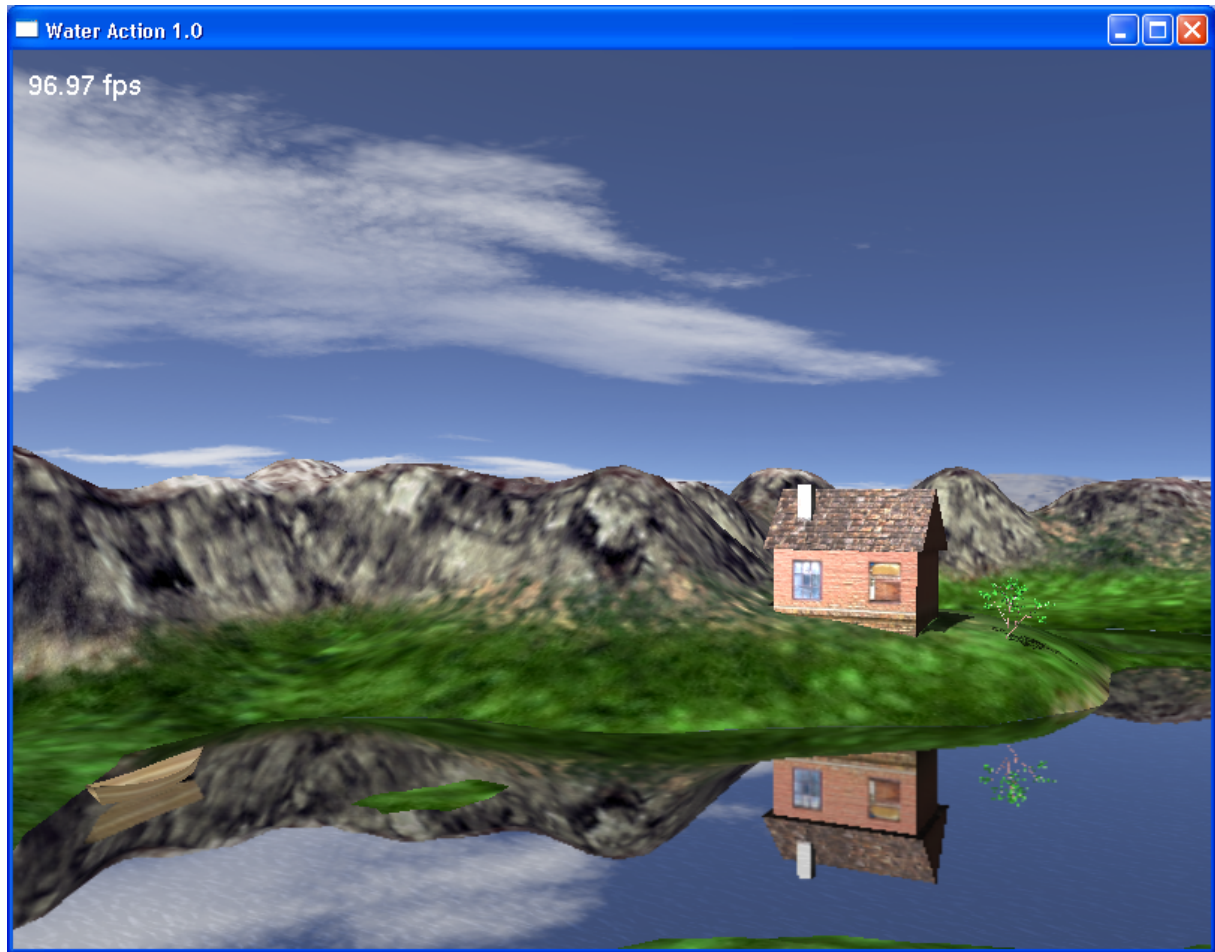


Water Action



Harald Meyer (932, 0225448)
Tobias Schleser (932, 0225349)

Bump Mapping

Beschreibung:

Beim Bump Mapping wird versucht, Oberflächen von Objekten interessanter (detaillierter) zu machen, ohne dabei die Komplexität (Anzahl der Polygone) des Objekts vergrößern zu müssen.

Dazu wird – entsprechend einer „Normalmap“ – an jedem Oberflächenpunkt eines Polygons nicht der Normalvektor genommen, welchen man aus den Normalvektoren aller Vertices des Polygons errechnet, sondern ein anderer. Die Lichtberechnung wird dann per Pixel durchgeführt und so lassen sich raue bzw. Oberflächen mit Struktur darstellen.



Abbildung 1: Teekanne mit Bumpmapping

Methode:

Das Bumpmapping wurde mittels Shaderprogrammierung durchgeführt. Im Vertexshader wurde die TBN Matrix erstellt und der light- sowie view-Vector damit in den TBN Space transformiert.

Der TBN Space definiert sich über Tangente, Binormale und Normale auf der Normalmap und ist ein dreidimensionaler, kartesischer Raum.

In „Abbildung 2: Normalmap mit TBN Koordinatensystem“ ist eine Normalmap dargestellt, inklusive der Achsen in der richtigen Farbe. Soll der Normalvektor an einer Stelle z.B. „nach oben“ (in positiver y-Richtung) schauen, so wird der entsprechende Pixel vermehrt Grünanteile haben (an den Ziegeln oben sichtbar).

Da die Grundrichtung aber selbstverständlich „dem Leser entgegen“, also in positiver z-Richtung bleiben soll, ist der Gesamteindruck des Bildes bläulich (da an den meisten Pixeln der Blauanteil aus o.g. Grund am größten ist).

Außer der TBN Transformation wurden im Vertexshader noch die Texturkoordinaten weitergereicht.

Im Pixelshader wurden zunächst mit Hilfe der Texturkoordinaten die Farbwerte eines konkreten Bildpunktes ausgelesen und gleichzeitig die (r,g,b) Werte aus der Normalmap an der selben Stelle ermittelt. Da die RGB-Werte aus der Normalmap zwischen 0 und 1 liegen, mussten diese noch auf den Bereich -0,5 bis 0,5 gemappt werden.

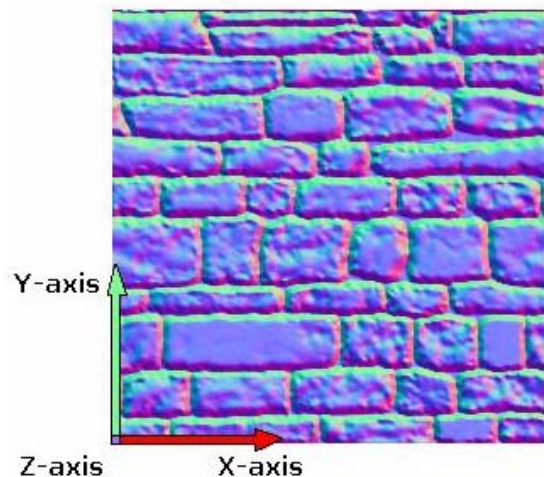


Abbildung 2: Normalmap mit TBN Koordinatensystem

Danach wurde eine normale per-Pixel Lichtberechnung durchgeführt, wobei die RGB-Werte der Normalmap – interpretiert als (x,y,z) eines Richtungsvektors – als Normalvektor dienten. Am Ende wurden die ermittelten Summanden aus ambientem, diffusen und spekulären Licht zu einer Endfarbe für den Pixel zusammengerechnet.

Shadow Mapping

Beschreibung und Methode

Beim Shadow Mapping wird die Szene (mind.) 2 Mal gerendert. Einmal von der Lichtposition aus, und einmal vom Viewpoint. Bei Water Action wird der erste Pass (Shadowmap) in einen PBuffer gerendert (da hiervon nur die Tiefenwerte interessant sind).

Die Pixel aus dem Viewpositionpass werden durch Multiplikation mit der Biasmatrix, der Lightprojection, der Lightview und der inversen Modelviewmatrix in den Space der Lichtquelle transformiert (Camera's Eyespace → Lightspace).

Durch Vergleich der z-Werte der Shadow Map mit den transformierten z-Werten kann festgestellt werden, ob der Punkt im Schatten liegt – ist der z-Wert der Shadow Map kleiner als der des transformierten Pixels, so liegt er im Schatten.

In Water Action ist Shadow Mapping über die ARB_Shadow Funktion implementiert.

Anmerkungen

Das Haus und das Wasser sind jeweils gebumpmapped. Das Wasser basiert auf dem Tutorial von Bonzaisoft [6] wurde aber komplett (bis auf die Projektion der Reflection Map) neu geschrieben.

Wir haben zum Vergleich auch den Originalcode von Bonzaisoft integriert. Dieser kann durch Setzen des Flags „g_showGameTutWater“ in „gameengine/GameEngine.cpp“ auf „true“ aktiviert werden und befindet sich in den Files „WaterFX.*“, sowie „data/water/*.*“.

Der Code der Matrixklasse und des PBuffers basiert auf Quellen von [10].

Der MS3D Loader basiert auf [11] wurde aber zu großen Teilen neu geschrieben („Vertex Arrays“).

Quellen:

- [1] http://www.blacksmith-studios.dk/projects/downloads/bumpmapping_using_cg.php
- [2] <http://www.ozone3d.net>
- [3] <http://3dshaders.com/shaderSource.html>
- [4] http://gpwiki.org/index.php/OpenGL:Tutorials:GLSL_Bump_Mapping
- [5] <http://www.paulsprojects.net/tutorials/simplebump/simplebump.html>
- [6] http://www.bonzaisoftware.com/water_tut.html
- [7] http://www.cg.tuwien.ac.at/courses/Realtime/slides/09shadows_rtr2004.pdf
- [8] <http://www.paulsprojects.net/tutorials/smt/smt.html>
- [9] <http://www.turbosquid.com/>
- [10] <http://developer.nvidia.com/page/home.html>
- [11] <http://rsn.gamedev.net>