

House „Adventure“

Stefan Hackl, Florian Keclik, Kornelia Ganglbauer

Ziel:

Unser Hauptziel für unser CAVE Projekt war, eine Szene aufzubauen, in der sowohl Collision Detection als auch Interaktion eingebaut war. Außerdem sollte das Ganze auch einen gewissen sozialen Charakter enthalten.

Das realisierten wir in Form eines Hauses, in welchem man sich herumbewegen kann. Dieses Haus besteht aus zwei Räumen, welche auch eingerichtet wurden und somit wirklich Teil einer Wohnung sein könnten.

Verwendete Hilfsmittel:

Wir entwickelten dieses Projekt in der Programmiersprache C++ unter Verwendung der OpenGL Performer API. Damit wurde das Haus und seine Umgebung realisiert. Um unser Projekt dann am CAVE laufen zu lassen, verwendeten wir weiters die CaveLIB.

Um die Einrichtungsgegenstände zu kreieren, setzten wir Maya ein. Die entstandenen Objekte wurden mit Hilfe von Deep Exploration in .3ds Format konvertiert und mit Texturen versehen. Für die Endversion wurde noch die pfconv.exe des Performers eingesetzt, um die Objekte als .pfb Knoten zu erhalten. Weiters kam auch das Programm Gimp zum Einsatz, um Texturen in das benötigte .rgb Format zu konvertieren.

Navigation:

Unser Projekt ist sowohl am Desktop als auch im CAVE zu verwenden. Dabei unterscheidet sich die Navigation allerdings ein wenig.

Am Desktop erfolgt die Navigation grundsätzlich mit den Cursor Tasten. Durch Verwendung dieser Tasten ist es möglich, sich durch den Raum zu bewegen. Wir haben die Pfeil-oben Taste verwendet, um vorwärts und die Pfeil-unten um rückwärts zu gehen. Um sich im Raum drehen zu können wurden die Pfeil-links und die Pfeil-rechts Taste verwendet.

Um die Interaktion zu ermöglichen, verwendet man am Desktop die Maus. Wird mit der linken Maustaste auf ein interaktives Objekt geklickt, wird die dazugehörige Aktion ausgelöst. Um die Applikation am Desktop zu beenden wird die Escape Taste gedrückt.

Im Cave erfolgt die Navigation mit dem Wand. Dabei kann man sich unter Verwendung des Wand-Cursor-nachoben und des Wand-Cursor-nachunten vorwärts bzw. rückwärts bewegen. Für die Drehungen muss man die Wand-Cursor-nachlinks und Wand-Cursor-nachrechts einsetzen.

Auch die Interaktion wird mittels Wand ermöglicht. Dabei erfolgt durch Zeigen auf ein Element die Auswahl. Erst wenn auf den dritten (rechten) Wandbutton geklickt wird, wird die dazugehörige Aktion ausgelöst. Weiters gibt es im Cave noch die Flugfunktion. Diese wird durch Drücken der mittleren Wandtaste aufgerufen. Danach kann man in Wandrichtung aufsteigen. Wird die mittlere Taste ein zweites Mal gedrückt, so wird dieser Modus wieder verlassen.

Um die Applikation zu verlassen, muss man im Cave den ersten und dritten Wandbutton (rechts) gleichzeitig drücken.

Cave:

Um die Applikation im Cave laufen zu lassen sind folgende Änderungen im Source notwendig:

Die Zeile „#define CAVE“ muss entkommentiert werden (bzw. kommentiert für den Desktop-Einsatz), dann wird der Source für die CaveLIB und für den Caveeinsatz ansich vorbereitet.

Interaktionen:

Das vorliegende Ergebnis kann wiederum als Ausgangspunkt für ein einfaches 3D-Adventure gesehen werden, und ist auch als solches ausgelegt.

Interaktionen sind einfach zu erstellen, es ist im wesentlichen nur nötig, eine SwitchAble oder MoveAble-Klasse (je nachdem, ob ein pfSwitch oder pfDCS-Knoten sinnhaft ist) instanziiert werden, die eine Interaktion beinhaltet, und dann in das interact-Array gehängt werden, um die Interaktion zu aktivieren.

Wird das interaktive Element ausgewählt (mit der Maus am Desktop, oder mit der Wand im CAVE), wird das entsprechende interaktive Objekt mittels der interact()-Methode aufgerufen. Zurzeit sind folgende Interaktionen implementiert:

Türe öffnen: Ist per Klick auf die Tür möglich, aber nur wenn der Schlüssel bereits aufgenommen wurde.

Türmatte entfernen: Durch Klick auf die Türmatte wird diese entfernt.

Schlüssel aufnehmen: Durch Klick auf den Schlüssel wird dieser in das Inventar (nicht sichtbar) eingefügt.

Lichtschalter betätigen: Ein Klick auf eine der Lichtschalter schaltet das Licht in den Räumen ab/ein – die Schaltung erfolgt elektrisch korrekt.

Kastentüre öffnen: Die Kastentüre ist durch einen Klick ebenfalls zu öffnen.

Die Interaktion erfolgt am Desktop mit der Maus (auf ein Element mit der linken Maustaste drücken), oder im Cave mit der Wand (auf ein Element zeigen, dieses wird dann auf „Highlight“ gestellt) und der rechten Wand-Taste.

Probleme:

Im Laufe unseres Projekts sind wir doch auf einige Probleme gestoßen. Vor allem der Einsatz des OpenGL Performer API war ein schwieriges Unterfangen. Leider gab es keine wirklich hilfreiche Dokumentation für die Verwendung des Performers in C++. Daher verbrachten wir auch viel Zeit mit Suchen und Ausprobieren, bis wir die gewünschten Lösungen erreichten. Oft war ein Problem in nur wenigen Zeilen Code zu lösen. Das Hindernis war also nicht das Programmieren an sich, sondern viel mehr das Finden der gesuchten Methoden.

Ein weiteres Problem entstand durch den Einsatz von Maya zum Modellieren der Möbel. Bei der Konvertierung von fertigen Maya Szenarien ging leider die Textur wieder vollständig verloren. Dieses Problem lösten wir, indem wir die Texturen erst im Programm Deep Exploration hinzufügten.

Abgabe:

Die Source/Objekt-Dateien liegen auf der Origin im Account vprk19 unter perfHouse. Die Klassendateien sind kommentiert.