VU Echtzeitgrafik, 2. Abgabe

Projekt: TUI

The Unknown Island

Michael Rebec, 0125590, 066 932
Reinhard Ruß, 0605016, 033 534

Szene

An unserem Setup hat sich seit der ersten Abgabe nicht viel geändert. Statt dem Baum befindet sich jetzt eine Palme auf der Insel und im Haus kamen ein paar Items dazu.
Die gesamte Szene wurde in Maya modelliert. Die Modelle sind entweder von Web-Seiten mit Gratis-Lizenz oder wurden selbst modelliert. Die Kamerafahrt wurde ebenfalls in Maya modelliert und dann als „anim“-File exportiert. Wir haben uns dann noch ein kleines Tool geschrieben, dass die Keyframes aus dem „anim“-File ausliest und die Translations-/Rotationswerte linear interpoliert (über die Zeit). Damit es ein wenig stimmungsvoller wird, haben wir auch noch Sound integriert.

Gamma-Korrektur

Wir verwenden sRGB-Texturen, wie in diesem Tutorial beschrieben:
http://http.developer.nvidia.com/GPUGems3/gpugems3_ch24.html

Effekte

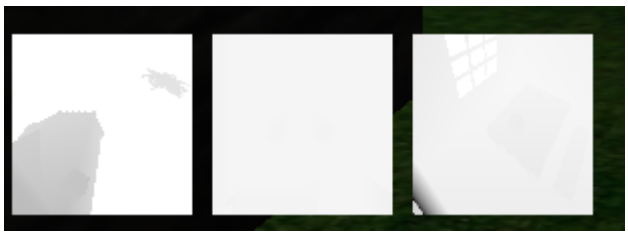
- Schatten (LiSPSM) und Lichtquellen

  In unserer Szene gibt es insgesamt drei Lichtquellen:

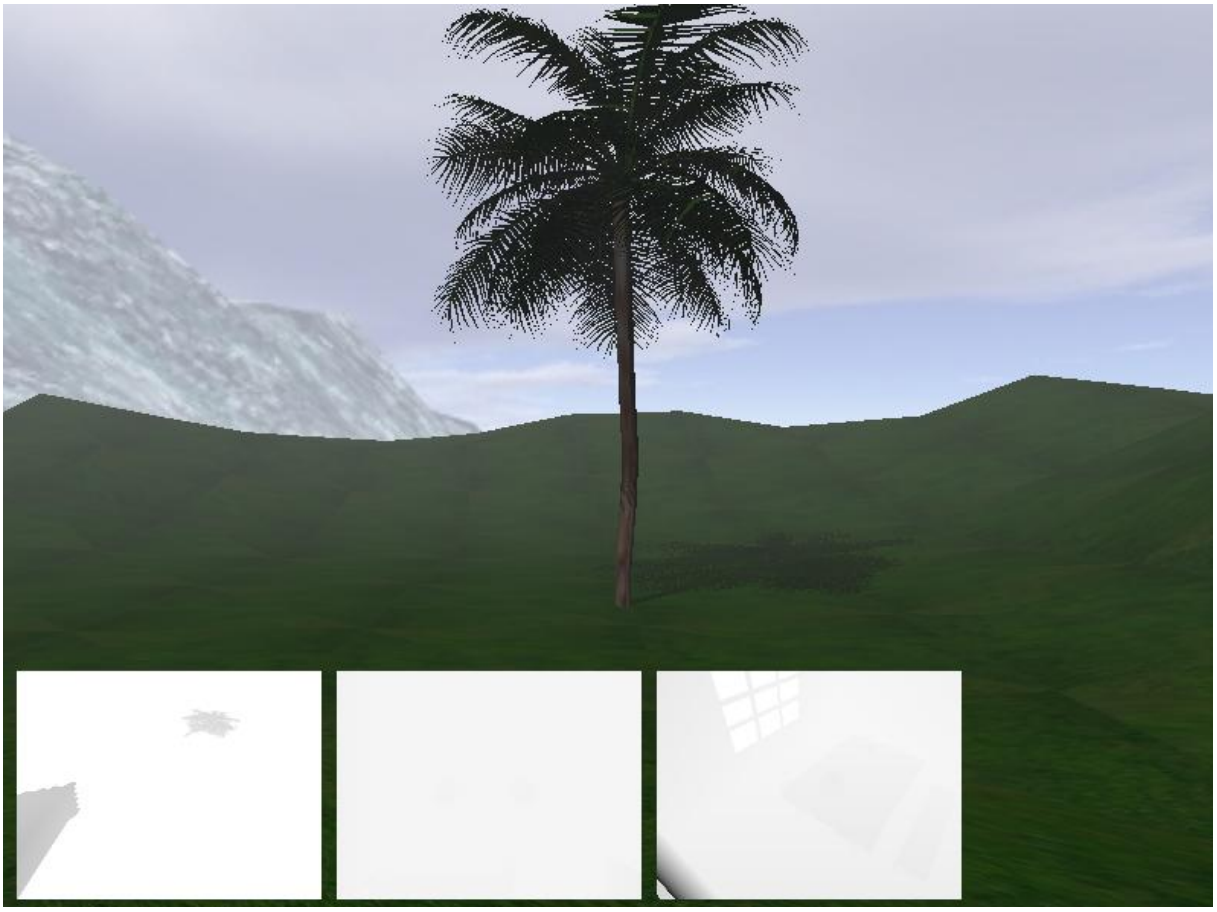
  1) Sonne (Direktionales Licht)
  2) Tischlampe (Spotlight)

3) Lampenschirm (Spotlight mit Light-Scattering)

Für jede Lichtquelle wird eine Shadow-Map erstellt, welche man sich mit der Taste F2 ansehen kann.



Die erste Shadow-Map ist für das direktionale Licht und verwendet die LiSPSM-Methode (Quelle: http://www.cg.tuwien.ac.at/research/vr/lispsm/). Demzufolge wird diese Shadow-Map für jedes Frame neu erstellt (Shadow-Map ist von der Kamera abhängig, etc). Wir haben diese Methode gewählt, weil unser Terrain ziemlich groß ist und Uniform Shadow Mapping keine guten Ergebnisse geliefert hat. In der Debug-Ansicht sieht man schön, wie sich die Shadow-Map mit der Kamera mitbewegt.



Die Shadow-Map in der Mitte und die Shadow-Map ganz rechts, sind für die beiden Spotlights in der Szene. Nachdem diese beiden Lichtquellen nur einen kleinen Bereich abdecken, haben wir für diese beiden Uniform Shadow Mapping verwendet.

Die untschiedlichen Lichtquellen und Schatten werden dann in den Shadern aufsummiert – wir verwenden dafür ein Texture-Array. Hier werden beispielsweise der Schatten von der Tischlampe (Tisch) und der Sonne (Licht durch das Fenster) miteinander kombiniert:

Der Lampenschirm hat zusätzlich einen Light-Scattering-Effekt, welcher im Pixel-Shader implementiert ist. Folgende Quelle wurde verwendet: http://mmack.wordpress.com/2010/05/29/in-scattering-demo/

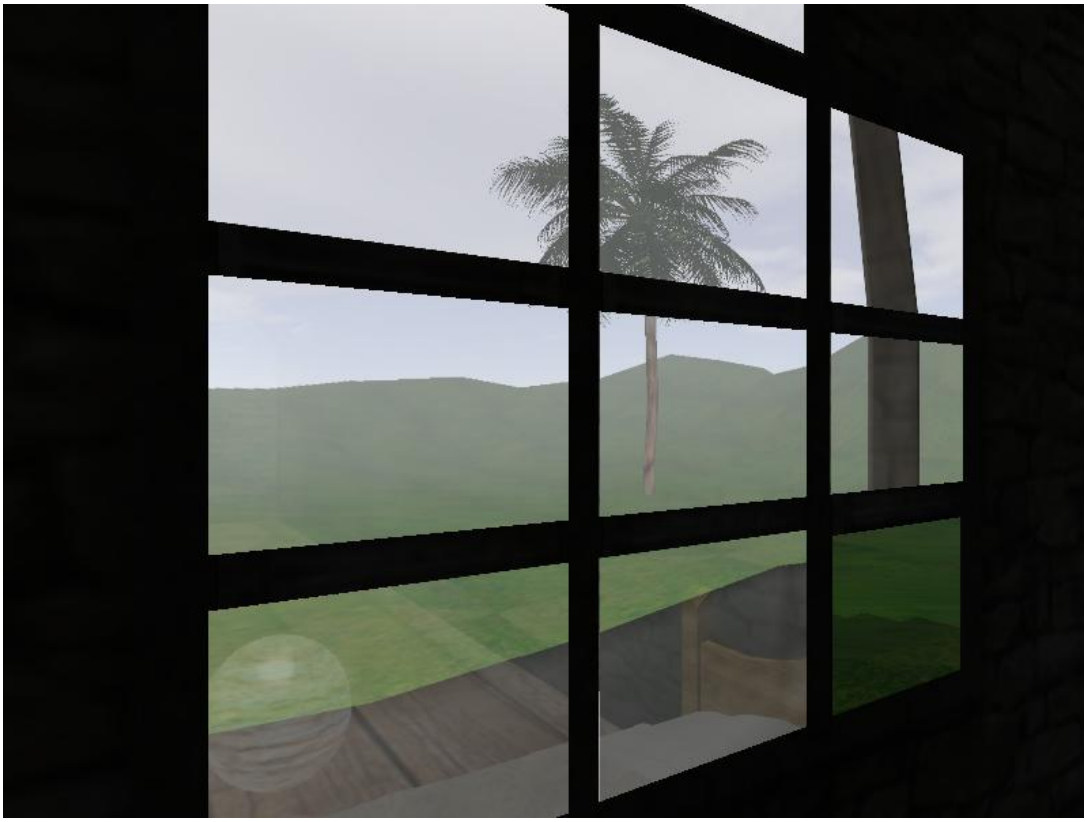
Für die Tischlampe wollten wir das Light-Scattering als Post-Process (http://http.developer.nvidia.com/GPUGems3/gpugems3_ch13.html) implementieren, weil mit dieser Technik wohl das Durchdringen des Lichtes durch die Fenster am Besten ausgesehen hätte. Nachdem sich dieser Effekt nicht mehr ausgegangen ist und das Light-Scattering vom Lampenschirm, bei der Tischlampe unrealistisch ausgesehen hat, ist diese Lichtquelle ein einfaches Spotlight ohne Light-Scattering-Effekt.

- Reflective Environment Mapping

Die Reflective Environment Maps (Cube maps) werden zu Beginn der Szene für die spiegelnden Objekte (Löffel und Kugel am Tisch) erstellt.

- Refractive Environment Mapping

Bei den Refractive Environment Maps wird prinzipiell dasselbe Prinzip wie vorhin angewendet. Anwendung finden diese ben den Fenstern des Hauses und dem Glas auf dem Tisch. Zusätzlich wird bei den Fenstern noch die „Nebeleffekt“-Berechnung durchgeführt, da es gleich aussieht wie das Haus, wenn man weiter weg ist. Für die Präsentation haben wir noch den Fresnel Term für Environment Refractions hinzugefügt und das Environment Mapping an den Fensterscheiben implementiert.



Quelle: http://http.developer.nvidia.com/CgTutorial/cg_tutorial_chapter07.html

- Fog

Dieser atmosphärische Effekt stand auf unserer Liste. Es hat sich jedoch herausgestellt, dass diese sehr einfach im Pixel-Shader zu implementieren ist und die Wirkung der Beleuchtung der Szene eher negativ beeinflusst. Aus diesem Grund wird dieser Effekt nur ganz leicht angewendet.

- Bump Mapping

Das Bump Mapping hat bei uns schon mit einer direktionalen Lichtquelle funktioniert. Die Kombination aus mehreren Spotlights und einer direktionalen Lichtquelle hat leider nicht mehr funktioniert, deshalb ist das Bump Mapping derzeit auskommentiert.

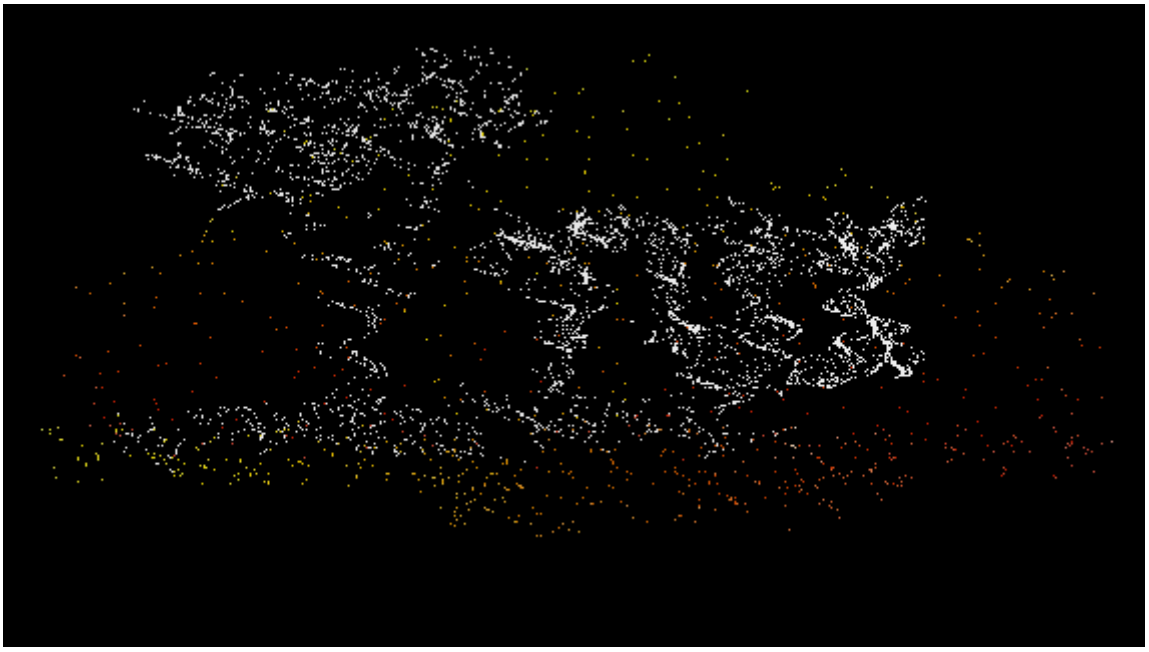
Quelle: http://fabiensanglard.net/bumpMapping/index.php

- Wasser

Der Wasser-Effekt ist eine Mischung aus Reflective Environment Mapping und zwei verschachtelten Sinusfunktionen, welche die Pixel im Vertex-Shader verschieben und dadurch die Wellenbewegung simulieren.

- Partikelsystem

Dieser Effekt stand eigentlich nicht auf unserer Liste, aber er sieht cool aus ;) Es handelt sich um einen reinen GLSL-Effekt. Es wird ein Mesh, welches sehr viele Vertices enthält im ersten Aufruf mit GL_TRIANGLES gerendert. Danach werden dynamische Werte an den Vertex-Shader übergeben und es werden GL_POINTS gezeichnet. Die Vertices werden mit Hilfe einer Noise-Funktion in alle Richtungen verschoben und erzeugen so diesen Effekt. Außerdem wird noch die Lebensdauer abhängig vom Vertices-Index berücksichtigt. Anwendung findet dieser Effekt beim Lade-Screen zu Beginn und beim Credit-Screen am Ende.



Verwendete Libraries

- Assimp: Zum Laden der Modelle
- Fmod: Sound-library
- Freetype: Darstellung von Text
- Glew: OpenGL Window handling
- Glfw: Key/Time handling
- Glm: Vektoren, Matrizen, etc.
- Sdl: Laden der Texturen

Sonstiges

Die Kamerasteuerung und die Auflösung kann im Config-File eingestellt werden.