Bananarama

Echtzeitgrafik, 2. Abgabe

Wolfgang Berger, 0225152 / 932
Matthias Buchetics, 0225149 / 932

Effekte

Wasser

Die Landschaft sowie alle anderen 3D Objekte werden gespiegelt in eine Reflectiontextur gerendert, welche als projektive Textur für das Wasser verwendet wird. Über Perlin Noise (vorberechnet auf der CPU) werden die großen Wellen realisiert, über eine Normalmap die kleinen. Diese Wellen bzw. deren Normalen werden beim Zugriff auf die Wellentextur verwendet.

Weiters wird der Himmel über eine Cubemap gespiegelt und so – zusammen mit einer festlegbaren Grundfarbe – die Wasserfarbe ermittelt. Für die Spiegelung der Sonne wird der Specular Term des Phong Shadings hergenommen, welcher ebenfalls im Pixelshader berechnet wird und die Farbe der Sonne erhält.
Auch das Terrain unter dem Wasser ist teilweise noch zu sehen und wird entsprechend der Wassertiefe ausgeblendet.

Referenzen:

- GameDev.net: Realistic Natural Effects Rendering: Water I (http://www.gamedev.net/reference/articles/article2138.asp)
- ShaderX: Rippling Reflective and Refractive Water (http://www.ati.com/developer/shaderx/ShaderX_RipplingRefractiveAndReflectiveWater.pdf)
- Claes Johanson: Real-time water rendering - introducing the projected grid concept (http://graphics.cs.lth.se/theses/projects/projgrid/)

Terrain Texturierung in Echtzeit

Die Texturierung des Terrains erfolgt vollständig im Pixelshader, was eine dynamische Änderung der Texturen und ihrer Parameter ermöglicht. Beispiel: Schneefall, bei dem sich die Schneegrenze nach unten verschiebt. Als Parameter steht auch die Steigung des Terrains zur Verfügung.
In der implementierten Demo wird eine dynamische Verschiebung der Texturgrenzen jedoch nicht verwendet.

PRT

Die Schattenwürfe der gesamten Landschaft werden mittels Precomputed Radiance Transfer erzeugt. Der dazu notwendige, selbstimplementierte Transfersimulator berechnet die Transfervektoren für direkte, monochrome Beleuchtung pro Vertex. Die Ergebnisse werden dann abgespeichert, damit dieser langwierige Schritt nur einmal durchgeführt werden muss.

Zur Laufzeit gibt es zwei Lichtquellen (Tag/Nacht), die abhängig von der Tageszeit entsprechend geblendet, rotiert und eingefärbt werden. Mit diesen wird dann im Vertexshader die Ausleuchtung der Szene berechnet.

Um beim Rendern der Landschaft nicht in den High Dynamic Range Bereich zu gelangen, bzw. die Szene nicht zu dunkel ausfallen zu lassen, werden die beiden erzeugten Lichtquellen zu Beginn normalisiert, was über die Berechnung der Helligkeit eines der Lichtquelle direkt zugewandten Vertex geschieht. Danach wird ein Faktor berechnet, welcher diese Helligkeit auf 1 skaliert.

Weiters wird danach noch ein Ambient-/Schattenwert berechnet, der über die Helligkeit eines von der (skalierten) Lichtquelle abgewandten Vertex bestimmt wird. Dieser wird dann für die Ausleuchtung der gouraud-schattierten Objekte sowie für die Schattenwürfe verwendet, damit ein einheitlicher Eindruck entsteht.

Referenzen:

- Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments (http://research.microsoft.com/~ppsloan/shillum_final23.pdf)
- Spherical Harmonic Lighting: The Gritty Details (http://www.research.scea.com/gdc2003/spherical-harmonic-lighting.pdf)

Skydome

Abhängig von der Tageszeit wird der Skydome eingefärbt und die Sonne positioniert. Die Farben werden aus Texturen ausgelesen und für die Lichtfarbe beim PRT verwendet. Außerdem stellt er noch die Rotationswinkel der Lichtquellen, einen Lichtvektor und die für das Wasser benötigte Cubemap zur Verfügung. Der Skydome ist damit der Ausgangspunkt für sämtliche anderen Licht- und Schattenberechnungen.

Referenzen:

- A simple model for fast, realistic sky dome color rendering (http://www.geocities.com/ngdash/whitepapers/skydomecolor.html)

Shadowmapping

Alle Objekte auf der Insel werfen mittels Shadowmapping auf das Terrain Schatten. Dabei werden ihre Tiefen aus der Sicht des Lichtes mittel Orthogonalprojektion in eine Depthmap gerendert, welche dann bei der Terraintexturierung ausgewertet wird. Spezielle Optimierungen wurden nicht eingesetzt, das View-Frustum wird immer so definiert, dass die gesamte Insel darin Platz findet. Dadurch ist die Qualität der Schatten auch von der Terraingröße abhängig. Um das Kantenflimmern ein wenig abzuschwächen, wurde 2 x 2 PCF verwendet.

Referenzen:

- Wolfgang Engel: Programming Vertex and Pixel Shaders, Shadow Mapping (S. 287 ff)

Blooming

Um die karibische Stimmung besser einfangen zu können haben wir als Post-Process-Effekt Blooming einführt. Dabei wird die gesamte Szene zuerst in eine 32-Bit RGBA Textur gerendert, dann in eine fp16 Surface mit einem Viertel der Originalgröße herunterskaliert, mit einem horizontalen und vertikalen Gauss-Filter belegt und wieder aufskaliert. Schließlich wird das Ergebnis mit dem ursprünglichen Rendering kombiniert. Durch Multiplikation mit einem bestimmten Exposure-Value wird nun das gesamte Bild in den High Dynamic Range gebracht und durch einen fixen Exponenten wieder in den Low Dynamic Range skaliert (Gamma-Korrektur).

Referenzen:

- Wolfgang Engel: Programming Vertex and Pixel Shaders, High Dynamic Range Lighting (S. 257 ff)

Weitere Besonderheiten

Kamerasteuerung

Die automatisierte Kamerasteuerung wird über Catmull-Rom Splines realisiert.

XML-Anbindung

Nahezu alle Parameter lassen sich von außen über entsprechende XML-Dateien verändern. Das Ein- und Auslesen dieser wurde dabei mit C# internen Bibliotheken realisiert.

Manuelle Steuerung

Mit der Taste „c“ kann die automatische Kamerafahrt abgebrochen werden. Danach lässt sich die Kamera über die Maus + den Tasten w, a, s, d sowie q und e manuell bewegen.

Verwendete Bibliotheken

- DirectX SDK 9 (August 2005)
- .NET 1.1
- FMOD 4