

CG 2/3 Übungen, SS 2006

Abgabe 3

Mistelbauer Florian, 534, 0226239, e0226239@student.tuwien.ac.at
Mistelbauer Gabriel, 532, 0326641, e0326641@student.tuwien.ac.at

Hell's Offspring

Spieldesign & Spielprinzip

Das Prinzip unseres Spieles ist es, in den Dungeons Monster abzuschießen und dabei Levels aufzusteigen. Auf einem Terrain, fängt man an, wo man durch eine Person, hier ein Ninja erfährt, dass man alle Monster umbringen soll (das ist kein Quest, sondern nur ein Hinweis!). Um dies zu bewerkstelligen, muss man in die Dungeons hinuntergehen (zwischen den Fackeln; der erste Dungeon wird dann automatisch generiert, was man mit einem Fortschrittsbalken sieht).

Im Dungeon (in einem Level sozusagen), hat man einen Ausgang zum vorigen Dungeon und einen Abgang zum nächsten. Ist man im letzten Dungeon angelangt, so gibt es hier keinen Abgang mehr. Ist man im ersten und man geht durch den Ausgang, mit der Aktionstaste *space*, so gelangt man wieder auf die Oberfläche.

Nichttriviale Objekte

- Terrain: Das Terrain ist ein nichttriviales Objekt, da es ersten einen sehr feinen Mesh hat und noch dazu jeder Vertex Normalvektor der Durchschnitt seiner Nachbarn ist.
- Displacement mapping: Auch hier kann man die Komplexität des Meshes einstellen und auch hier sind die Vertex Normalvektoren der Durchschnitt ihrer Nachbarn.
- Models: Die Modelle sind auch nichttriviale Objekte, welche interpoliert zwischen 2 aufeinander folgenden Frames gerendert werden. Unterstützt werden *.md2 (Quake 2) und *.ms3d (Milkshape) Modelle.

Transparenz Effekte

Transparenz kommt beim Charaktersheet, HUD, Cursor, bei den Partikeln und bei den Zaubern vor. Kann man ausschalten mittels F9.

Spezialeffekte

- Bewegter Rauch: Sind Texturanimationen mit mehreren Texturen, zu sehen bei den Fackeln vor dem Abgang zum ersten Dungeon.
- Rauchspuren: Auch bei den Fackeln, da die Partikeln, folglich der Rauch, Billboards sind.
- Feuer: Die Fackeln.
- Bump Mapping: Nur in den Dungeons. Mittels F10 kann man zwischen keinem Bumpmapping, Emboss Bumpmapping und Bumpmapping mittels Multitexturing wählen.
- Displacement Mapping: Mittels F11 kann man Displacement Mapping für die Wände der Dungeons ein und ausschalten (können nur im VertexArray mode gerendert werden (!!!), nicht im Immediate Mode). Normalvektoren der Vertices sind der Durchschnitt der Nachbarn.
- Portal rendering: Das wichtigste und notwendigste, da man ansonsten fast keine Frames mehr hat. Sieht man gut im Wireframe Modus und ist auch nur in den Dungeons verfügbar.

Referenzen:

- Red Book
- Hearn & Baker, Computer Graphics 3rd Edition
- Stefan Zerbst, 3D Spieleprogrammierung mit DirectX in C/C++
- Marius Apetri, 3D-Graphik Programmierung
- Displacement Mapping & Height Mapping: www.joachimrohde.com
- BSP Tree & Portal rendering: Binary Space Partitioning Trees and Polygon Removal in Real Time 3D Rendering, Samuel Ranta-Eskola

Monster

Die Monster werden am Anfang des Spieles erzeugt und in der *hell_monster_arena* abgelegt. Ihre Eigenschaften werden aus der Konfigurationsdatei *monsters.ini* geladen. Ein Monster hat folgende Eigenschaften: Skalierung, Health, Damage, Experience, Speed und Fire-, Lightning- und Iceresists. Jede Eigenschaft hat einen Basiswert haben, einen Wert um wie viel die Eigenschaft pro Player Level steigt, einen Maximalwert (wichtig bei Geschwindigkeit und Resistenzen der Monster, damit sie nicht zu schnell, oder immun gegen jeden Zauber werden) und einen Multiplikator für einen Boss und einen für einen Endgegner (dieser ist dann folglich am stärksten, was man deutlich merkt ☺). Man kann auch für jeden Typ des Monsters (Normal, Boss, Endgegner) ein eigens Skin angeben, damit man sie unterscheiden kann.

Wird ein Monster in einem Dungeon platziert, so wird es aus der *hell_monster_arena* geklont, damit nicht immer die Konfigurationsdatei geparkt wird und/oder redundante Eigenschaften mehrfach abgespeichert werden. Man bekommt dadurch auch schnellere Ladezeiten beim Dungeon erzeugen.

Weiters hat jedes Monster, sowie der Spieler, einen Treffen- und Sterbesound (müssen *.wav Dateien sein).

Charaktersystem & Charaktesheet

Das Charaktersystem unterteilt sich primär in zwei Kategorien: Den Skills und den Spells. Jede dieser Kategorien hat verschiedene Eigenschaften, in welche man Skillpunkte vergeben kann. Manche Eigenschaften sind erst ab einem höheren Level verfügbar, welche ausgegraut dargestellt werden.

Skills

Hier gibt es mehr Feuer/Eis/Blitz Schaden in der ersten Reihe des Charaktesheets. In der Zweiten gibt es den Gegner anzuzünden, einzufrieren und mehrere Gegner mit einem Blitz anzugreifen, was leider nicht implementiert ist.

Die nächst Rubrik ist mehr Leben und Mana in der ersten Zeile und schneller Leben- und Manaregeneration in der zweiten Zeile. Diese Skills sind implementiert und es ist empfehlenswert die ersten paar Levels diesen Skills zu widmen.

In der nächsten Kategorie gibt es *spell_rate*, *more_speed*, *blood_rage* und *wall_running*. Hier ist nur *more_speed* implementiert.

Spells

Hier gibt es vier Kategorien (* für implementiert):

- Fire: FireBolt*, FireBall*, FireNova, FlameWave, Inferno
- Ice: IceBolt*, IceBall, Blizzard, FrostBeam, FrostNova
- Lighting: ChargeBolt*, Lighing, LightingNova, ChainLighting, ThorsHammer
- Special: BoneSpirit, BloodStar*, Apocalypse, Incinerate

In jeden dieser Sprüche kann man Skillpunkte vergeben, wodurch dann deren Schaden, Recast-time und deren Speed steigt. Die Mana Anforderungen der Sprüche sinken pro Level. Für Special Sprüche haben die Monster keine Resistenzen.

Weiters passen sich die Monster dem Spieler Level an, das heißt, je höher man selber ist, je stärker die Monster und noch mal stärker die Bosse und erst die Endgegner....

Leider ist die Implementierung des Charactersheets sehr sehr inperformant, da wir eine eigene Font Klasse (*vlx_plain_font*) verwenden, welche den Text pixelweise in ein Image kopiert. Wir wollten uns dann aber nicht weiters damit beschäftigen, weil es ja nicht sehr viel mit der Engine und dem eigentlichen Spiel zu tun hat.

HUD

Hier sieht man links den Lebens-, Mana- und Erfahrungspunktebalken, welche auch Texturanimationen sind. Rechts sieht man oben die Lebens- und Manadränke, sowie als Index die Anzahl. Unten sieht man den aktuellen linken und rechten Zauber.

Dungeon

Die Dungeons werden immer neu generiert, wenn man sie zum ersten Mal betritt. Weil es etwas länger dauert, kommt eine Fortschrittsanzeige. Man kann die Levels nicht verlassen und die Sprüche fliegen auch nur solange bis sie entweder auf eine Tür oder auf eine Wand treffen.

Türen

Öffnet man mit der Aktionstaste (*space*) und sie schließen sich automatisch nach 3 Sekunden, sofern man nicht in der Tür steht.

Sonstiges

Wir verwenden keine Libraries, auch nicht für die Loaders, wir haben alles selbst implementiert. Modelliert wurden die Models in Milkshape (selbst gemacht sind nur die Fackel und die Tür).

Weiters haben wir Wert auf Performance gelegt (bis auf das Charactersheet), weswegen wir auch viel mit VertexArrays und DisplayListen (merkt man am deutlichsten, wenn man sie ausschaltet mittels F7) rendern.

Man kann leicht zusätzliche Levels hinzufügen. Einfach die *dungeon.ini* Datei öffnen und einen weiteren Dungeon hinzufügen (muss die gleichen Eigenschaften haben wie die anderen). In der *hell.ini* muss man dann noch *num_dungeons* erhöhen.

Den Charakter kann man auch speichern und laden, indem man in der Datei *hell.ini* *player_save* ändert.

Tastaturbelegung

F1 - display stats

F2 - show fps
F3 - wireframe/solid
F4 - linear/bilinear filtering
F5 - mip mapping on/off
F6 - vertex array/immediate mode
F7 - display lists on/off
F8 - portal rendering on/off
F9 - transparency on/off
F10 - textured/emboss bumpmap/multitex bumpmap
F11 - displacement mapping on/off

W - move forward
S - move backward
A - strafe left
D - strafe right

Q - next left spell
E - next right spell

1 - drink health potion
2 - drink mana potion

Space - open doors, speak

Left click - shoot left spell
Right click - shoot right spell

PageUp - enter prev dungeon level
PageDown - enter next dungeon level or town

Esc - close window/show menu