CG 2/3

„Speckled Action“

Günter Wallner, 9925247 (881)
Simone Kriglstein, 9925051 (881)
Stefan Felkel, 9926464 (881)

„Speckled Action“

3. Abgabe

Allgemein

- Spielbeschreibung

Gespielt wird auf offenem, inselartigem Terrain. Auf der Spielfläche befinden sich 1-3 gegnerische Giraffen, außerdem feindliche Tiere wie Krokodile, Schlangen oder Ameisen, denen man nicht in den Weg kommen sollte.
Außerdem liegen auf der Insel verstreut Ringe, deren Sinn im folgenden Unterkapitel erklärt wird.

- Ziel des Spieles ...

... ist, so schnell wie möglich eine bestimmte Anzahl von frei herumliegenden Ringen in ein vordefiniertes Basislager oder Nest zu bringen. Derjenige der Giraffen gewinnt, wer als erster fleißig alle Ringe lebend nach Hause gebracht hat.

- Leben

Auf der Insel lauern Gefahren, welche der Giraffe Leben rauben können. Dazu zählen jegliche Berührungen mit territorialen Gegnern, also z.B. Krokodile, Schlagen, Ameisen.
Lebensabzug bekommt man auch, wenn man von gegnerischen Giraffen abgeschossen wird.

- Steuerung und Tastenbelegung

Cursor (Vor / Zurück / Links / Rechts)	Steuerung der Giraffe
Leertaste	Bewaffnung aufnehmen bzw. ablegen, Intro bzw. Outro beenden
Enter	Schuss, falls bewaffnet
Mouse nach links / rechts bewegen	Links/Rechts Schwenk
linker Mousebutton	Schwenk zurücksetzen
Esc	Beenden des Spiels
F1	Hilfe-Bildschirm an/aus
F2	Mipmap an/aus
F3	Display Listen an/aus
F4	Texturqualität an/aus
F5	FPS an/aus
F6	Wireframe Modus an/aus

Umsetzung der Angabe

- Nichttriviale Objekte

  - Giraffe
    Sie ist sicher das aufwendigste Objekt des Spiels mit den meisten Dreiecken. Ihre Oberfläche ist teilweise gekrümmt.

  - Vegetation
    Im Spiel gibt es Palmen, Bäume, Büsche, Steine verschiedener Art. Bis auf die Steine sind die Objekte nicht-konvex.

  - Krokodil / Schlange / Skorpion / Ameise / Delphin
    Ebenso diese Objekte sind aufwendiger modelliert und nicht-konvex.

- Experimentieren mit OpenGL

  - Vertex Arrays

    ... wurden benutzt, um das Terrain mit Triangle-Strips zu rendern

  - Mip Mapping

    ... wurde beim Terrain verwendet

  - Textur-Qualität

  - Display-Listen

- Transparenz

  Wir haben TGA-Dateien verwendet und diese mittels Alpha-Kanal über die Szene geblendet.

  - UID

    Man sieht durch den grauen Balken hindurch, auch die Schriften vor Start des Levels („Lets GO“) und nach dem Sieg oder der Niederlage („You won“, „You lost“) sind halbtransparent.

  - Gegnerische Giraffen

    Diese besitzen ein Farbsymbol über ihrem Körper, welches halbtransparent ist.

  - Partikelsystem

  - Meer

    Das Meer ist mit einer halbtransparenten Textur gerendert und texturanimiert.

- Beschleunigung der Sichtbarkeitsberechnung

Wir haben View-Frustum-Culling eingebaut und damit die Spielgeschwindigkeit um ein Vielfaches erhöhen können.

Spezialeffekte

- Rauchspuren

  Beim Abschuss der Geschosse bildet sich hinter den schwarzen Kanonenkugeln (ja, es sind tatsächlich Kanonenkugeln) eine graue Rauchspur.
  Diese wurden mit Hilfe eines Partikelsystems und transparenten Partikeln gerendert.

- Wasser

  Das Meer rund um die Insel ist ein animierter Mesh aus Dreiecken, die Texturkoordinaten dazu werden berechnet.

- Transparenz

  Transparente Objekte wie oben bereits erwähnt, wurden in die Landschaft bzw. ins UID hinzugefügt. Die transparenten Flächen werden in der richtigen Reihenfolge, falls überlagert, ausgegeben.

- Partikelsysteme

  Es existieren mehrere Partikelsysteme mit unterschiedlichem Aussehen.
  Bei den Nestern brennen zwei Flammen in der vom Nest vorgegebenen Farbe.
  Um die Ringe befindet sich ein goldener Schein, der Partikel in alle Richtungen ausstrahlt.
  Beim Abschuss der Kanonen entsteht ein Rauchspur-Partikelsystem.
  Beim Auftauchen des Skorpions aus der Erde wird Staub durch Partikel simuliert.
  Am Ende gibt es Schlusssequenzen mit einem Feuerwerk-Partikelsystem bzw. mit herunterfallendem Regen.
  Auch beim Treffen einer gegnerischen Giraffe erscheint eine Explosion.

- Projizierte Schatten

  Mit Hilfe des Stencil Buffers werden die Schatten der statischen Objekte (also Vegetation und Steine) berechnet und auf die darunter liegende Ebene projiziert.

Besonderheiten

- KI der gegnerischen Giraffen

  Das Grundgerüst ist der Pfad. Sie geht den Pfad entlang, falls sich ein Ring in absehbarer Nähe befindet, so verlässt sie den Pfad und holt den Ring.
  Der Pfad befindet sich ausschließlich aus Punkten, welche ihre Nachbarn kennen. Zusätzlich weiss jeder Punkt anfangs, welche Distanz er zu welchem Nest hat (also wie viele Wegpunkte die Giraffe zurücklegen muss)

Mittels ein paar Regeln wägt die Giraffe das Verhältnis Ring : Weg-nach-Hause ab und entscheidet, ob sie dieselben zum Nest bringt. Nachdem die Distanz der Wegpunkte schon von vornherein definiert ist, folgt sie einfach der niedrigsten Distanz und findet so auf dem schnellsten Weg zurück.
Falls eine Giraffe den Spieler berührt, so rennt sie einen kurzen Weg fort und schießt auf den Spieler.

- KI der statischen Feinde

  Wir unterscheiden zwischen Feinden, die sich verstecken, und welche, die sich offen zeigen.
  Grundsätzlich hat jeder Feind ein Territorium zugewiesen, in dem er sich bewegt. Falls der Spieler das Territorium betritt und somit verletzt, wird die KI auf denselben losgehen. Falls sie sich versteckt hat, so wird sie schnell auftauchen und versuchen, den Eindringling zu vernichten.

- Level-Editor mit eigenen Dateiformaten

  Mit Hilfe eines selbstprogrammierten Leveleditors mit Windows-Oberfläche (programmiert mit Delphi 6) ist es uns möglich, ein Dateiformat mittels eines geeigneten User Interface Designs zu erstellen.

  Features:
  - Setzen aller statischen Objekte, der Ringe, der statischen Feinde, der Nester der gegnerischen Giraffen (ist zugleich Startposition der Giraffen)
  - Setzen der Skybox-Texturen, sowie der Lightmap, der Höhe der Insel und deren Seiten
  - Festlegen des Pfades der KI und der Distanzen mittels Graphentfernungsalgorithmus

- Multipass Multitexturing (Terrain) inclusive Lightmap

  Das Terrain wird mit drei verschiedenen Texturen und mit einer Lightmap mittels Multipass gerendert. Dazu werden die OpenGL-Extensions glMultiTexCoord2fARB, glActiveTextureARB, glClientActiveTextureARB.

- Milkshape-Loader (mit Keyframe-Animationen)

  Der selbstgeschriebene Objekt-Loader setzt die Objekte in die Landschaft und übernimmt die Animationen aus den zugrundeliegenden MS3D-Dateien. Für die Skeleton-Animation wurden Matrizen und Quaternionen verwendet.

- TGA und BITMAP-Loader

Sonstige zu erwähnende Elemente des Spiels bzw. der Programmierung

- Collision Detection

  - mit Terrain

    Ein Strahl wird nach unten geschossen und mit den darunter liegenden Dreiecken geschnitten. Der resultierende Punkt ergibt die Höhe des zu setzenden Objekts.

- mit steilem Gelände

  Die Steilheit aller Triangles des Terrains wird von vornherein über den Normalvektor festgestellt. Über eine boolsche Variable wird festgelegt, ob das Dreieck von der Spieler-Giraffe betreten werden darf oder nicht.
  Im Spiel wird dann einfach überprüft, ob die Giraffe sich auf ein „verbotenes“ Dreieck bewegen will (Strahl nach unten, Schnittberechnung); die Bewegung wird dann unterbunden.

- mit anderen Objekten

  Mittels Kugel-Kugel-Schnitttest wird die Kollision ausgewertet. Jedes Objekt besitzt eine Bounding-Sphere.

o Radar

Ein Radar wird in der unteren linken Ecke angezeigt, man kann darauf erkennen, wo der Spieler und wo sein Nest sich befinden.

o Quadtree

Fast alle statischen Objekte und alle Dreiecke werden in einem Quadtree verwaltet.

o Licht

Die Lichtquelle befindet sich zentral über der Insel. Es wird sowohl ambientes, diffuses und spekulares (sichtbar an den Kanonenkugeln) Licht verwendet.

o Kamera

Die 3rd-person-camera befindet sich hinter dem Spieler und etwas höher. Falls man sich an einem steilen Hang befindet und sich dreht, schneidet die Kamera den Berg nicht, sondern schwenkt über den Berg.

o Menü

- Start / Spiel / Credits

o Intro

Änderung der Kameraperspektive über der Insel mit kreisendem Schwenk. Eine transparente Textur fordert den Spiele auf, den Level mit Schlachtrausch zu beginnen.

o Abspann

- wenn gewonnen

  Die Kamera wechselt in die Vogelperspektive und von unten wird ein riesiges Feuerwerk in allen Farben gestartet, der Schriftzug „You won“ erscheint mit der

Aufschlüsselung der eingesammelten Ringe aller Giraffen am unteren Rand des Bildschirms.

- wenn verloren

  Die Texturen der Skybox werden auf Schlechtwetter gewechselt, von oben prasselt Regen auf das arme Häufchen Elend von Giraffe, ein Schriftzug darunter meldet „You lost“ und in der rechten unteren Ecke wird der Spielstand aller Giraffen angezeigt.

o Sound

Verwendete Tools sind Goldwave, Sound Forge 6.0, Multiquence und der Sound-Teil der pLib.
Bis auf das Wavefile im Kapitel „Quellen ...“ (s.u.) wurden alle Soundfiles selbst erstellt.

o Backface Culling

Zusatztools

o Sound-Library von pLib (http://plib.sourceforge.net/sl/index.html)

Modellierung

Alle Objekte und Texturen wurden von eigener Hand geschaffen.
Die dafür verwendeten Tools waren Maya 4.0, 3DS Max 5 und Milkshape 1.6.5..

Zuerst wurden die Modelle in Maya 4.0 beziehungsweise in 3DS Max 5 erstellt. Mit Hilfe von LithUnwrap wurden die UV-maps erstellt und anschließend mit Adobe Photoshop 7 bearbeitet. In Milkshape wurde die Textur zugewiesen und das Modell mit Bones animiert.
Die Anzahl der Dreiecke wurden in Maya 4.0 reduziert um eine gute Performance zu gewährleisten.

Die Skyboxen wurden unter Bryce 5.0 erstellt und im Adobe Photoshop 7 angepasst.

Für das Menü wurden Screenshots vom Spiel verwendet.

Verwendete Quellen, Dateien, Hilfen

- Wavefile (http://www.alexsoftware.com/SoundFX/laughcartoon.wav)

- Milkshape Model Loader:
  - File Format Specification (http://koci.opengl.cz/vc/ms3dspec.txt)
  - Model Animation (http://rsn.gamedev.net/tutorials/ms3danim.asp)

- Partikelsystem
  http://www.gamasutra.com/features/20000623/vanderburg_pfv.htm
  http://www.darwin3d.com/gamedev/articles/col0798.pdf

- Shadows: [1]

- BMP, TGA Loader: [1]

- Terrain (http://www.delphi3d.net/articles/viewarticle.php?article=terraintex.htm)

- Kamera : [1]

- Sonstige Quellen :

  - OpenGL Forum auf http://www.opengl.org/
  - „OpenGL Superbible 2nd Edition", Richard S. Wright Jr. & Michael Sweet, Waite Group Press, 1999
  - OpenGL Programming Guide, 2nd Edition, 1997, Addison-Wesley, (Woo, Neider, Davis)

[1] "OpenGL Game Programming",
Kevin Hawkins & Dave Astle, Prima Tech, 2001