MooSlider

CG 2/3 Übung SS 2004

Altschach, Marion, 532, 0225184, altschach_marion@gmx.at
Kogelnig, Philipp, 535, 0225185, kogelnig_philipp@gmx.at
Kogelnig, Richard, 535, 0326319, Goblin@gmx.at

Spieldesign

In der Mittelpunkt des Spieldesigns, steht die Vorstellung von Wichteln, die auf einer umgedrehten Kuh einen Berg hinunterrasen. Die ganze Idee des Moosliderspiels ist nun schon mehrere Jahre alt- endlich ist es uns gelungen der ursprünglichen Eingebung gerecht zu werden.
Das Szenario bietet einen guten Hintergrund für ein „Funracing“-Spiel.

Bei der Umsetzung wurde untere andrem Wert auf Soundeffekte und Musik gelegt, um das Spielerlebniss abwechslungsreicher zu gestalten, desweiteren sollte die surreale Situation zum Schmunzeln anregen.

Umgesetzt wurden mehrere Spielmodi:
- Single Player
- Single Player (Ghost Mode)
- SplitScreen (1vs1)

Allgemein:
Mit dem Gefährt sollte sanft umgegangen werden, da die Kuh sonst aufwacht, wird zum Beispiel ein Baum getroffen, so verringert sich Balken für die Schlafanzeige auf der Rechten Seite (dargestellt durch ein Bett mit den typischen ZZZ darüber).
Je kürzer der Balken, desto weniger Schaden kann die Kuh noch nehmen bevor sie aufwacht und die auf ihr sitzenden Wichtel verprügelt. Desweiteren ändert sich die Farbe von grün (alles ok) über gelb (langsam wird’s gefährlich) zu rot (Achtung, die Kuh wacht gleich auf).
Desweiteren muss in allen Modi das Ziel erreicht werden bevor eine gegebene Zeit verstreicht. Durch das Aufsammeln von Flaschen, welche mit Bier gefüllt sind, kann man mehr Zeit herausschinden (anscheinend sind die Wichtel auf dem Weg in das nächste Gasthaus, und man muss dafür sorgen, dass sie nicht nüchtern werden).
Eine Geschwindigkeitsanzeige am unteren rechten Bildschirmrand zeigt dem Spieler wie schnell er unterwegs ist. Diese Anzeige ist wiederrum farbig markiert (grün= langsam, gelb = mittel, rot = schnell)

Die Musik lässt sich über + und – lauter und leiser stellen.

Das Material des Bodens hat Auswirkungen auf das Fahrverhalten der Kuh.
Auf Eis fällt das Steuern wesentlich schwere, Wiese bremst die Kuh, und nur auf Schnee ist das Fahren unerschwert möglich.
Diese 3 Bodenarten gehen ineineander über. (Eis auf Schnee), hierbei werden die beschrieben Effekte kombiniert.

Single Player:

Im Single Player Modus muss man das Ziel erreichen bevor eine vorgegebene Zeit abläuft. Durch das Sammeln von Flaschen erhält man Zeit hinzu, man muss jedoch aufpassen, dass man mit keinen Bäumen kollidiert, da die Kuh sonst aus ihren sanften Träumen erwacht. Das Ziel befindet sich auf der Gegenüberliegenden Kartenkante (es ist recht weit)
Die Anzahl der Sekunden, welche am Ende übrigbleiben werden als Punkte für eine Highscore gewertet, welche man über das Mainmenu erreichen kann.

Single Player Ghost Mode:

Hier tritt man gegen den besten Spieler an, welcher bis jetzt das Spiel gewonnen hat.
Ein Transparenter Kuhkörper repräsentiert hierbei den Gegenspieler. Man kann also quasi gegen sich selber spielen und versuchen immer besser zu sein.
Das Spiel endet hier, wenn die Zeit ausgeht oder die Kuh aufwacht.

Splitscreen:

Zwei Spieler spielen an einem PC (Hotseat) gegeneinander.
Gewonnen hat man, wenn man als erster das Ziel erreicht, die Kuh des Gegners aufwacht, oder die Zeit des Gegners abläuft. Hier geht es um das geschickte aufsammeln der Flaschen, denn der Gegner wird versuchen alle wegzuschnappen.

Wohl das schwerste war, für uns doch eher technischorientierten Studenten, dass umsetzen des Contents.
Das Spiel ist in der vorliegenden Form voll spielbar (gegen die Zeit und gegen seinen Ghost), es existiert eine Bestenliste, und man fährt immer gegen den besten Spieler im Ghost mode.

Kurze Beschreibung, wie die einzelnen Punkte der Angabe (siehe oben) umgesetzt wurden.

Nichttriviale Objekte

Alle Objekte im spiel werden aus 3ds Datenfiles aus von 3d-studio max erzeugten Dateien geladen.

Beschleunigung der Sichtbarkeitsberechnung

View Frustum culling mit bounding spheres in der collsision-Abfrage.

Transparenz-Effekte

Ghost mode

Ein reduziertes Model der Kuh wird mit in den Stencilbuffer geladen, danach wird ein transparenter Quad über den screen gezeichnet wodurch der Ghost sichtbar wird.

Trees

Zusätzlich zur normalen material Textur wird eine Alpha texture geladen, welche zusammen mit einem Alphatest für einen strukturierteren Baum sorgt.

Dust

Der Staub besteht aus einer reinen Alpha Texture.

Text

Der Text besteht aus einer Alphamask und einer Colortexture, die Alphamask schwärzt zuerst die Stellen des Bildschirms auf dem der Text entstehen soll aus und der Text wird danach nur auf Stellen geschrieben die auzsgeschwärzt sind.

Hud

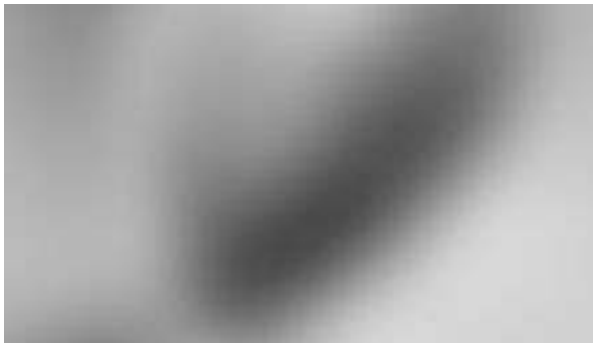
Die Geschwindigkeitsanzeige des Huds besteht aus 3 texturen eine allgemeine Alphamap, eine Anzeigen Alphamap und einer Colortexture. Die Anzeigen-Alphamap erzeugt durch Texturrotation abhängig von der Geschwindigkeit einen Bereich in dem die Farbe der Geschwindigkeitsanzeige besser sichtbar wird.

Spezialeffekte:

Lightmap für die Landschafft

Für das Terrain wird eine Lightmap berechnet, die daraufhin für weitere verwendeungen als bmp abgespiechert wird, bei der Berechnung lässt sich die Qualität der lightmap sowie der Einfallswinkel einer direktionalen Lichtquelle bestimmen.
Die Lightmap wird durch Berechnung der Normalvectoren des Trianglemeshes der Landshaft und deren Interpolation berechnet. Zusätzlich wird die Lightmap nach der Berechnung durch einen cosquadrat-Filter gesmoothed so das eine glatte Lightmap erzeugt wird.

Shading texutre without filtering



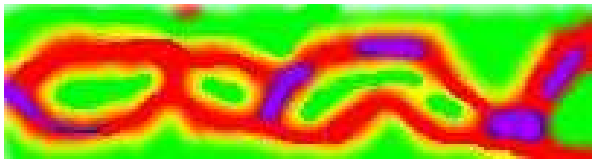
Shading texutre with 4*filtering

MultiTexturing für die Landschafft

Die Material Eigenschafft des Terrains wird aus einer bmp datei in einen Rgba-Color-Buffer geladen aus dem dann die durch Texture combiner functions + Multitexturing + Shadingtexture die Landschafft gezeichnet wird.

Die Schritte dabei sind:

- Schnee-Texture replaced die Vorhandenen Farben zu 100%.
- Eis Texture wir mit der Schnee Texture combiniert als Koeffizient dient dabei die rgb werte (die für rot grün blau identisch sind und als prozentuelle Aufteilung von Eis zu Schnee erzuegt wurden)
- Grass Texture wird mit Eis und Schnee kombiniert als Koeffizient dient dabei der Alphawert des ColorBuffers.
- Als Letztes wird noch die Shadintexture mit *GL_MODULATE* mit Eis, Schnee und Grass kombiniert.



Materialmap

Resourcen:

http://www.cs.auckland.ac.nz/~jvan006/multitex/multitex.html

Stencil shadow (mit directionaler Lichtquelle)

Beim Laden werden die Nachbarschaffts Beziehungen der einzelnen Triangles des Spielermodels berechnet. Um den Schatten zu zeichnen wird über Face-Normals berechnet ob ein Face beleuchtet wird oder nicht. Wenn eine Edge zwischen einem beleuchtet und einem unbeleuchtetem Face gespannt ist erzeugt sie eine Shadowgrenze. Um den Schatten zu zeichnern werden ausgehend von Shadowgrenzen in die Richtung der Lichtstrahlen quads gezeichnet die sich mit dem bereitz gefüllten z-Buffer schneiden.
Zuerst werden die Front-facing Quads gezeichnet welche den Stencilbuffer erhöhen, danach die Back-facing Quads welche den Stencilbuffer erniedrigen. Dadurch das der Z-Buffer bereits gefüllt ist bleiben nur jene Teile des Stencilbuffer ungleich null welche zwischen einem Front und einem Backface liegen.
Wir der double Sided-stencil test unterstütz (der Front und backfaces gleichzeitig mit unterschiedlichen Tests behandeln kann) kann der Stencilbuffer in einem pass befüllt werden. Danach wird über die ganze Scene eine Alphaquad gelegt das jene Pixel die Werte ungleich null im Stencilbuffer besitzen verdunkelt werden.

Resourcen:

Nehe – Tutorials
Lesson 27

Projectionshadows für statische Objecte auf der Landschaft

Beim Laden wird die Landschaft in shadowPatches zerschnitten und zu einer grossen Shadowtexture direkt aus dem Framebuffer zusammengefügt.
Diese wird danach als verdunkelnde Alphatexture über die Landschaft gelegt.
Als erstes wird eine Shadow-Projection-Matrix für eine directionale Lichtquelle erzeugt, diese wird mit den einzelnen Punkten der Trianglemeshes multipliziert worduch die Objecte auf die x – z Plane Projeciert werden. Die Scene wird jewils von der Mitte eines Shadowabschnitts in Beleuchtungsrichtung mit orthogonalem View-Frustum gezeichnet. Nach dem Zeichnen wird der Inhalt des ColorBuffers mit *glCopyTexSubImage2D(...)* in einen Teil der vorbereiteten Texture gespeichert.
Der Vorteil das nur eine Texture benötigt wird, ist das die Landschaft mit dem bereitsvorhandenem Triangle-Mesh gezeichnet werden kann und das dafür das einmalige binden der texture nötig ist.
Die Verzerrungen die einem directionalen Schatten wiederfahren wenn er auf eine schiefe oberfläche geworfen wird entstehen durch eine gleichmässige Verteilung der Texturekoordinaten nur über x und z wodurch eine Veränderung der y komponente zu einer Dehnung des shattens führt.

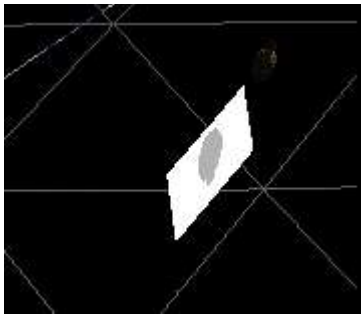
Resourcen:

http://gpwiki.org/index.php/Generating_Mesh_Shadows_On_Terrain_Using_OpenGL
Real-Time Rendering (Second edition)
Projection Shadows (250 - 254)

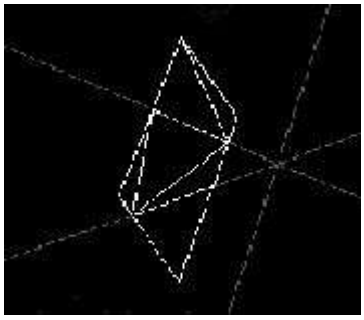
Projection shadows für actionObjects:

Objecte drehen sich um die Y-Achse.

Vom mittelpunkt der Objecte wird ein shadowPatch auf der landschaft berechnet auf dem während dem Spiel die shadow texture des objects gezeichnet wird. Die Bewegung des Schattens wird durch Texturerotation um den Mittelpunkte der Texture und des pathces erzeugt.
Die Alphatexure des Schattens wird wieder als Projection auf die x-z-Plane erzeugt.



Shadowpatch



Outlines of a shadowPatch

Partikeleffekte

Der Staub/Schnee besteht aus Screenalign-Billboards, die Teilchen aus Points.
Die Partikel besitzen Lebensdauer, Position, Richtung und Farbe. Die Richtung und die Anzahl der Partikel wir düber die Geschwindigkeit der Kuh geregelt, die Farbe über den Untergrund. Die Partikel werden von der Schwerkraft beeinflusst. Um das Billboarding zu erzeugen wird aus dem richtungsvektor der Kamera eine Multiplikationsmatrix berechnet.

Resourcen:

Real-Time Rendering (Second edition)
Billboarding (320)

Allgemeine resourcen für Spezialeffecte :

Real-Time Rendering (Second edition)
Nehe – Tutorials
http://www.gamedev.net

Implementierung von Texture Mapping

Wir haben einen eigenen TextureLoader implementiert mit dem es möglich ist tga, gif, sgi, bmp zu laden und in TextureObjecten zu speichern.
Die Klasse behält den Zeiger auf das TextureObject und kann es bei bedarf binden.
Die Qualitätsvactoren (filtering, mipmapping usw.) sind als statische Variablen definiert, welche man durch Subroutinen für nachfolgende Texturen verändern kann.

Eigenschaften 3ds Modelle

Die 3ds Modelle werden je nach ihren Materialeigenschaften entsprechend texturiert.
Die gesamten Flächen werden als runde angenommen und dementsprechende Normalvectoren werden berechnet.

Die Modelle werden über eine direktionale Lichtquelle beleuchtet und deren Materialeigenschaften aus einer Displaylist geladen.
Die Vertices werden wenn möglich in VBO's abgelegt.
Die Objecte wurden in 3d studio max erzeugt.

Steuerung

	Player 0	Player 1 (num lock)
beschleunigen	w	8
bremsen	a	4
rechts	s	5
springen	d	6
treten	space	Arrow up
hupe	y	entf

Zusätzliche Libraries:
Open Dynamics Engine (ODE):
Quelle: http://www.ode.org
Verwendung: Anwendung in MooWorld zur Kollisionserkennungmit der Spielfigur (Kuh)

Open Audio Library (openAL) & Audio Library Toolkit (alut):
Quelle: http://www.openal.org

Verwendung: Anwendung in den MooSound Klassen (MooSound, MooClip, und der noch nicht fertiggestellten MooSoundSystem) zur Ausgabe von Wavedateien (3D Positionierung noch nicht genützt). Mit Hilfe von libogg und libvorbis ist openAL in der Lage ogg-Dateien zu streamen (Hintergrundmusik)
libvorbis & libogg:

Quelle: http://http://xiph.org/ http://vorbis.com
Verwendung: Wie bereits erwähnt werden diese beiden Bibliotheken verwendet um Hintergrundmusik abzuspielen.