

Übersicht:

Inhalt

Übersicht:	1
Gameplay.....	1
Nichttriviale Objekte	1
Animierte Objekte	2
Experimentieren mit OpenGL.....	2
Effekte	2
Besonderheiten	3
Steuerung	4
Funktionstasten:.....	4

Gameplay

Das Gameplay ist durch Physx von Ageia geprägt. Die Hovercrafts sliden in den Kurven nach außen, um eine möglichst realistische Steuerung zu simulieren. Eine Collision-Detection ist durch Physx ebenfalls vorhanden. Um immer zu wissen, welche Boje als nächstes angefahren werden muss, befindet sich für beide Player über der nächsten anzufahrenden Boje ein roter Pfeil. Kommt man nahe an diese rote Boje heran, springt der Pfeil automatisch zur nächsten Boje. Kommt es vor, dass ein Hovercraft auf dem Rücken zum Liegen kommt, kann dieses durch Betätigen einer bestimmten Taste wieder gewendet werden (Hovercraft1 -> e ; Hovercraft2 -> o). Für beide Player ist eine Geschwindigkeitsanzeige vorhanden. Der Player 1 steuert mit den Tasten W, A, S und D, der Player 2 mit den Tasten I, J, K, L. Ist das Rennen beendet, kann es mit der Taste M neu gestartet werden. Während dem Spiel ist es nicht möglich, durch betätigen der Taste M das Spiel neu zu starten. Jedem Spieler werden die bereits abgefahrenen Bojen angezeigt.

Sieger ist der, der zuerst alle Bojen abgefahren hat ☺.

Nichttriviale Objekte

Es sind mehrere nichttriviale Objekte im Spiel vorhanden. Dazu gehören die Hovercrafts selbst, sowie ein Boot, welches sich in der Strecke befindet. Weiters ist ein Grabstein im Spiel zu sehen. Als gekrümmtes Objekt ist die Welt selbst vorhanden, welche trotz der relativ geringen Polygonanzahl, den Eindruck einer kontinuierlichen Fläche erweckt.

Animierte Objekte

Animierte Objekte sind die Bojen, welche angefahren werden müssen. Diese bewegen sich mit einer Sinus-Schwingung in mehreren Achsen. Die Wasseroberfläche wird ebenfalls mit einer Sinus-Schwingung animiert.

Experimentieren mit OpenGL

Während dem Spiel kann mit F3 der WireFrame-Modus ein- und wieder ausgeschaltet werden. Um das Spiel nicht frameabhängig zu spielen, kann mit F4 zwischen frameabhängigem und nicht frameabhängigem Modus gewechselt werden. Das Umschalten auf den frameunabhängigen Modus bringt den Vorteil, dass die Kamera nicht ruckelt (PhysX bugt ein bisschen mit der frameabhängigen Simulation)

Die FPS-Anzeige kann mit F2 ein- und wieder ausgeschaltet werden.

Zur Beschleunigung des Spiels wurde View-Frustum-Culling implementiert, wodurch alle Objekte, die sich nicht im momentanen Sichtfeld befinden, gar nicht erst zur Graphikkarte geschickt, sondern schon vorher ausgesondert werden.

Effekte

Als Spezialeffekte wurden ein Partikelsystem, Wasserspiegelung sowie eine animierte Wasseroberfläche implementiert. Das Partikelsystem wurde wie im NEHE-Tutorial (<http://nehe.gamedev.net/data/lessons/lesson.asp?lesson=19>) implementiert.

Die Wasserspiegelung sowie die Wasseroberfläche wurden selbst mit der Hilfe von Benedikt Stehno geschrieben.

How it works:

Um den Wassereffekt mit Spiegelungen zu erzielen, wird zuerst die komplette Welt normal gerendert, dann wird eine Clipping Plane erstellt (

```
GLdouble plane[4]={0.0f,1.0f,0.0f,0.0f};
```

```
glClipPlane(GL_CLIP_PLANE0, plane);
```

```
glEnable(GL_CLIP_PLANE0);
```

) und das Koordinatensystem um die y Achse gespiegelt(

```
glScalef(1, -1, 1);
```

). Weiters wird eine Blending Fuction (

```
glBlendFunc(GL_SRC_COLOR, GL_DST_COLOR);)
```

genommen um die alte Welt noch ein bißchen im Wasser zu sehen.

Nun wird die Welt einfach nochmal gespiegelt, mit Blending Function und Clipping Plane versehen gerendert.

Danach wird noch das Wasser, welches auf den Texturkoordinaten eine zeitversetzte Sinusschwingung hat gerendert (Natürlich auch mit Blending Function).

Besonderheiten

Als Besonderheit des Spiels gilt der selbst geschriebene FBX-Model-Loader, mit dem Models, die im Alias FBX-Format gespeichert wurden, in das Spiel geladen werden können. Dazu muss lediglich ein Eintrag in der garden.course, welche sich im Ordner data befindet, gemacht werden. Dort können auch etwaige Translationen, Rotationen, Skalierung angegeben werden, die schon beim Laden auf das Objekt angewendet werden.

Zusätzlich kann in der garden.course auch auf einfache Weise die Rennstrecke verändert werden durch umsetzen der Koordinaten der Waypoints (Achtung die Koordinaten der Bojen sollten auch angepasst werden, sonst sieht man keinen Hint wohin man fahren soll.)

Zur Erzeugung des Fensters wurde GLUT, als Physik-Engine wurde Physx von Ageia verwendet. Für den FBX-Model-Loader wurde das FBX-SDK von Autodesk verwendet. Weiters wurde FMOD verwendet, um Hintergrundmusik in das Spiel zu bringen.

Wir haben unser Projekt auf dem SampleRaycastCar Beispiel, welches mit dem PhysX SDK mitgeliefert wurde aufgebaut und unsere Modelle durch PhysX Modelle abstrahiert um Collision Detection und Fahrtsimulation des Hovercrafts zu ermöglichen. Für das Laden der Modelle wurde unser selbstgeschriebener FBX Loader verwendet. Das Rendering haben wir dann über unsere eigenen Rendermethoden durchgeführt.

Um die Objekte des Spiels zu modellieren kamen Maya und Milkshape3D zum Einsatz. Milkshape3D war vor allem notwendig, um die Objekt im FBX-Format abzuspeichern, da die direkt aus Maya exportierten FBX-Files Fehler verursachten. Weiters war es nur mit Milkshape3D möglich, die Pfade auf die als BMP-Dateien vorliegenden Texturen relativ anzugeben, da Maya lediglich absolute Pfade zuließ.

Weiters gibt es einen Splitscreen Modus, der das Spielen für 2 Spieler gleichzeitig möglich macht.

Steuerung

w,a,s,d -> Steuerung von Hovercraft1

e -> Flip Hovercraft1, wenn man wo feststeckt

i,j,k,l -> Steuerung von Hovercraft1

o -> Flip Hovercraft1, wenn man wo feststeckt

m -> Restart Game

Funktionstasten:

F1 - Hilfe

F2 - Framerateanzeige ein/aus

F3 - Wire Frame ein/aus

F4 – Framedependant Simulation ein/aus

F5 – Koordinaten Ausgabe ein/aus

F6 - View frustum culling ein/aus