und aus wenn der Spieler Räumlichkeit wechselt.

- Pointlight:

  Gleich vor dem Spieler (Anfangsposition) etwa so hoch wie die obere Rahmen der Tür

  Zwischen den Vasen, genauso hoch wie der anderen Lichtquelle

- Spotlight

  Zwischen dem Tisch und Bett, wir auf kleineren Bereich fokussiert

  Zwischen den beiden Zimmer (Farbe: Türkis)

  Bei der Ecke des Schrankes (Farbe: Rosa)

  (Diese Lichtquellen Beleuchten immer das Boden)

Die geladene Objekte haben mit verschiedenen Softwares gemacht (großteils Blender, einige mit Maya) und viele haben wir vom Internet untergeladen oder von einer Kollegin bekommen. (Bettina Schlager, vielen Dank für ihre Arbeit).

Spielanleitung:

Der Player kann sich mit den Tasten WASD bewegen mit Space springen und mit der Maus die Blickrichtung ändern. Falls man mit einem Objekt interagieren möchte muss man die E Taste benutzen. Auf eineigen Objekten kann man auftreten, anderen aufspringen die größeren sind aber dafür nicht geeignet. Das Ziel ist den Schlüsselgegenstand zu finden, der die Tür automatisch öffnet.

(**Spoiler:** *man muss auf den Bett klicken*)

Libraries:

GLEW

http://glew.sourceforge.net/

GLFW

http://glew.sourceforge.net/

Assimp

http://assimp.sourceforge.net/main_doc.html

FreeImage

http://freeimage.sourceforge.net/

PhysX

https://developer.nvidia.com/gameworks-physx-overview

Quellen:

http://learnopengl.com/ *(Framebuffers, Shadowmap)*

http://www.opengl-tutorial.org/ *(first steps, Text)*

http://resources.blogscopia.com/ *(Blender Objekte)*

Bettina Schlager *(Maya Objekte)*

Es tut mir Leid für meine Deutsch... Ich habe mir die Mühe gegeben!