

Jedi Knight for Cave

1) Spielbeschreibung

Spielprinzip:

Der Spieler befindet sich im Cave auf einer, seitlich durch Mauern begrenzten, offenen Fläche. Ziel ist es, den fiktiven Raumgleiter von Prinzessin Leia, der sich gedanklich hinter dem Spieler befindet, von den von vorne anfliegenden gelben, orangen und roten Plasmabällen zu schützen, in dem man diese durch das Lichtschwert des Jeditritters (eine Projektion als Verlängerung des Wand) ablenkt.

Spielablauf:

Vor Spielbeginn wird ein Willkommens-Screen angezeigt. Durch Drücken eines beliebigen Wandbuttons startet das Spiel. (Falls keiner zu funktionieren scheint, muß wahrscheinlich das Trackersystem neu gestartet werden.) Nach dem Spiel wird ein Screen mit dem Score angezeigt. Erneutes Drücken eines Wandbutton startet eine weitere Runde des Spiels.

Punkte:

Der Spieler erhält für jeden getroffenen Plasmaball einen Punkt. Falls es gelingt einen Plasmaball in Richtung der Angreifer zurück zu lenken, gibt es weitere 4 Punkte (siehe auch Punkt räumliche Abmessungen). Falls es gelingt, einen Plasmaball so abzulenken, daß dieser einen weiteren anfliegenden Plasmaball trifft, gibt es weitere 5 Bonuspunkte.

Spieldauer:

Der zu beschützende Raumgleiter startet mit 100% Health. Jeder Plasmaball, der den Spieler passiert und den Cave an der offenen Seite verläßt, zieht dem fiktiven Raumgleiter zwischen 1% (gelbe Bälle) und 5% (rote Bälle) Health ab. Sinkt der Schild des Raumgleiters auf 0% ist das Spiel zu Ende. Punktstand und Health werden ständig im rechten oberen Sichtbereich des Spielers eingeblendet.

Kollisionen:

Die Plasmabälle werden von den Wänden und vom Boden reflektiert. Bei Ablenkung durch das Lichtschwert fliegen die Bälle in jene Richtung weiter, in die das Lichtschwert bewegt wurde.

Schwierigkeitsgrad:

Der Schwierigkeitsgrad ist als Kommandozeilen Parameter einzustellen.
./jedi --sphere-rate 150

Der Parameter gibt die Wahrscheinlichkeit an, daß während eines Simulationsschrittes eine neuer Plasmaball erzeugt wird. Ein Simulationsschritt dauert mind. 10msec (abhängig von der Prozessorlast). Ein Wert von 150 würde bedeuten, daß alle 10 Millisekunden mit der Wahrscheinlichkeit von 1:150 ein neuer Plasmaball erstellt wird. Der Standardwert liegt bei 150. Je niedriger der Wert gewählt wird, desto mehr Bälle werden erzeugt. Die Generierung (Flugbahn, Geschwindigkeit, Zerstörungskraft) erfolgt vollkommen zufällig.

Räumlich Abmessungen (siehe auch config.h):

Die Plasmabälle tauchen in einem Bereich von 10 - 15 Meter vor dem Cavemittelpunkt auf. Die horizontale Streuung im Ursprung beträgt +/- 8 Meter. Die vertikale Streuung beträgt +/- 2 Meter. Der Durchmesser der Plasmabälle beträgt 15cm. Das Lichtschwert ist 1 Meter lang und hat einen Durchmesser von 5cm. Die von den Wänden begrenzte Spielfläche wird trichterförmig nach vorne hin breiter. Die Distanz zwischen den Wänden beträgt 2,5 - 4 Meter. Die Wände sind 2,5 Meter hoch.

Spiel am Desktop:

Das Spiel kann auch außerhalb des Caves am Desktop getestet werden. Anstatt

über den Wand und das Headset werden das Lichtschwert und der virtuelle Betrachter über die Tastatur gesteuert.
Bei gedrückter ALT-Taste erfolgt die Steuerung des Lichtschwerts, ansonsten die des Betrachters.
Für die Positionsänderung werden die Tasten 4-6 (X), 2-8 (Y) und 1-9 (Z) verwendet (NumLock nicht vergessen!). Die Cursortasten werden für die Änderung der Ausrichtung gebraucht.
F1 - F4 entsprechen den Wand Buttons.
Die Tasten 0 und 5 resetten die Position des Lichtschwerts und des Betrachters.

2) Programmbeschreibung

Das Programm wurde im Rahmen der LVA "Virtual Reality im CAVE" im Wintersemester 2002/03 von den Studenten

Mitzner Gerhard
Hölzl Johannes
entwickelt.

Dateiliste:

Makefile	Makefile für die Plattform "Origin/CAVELib"
Makefile.glut	Makefile für die Plattform "Linux/GLUT"
colldetect.c	Hauptroutinen der Kollisionserkennung
colldetect.h	
colldetectmath.c	mathematische Routinen für die Kollisionserkennung
colldetectmath.h	
config.h	zentrale Konfigurationsdatei
datatransfer.c	Routinen für die Datenübertragung vom Hauptprozeß zu den Renderingprozessen
datatransfer.h	
floor128x128.png	Boden-Textur
font.c	Routinen zur Textausgabe
font.h	
gameengine.c	DIE zentrale Spielsteuerung
gameengine.h	
graphic.c	plattformabhängiger Teil der Grafikroutinen
graphic.h	
input.c	Input-Routinen (plattformabhängig)
input.h	
main.c	der Beginn von allem :-)
models.c	Routinen zur Erzeugung der 3D-Modelle
models.h	
readme.txt	diese Datei
renderer.c	Rendering-Routinen (plattformUNabhängig)
renderer.h	
sorority.txf	Font-Datei
texture.c	Routinen zum Laden von Texturen
texture.h	
vector.c	grundlegende Vektoroperationen
vector.h	
wall128x128.png	Seitenwand-Textur