padoix

CG2LU, SS2007

Name:	Martin Flucka	Name:	Karl Grosse
Matr. Nr.:	0127114	Matr. Nr.:	0225662
Knz.:	532	Knz.:	932
Email:	e0127114@student.tuwien.ac.at	Email:	e0225662@student.tuwien.ac.at

Kurzbeschreibung der Implementierung

Als Enwicklungsumgebung kam Microsoft Visual Studio 2005 zum Einsatz. Die Programmiersprache ist C++, um die Darstellung kümmert sich OpenGL. Für das Fenster-, Sound-, Musik- und Inputhandling wurde SDL 1.2.11 verwendet. Die Shader sind in glsl geschrieben. Zusätzliche GL Extensions werden mittels GLEW eingebunden.

Grundlegendes Gameplay:

Ziel bei Padoix ist es mittels des Spielballes möglichst viele Blöcke zu zerschießen und Powerups einzusammeln um eine hohe Punktzahl zu erreichen. Der Spieler hat anfänglich drei Leben. Berührt der Ball die untere Kante des Spielfeldes oder kommt es zur Kollision mit einer der Spinnen verliert man ein Leben. Sind keine Leben mehr vorhanden ist das Spiel zu Ende. Die Spinnen können nur mittels des Lasers abgeschossen werden.

Die horizontale Steuerung des Pads erfolgt über die Maus.

Auflistung der Funktionstasten:

- Esc = Hauptmenü on/off
- F1 = Debug Info on/off
- F2 = Wireframe on/off
- F3 = Texturefiltering
- F5 = Spiegelungen on/off
- F10 = Fliegende Objekte on/off
- F11 = Toon shader on/off
- SPACE = Pause
- W A S D E F = Kamerasteuerung (nur während Pause möglich)
- Left_Mousebutton = Schuss des Lasers

Nichttriviale Objekte:

- Raumschiff des Spielers

  Das Raumschiff ist ein selbst modelliertes 3ds Objekt, welches mittels Modelloader geladen wird. Es bewegt synchron mit dem Pad.

- Anzeigetafel

  Die Anzeigetafel ist ein selbst modelliertes 3ds Objekt, welches mittels Modelloader geladen wird. Es gibt zwei, jeweils links und rechts vom Spielfeld.

- Fels

  Der Fels ist ein selbst modelliertes 3ds Objekt, welches mittels Modelloader geladen wird. Er befindet sich aus dekorativen Gründen unterhalb des Spielfeldes.

Animierte Objekte:

Auf der oberen Seite unseres Raumschiffmodels befindet sich der „Antigravitationsgenerator“. Er besteht aus drei Teilen welche unabhängig voneinander in Abhängigkeit der Bewegung des Raumschiffes rotieren.

Transparenz Effekte:

Unser gesamtes Menüsystem funktioniert mittels Alphablending. Der jeweilige Menüpunkt wird über die Menühintergrundtextur geblendet.

Die Oberseite des Spielfeldes ist eine im Stencil Buffer erstellte, geblendete Spiegelung der Szene.

Darüber hinaus besitzen die Anzeigetafeln ein transparentes Feld mit wandernder Schrift.

Experimentieren mit OpenGL:

F1 schaltet die Debug Information ein/aus. Mittels F2 Taste lässt sich unser Spiel im Wireframe Modus darstellen. Durch die F3 Taste lässt sich das Textur Filtering einstellen, wobei während dem Ladevorgang die unterschiedlich gefilterten Texturen erstellt werden und per Tastendruck durchgeschaltet werden.

Features des Spieles

Oberflächeneffekte:

Per Pixel Lighting

Für diesen Effekt benutzen wir einen Vertex- und einen Pixelshader, die für uns die gesamte Beleuchtung des Spieles übernehmen. Die Ressourcen dazu sind auf www.lighthouse3d.com im Tutorial „Point Light Per Pixel“ zu finden. Hinzu kam, dass wir drei Lichtquellen verwenden, die Farbe der einzelnen Fragmente mit der Objekttextur kombiniert werden und die Positionen der Lichter noch um den negativen Wert der ModelViewMatrix * gl_Vertex Position transformiert werden.

Cartoon Shading

Dieser Effekt ist von Haus aus deaktiviert. Mittels F11 kann der Shader on/off geschalten werden. Als Ressource diente das Toonshader Tutorial auf www.lighthouse3d.com. Zusätzlich wurde jedem Objekt eine eigene Farbe gegeben und die Berechnung für die Intensität auf drei Lichtquellen erweitert.

Diverses:

Partikelsystem

Anhand eines Tutorials auf http://wiki.delphigl.com zum Thema Partikel Systeme frei nach dem Motto: „Container auf – Partikel rein – Container zu“ haben wir hiermit die Explosionen der Blöcke gestaltet. Aus performance Gründen ist die Anzahl der aktiven Partikelsysteme auf acht begrenzt, bei 500 Teilchen pro System. Jedes Partikel besteht aus drei ineinander gesteckten, geblendeten und texturierten Quads.

Planare Spiegelungen mit Stencil Buffer

Grundlage für diesen Effekt ist ein Tutorial auf http://wiki.delphigl.com. Im ersten Renderpass wird die gespiegelte Szene berechnet und mittels Stenciltest auf den Spiegel (Spielfeld) geclippt. Der zweite Renderpass kümmert sich dann um die gesamte Szene und blendet mit dem Bild im Stencil Buffer.

Features des Spieles

Textur Blending für die Darstellung des Menüs und des Ladeschirms um die Texturgrößen/anzahl niedrig zu halten und eine geeignete Performance beim Laden zu erreichen. Eine mittels Matrizenmanipulation rotier- und zoombare Kamera. Eine Collisiondetection per Bounding Box. Spinnen die den Spieler angreifen und nur mittels Laserstrahl eliminiert werden können. Die Skalierung der Spielgeschwindigkeit und Gamelogic anhand der Delta – Time, und eine funktionierende Highscore. Besonders Stolz sind wir auf unsere Powerups.

	Standard Powerup ohne Effekt	(100 Punkte)
	Pad vergrößert sich	(150 Punkte)
	Pad verkleinert sich	(150 Punkte)
	Ball wird schneller	(200 Punkte)
	Ball wird langsamer	(200 Punkte)
	Cam Target = Ball Pos	(300 Punkte)
	Cam Pos = Ball Pos	(400 Punkte)
	Cam Pos = Pad Pos	(500 Punkte)
	Block taucht wieder auf	(200 Punkte)
	Blocks werden invertiert	(1000 Punkte)
	Maussteuerung wird invertiert	(500 Punkte)
	Extraleben	(10 Punkte)
	Superduper	(10 Punkte)
	Multiball	(0 Punkte)

Als besonderes, erheiterndes Feature sehen wir noch die Tatsache, dass wir alle Sounds selber aufgenommen haben und alle Objekte selber modelliert haben.

Zusätzliche Libraries und Tools

SDL 1.2.11	http://www.libsdl.org/download-1.2.php
SDL_image 1.2	http://www.libsdl.org/projects/SDL_image/
SDL_mixer 1.2	http://www.libsdl.org/projects/SDL_mixer/
GLEW	http://glew.sourceforge.net/
Modelloader	http://www.garagegames.com/
3DS MAX	http://autodesk.com/