Bandicoot Run

Steuerung im Spiel-Modus

Aktion	Taste	Alternative Taste
Vorwärts laufen	W	Oben
Nach links laufen	A	Links
Rückwärts laufen	S	Unten
Nach rechts laufen	D	Rechts
Springen	Leertaste	
World-Movement stoppen	O	
Wireframe-Modus für den Felsbrocken EIN / AUS	F3	
Bloom EIN / AUS	F7	
Inspector Modus	I	
Spiel beenden	ESC	

Steuerung im Inspector-Mode

Aktion	Taste	Alternative Taste
Vorwärts fliegen	W	Oben
Nach links fliegen	A	Links
Rückwärts fliegen	S	Unten
Nach rechts fliegen	D	Rechts
Abwärts fliegen	STRG (Links)	
Aufwärts fliegen	ALT (Links)	
Licht entlang der Map vorwärts bewegen	J	
Licht entlang der Map rückwärts bewegen	G	
Licht nach oben bewegen	Z	
Licht nach unten bewegen	H	
Licht quer zur Map nach links bewegen	T	
Licht quer zur Map nach rechts bewegen	U	
World-Movement stoppen	O	
Wireframe-Modus für den Felsbrocken EIN / AUS	F3	
Bloom EIN / AUS	F7	
View Frustum Culling von “Game-Cam” auf “Inspector-Cam” umschalten”	F8	
Normaler Spiel-Modus	I	
Spiel beenden	ESC	

Implementierung

Der Game-Loop besteht grundsätzlich aus folgenden Teilabschnitten:

- Initialisierung der Shader, 3d Objekte, Texturen, Framebuffer und bspw. Variablen zur Berechnung der Zeit zwischen den Frames (damit die Spiel-Geschwindigkeit frame-unabhängig ist).
- Eingaben des Spielers entgegennehmen
- Aktualisierung der Modelle (Position, Camera mit einberechnen) zusammen mit Physik-Simulation
- Zeichnen aller als “sichtbar” gekennzeichneten Modelle

Features

- Lustiges Verfolgungs-Spiel aus einer ganz einzigartigen Perspektive
- „Verkehrte“ 3rd-Person-Perspektive
- Verschiedenste Hindernisse, die es zu überwinden gilt
- Unterschiedlichste Effekte auf den Objektoberflächen
- Volle 3D-Kollisionserkennung

Beleuchtung

In unserem Spiel gibt es nur eine globale Punktlichtquelle über der „Welt“, die alle Objekte beleuchtet: Die Sonne. Da auf den verschiedenen Objekten allerdings unterschiedliche Schader zum Einsatz kommen, werden wird das Licht von den Oberflächen natürlich unterschiedlich verarbeitet und reflektiert. Die Lichtquelle wird im normalen Spielmodus mit dem Bandicoot mit bewegt, lässt sich aber im „Inspektor“ Modus auch frei im Raum bewegen, wodurch Effekte wie das Normal-Mapping und das Cel-Shading gut sichtbar gemacht werden können. Auf jedem Objekt (mit Ausnahme des Bandicoots und der Duckies, die den Cel-Shader benutzen) sind aber sowohl Ambient-Light als auch Specular-Highlights sichtbar.

Effekte

Um etwas Kontrast zu dem doch recht einfachen Spielprinzip von „Bandicoot Run“ zu bieten, haben wir uns dazu entschlossen, auf den Verschiedenen Objekttypen unterschiedliche Effekte zu realisieren. Dadurch wird aus unserer lokal durchaus stark beschränkten Spiele-Welt, ein regelrechtes Effekt-Feuerwerk, dass einfach Spaß mach anzusehen und das Spiel zu spielen.

NormalMapping

Durch das Normal-Mapping erscheinen eigentlich plane Oberflächen plastisch, indem sogenannte „Normal-Maps“ zusätzlich zur eigentlichen Textur zum Einsatz kommen. Dieser Effekt ist besonders schön an dem großen Felsbrocken zu erkennen, die den Avatar verfolgt. Zusätzlich wurde der Effekt auf den rollenden Zahnrädern am Level-Anfang sowie auf den Baumstämmen und den Plattformen aufgebracht, welche dadurch ebenfalls eine sehr schöne und deutliche Struktur bekommen. Besonders deutlich tritt dieser Effekt zum Vorschein, wenn man die Lichtquelle im „Inspector“ Modus

recht knapp an einem entsprechenden Objekt – wie beispielsweise dem großen Felsbrocken – vorbeibewegt.

Cel-Shading (inclusive Backfaces)

Bei diesem Effekt wird kein konstanter Lichtabfall bei Beleuchtungen dargestellt, sondern vielmehr ein Flächen-Effekt erzeugt. Der Hauptcharakter unseres Spieles – unser Bandicoot – wird mittels Cel-Shading dargestellt und zusätzlich haben wir diesen Comic-Stil auch für die gelben Gummi-Enten, die einige der Hindernisse in unserem Spiel darstellen, angewandt. Auch diesen Effekt sieht man am besten, wenn man die vorhandene Lichtquelle herumbewegt. Wird der Bandicoot beispielsweide von schräg vorne angeleichtet, so sieht man hervorragend die entstehenden Helligkeits-Flächen.

Als besondere Erweiterung werden die Kanten bzw. Umrisse der Objekte durch den Einsatz der Backfaces visualisiert, indem sie schwarz nachgezogen werden. Dieser Comicartigen Stil passt optimal zu unserer Spiele-Idee, und trägt so erheblich zum Spielspaß bei.

Bloom

Die Szene wird mittels Framebuffer auf eine Textur gezeichnet. Diese wird anschließend mittels eines Shaders gefiltert auf einem Cube abgebildet und so eingestellt, dass sie den gesamten Screen füllt. Beim Filtervorgang werden zuerst die hellsten Pixel mittels eines Schwellwertes isoliert, mit zwei 1-dimensionalen Gauss-Kernels gefaltete und anschließen auf den Original-Frame addiert. Da dieser Effekt einige andere schwieriger zu sehen macht, kann man ihn in unserem Spiel mit einem Druck auf F7 einfach EIN und AUS schalten.

ShadowMapping

Mit Hilfe eines Framebuffers und eines Shaders werden aus Sicht einer Lichtquelle die Tiefenwerte der Vertex auf eine Textur gezeichnet (ShadowMap). Danach wird die Szene normal gerendert und jene Pixel, deren z-Wert kleiner als der entsprechende Tiefenwert auf der Shadowmap ist, dünkler (im Schatten) dargestellt.

ACHTUNG – Dieser Effekt scheint nur auf AMD-Grafikkarten zu laufen!

Andere Special-Features

Rudimentäre Skybox

Obwohl der Himmel bei uns nur in einem sehr kleinen Spalt zu sehen ist, der Entsteht wenn man sich weit vom Startpunkt entfernt, haben wir einen großen Würfel eingebaut, der von Innen über eine UV-Map mit Texturen versehen ist. Dadurch entsteht eine rudimentäre Form einer Skybox, welche den Eindruck eines vorhandenen Himmels entstehen lässt.

Config-File

Die von uns verwendete Config-File namens „ScreenRes.ini“, erlaubt ein Umschalten zwischen dem Fenster- und dem Vollbild Modus, sowie eine exakte Angabe der gewünschten Auflösung. Ist der Fenstermodus gewählt (also ein 0er in der ersten Zeile), dann wird das Fenster exakt so groß erstellt, wie die angegeben Auflösung die in der Config-file steht (Zeilen 2 und 3). Ist der Vollbildmodus aktiv (1er in erster Zeile), so wird die Bildschirmauflösung auf die angegeben Auflösung umgeändert, und das Spiel am ganzen Bildschirm dargestellt.

Verwendete Tools für Modelle

Abgesehen von dem Model des Bandicoots, wurde alle Objekte komplett in 3ds max erstellt, bearbeitet und mit UV-Maps versehen. Der Bandicoot-Avatar wurde über die Google Modellsuche bezogen (Link siehe unten) und war ursprünglich in einer „Hampelmann-Stellung“. Da diese natürlich für einen laufenden Avatar ungünstig ist, und wir anfangs Skeleton-Animation implementieren wollten, habe ich das Modell etwas bearbeitet und mit Bones versehen sowie diese mittels einer „Haut“ an das Model gebunden. Leider haben wir die Skeleton-Animation nicht zum Laufen gebracht (das exportierte Modell samt Bones ist aber noch bei den Assets), konnten aber durch das nun bereits mit Bones versehene Modell einfach die Posen des Charakters darstellen, die gewünscht waren. Die restlichen Objekte sind einfache Grundkörper, die leicht mit 3ds max erstellt werden konnten (Kugel, Zylinder und Box). Um die Oberflächen korrekt mit Texturen versehen zu können, wurden die meißten Objekte (Boden-Boxen, Skybox, Baumstämme, Plattformen, Zahnräder, der Felsbrocken und der Bandicoot) mit UV-Maps versehen.

Externe Bibliotheken

Wir verwenden externe Bibliotheken überall dort, wo dies auf den entsprechenden Seiten der LVA-Homepage empfohlen wird. Auch haben wir uns an die dort vorgeschlagenen Bibliotheken gehalten:

- GLFW – Window Handling
  http://www.glfw.org/
- GLEW – Extensions
  http://glew.sourceforge.net/
- GLM – Mathematic Library
  http://glm.g-truc.net/
- Bullet Physics – Kollisionserkennung und Physik-Engine
  http://bulletphysics.org/wordpress/
- FreeImage – Import von Bild-Dateien für die Realisierung der Texturen
  http://freeimage.sourceforge.net/
- AssImp – Laden von externen Modellen
  http://assimp.sourceforge.net/

Quellen und andere Links

- Grundlegender Anstoß zu unserer Spieleidee war eine Bonuslevel aus dem Klassiker „Crash Bandicoot“ auf der Sony PlayStation (One)
  http://de.wikipedia.org/wiki/Crash_Bandicoot

- Grundmodell des Bandicoots
  http://sketchup.google.com/3dwarehouse/details?mid=71aac959025747ebde17cfeb6d48afb9&prevstart=0