CG 2/3 Übungen, SS 2004

Dokumentation zu

„The Grand Theft“

Spielversion 3. Abschnitt

Johannes, Kiraly, 881, 9103041,
hannes.kiraly@gmx.at

Bernhard, Pflugfelder, 532, 0027467,
e0027467@student.tuwien.ac.at

1 Hintergrundstory

Als einer der berüchtigtsten und berühmtesten Kunstdiebe seiner Zeit soll Jack McNamara nun seinen möglicherweise letzten und gefährlichsten Auftrag durchführen, nämlich das bekannte London Museum nächtens zu besuchen.
Die Aufgabe von Jack McNamara ist es, so viele der wertvollen und einzigartigen Kunstgegenstände des Museums zu entwenden und natürlich auch wieder heil das Museum zu verlassen. Dies ist allerdings deutlich schwieriger als alles, was Jack McNamara bisher bei seinen uneingeladenen "Besuchen" erlebt hatte, da dieses renommierte Museum über eine beachtliche Sicherheitstruppe verfügt.

Der Spieler soll nun Jack McNamara durch diesen spektakulären Raubzug führen und möglichst wertvolle, d.h. mit vielen Punkten dotierte, Kunstgegenstände entwenden und danach wieder lebend das Museum verlassen.

2 Spielbeschreibung

Wie es aus den meisten bisherigen Labyrinth-Spielen bekannt und gewohnt ist, wird auch "The Grand Theft" aus der so genannten "first person" Perspektive gespielt, wobei eine gewissen Zahl an Räumen und Gängen vom Spieler durchschritten werden können.
Zu beachten ist, dass der Spieler in der Nähe eines der zahlreichen Fenster des Hauptganges beginnt, da angenommen wird, dass Jack McNamara durch genau dieses in das Gebäude eingestiegen ist. Deshalb muss sich der Spieler auch dieses Fenster in Erinnerung behalten, da er nur durch dieses seine geraubten Kunstgegenstände sichern bzw. das Museum endgültig verlassen kann.

Das Ziel des Spieles ist nun das Entwenden von so vielen und wertvollen Kunstgegenständen wie nur möglich, ohne dabei natürlich auf das Verlassen des Gebäudes zu vergessen. Das Spiel endet nämlich genau nur dann, wenn der Spieler die Figur wiederum zu dem Einstiegsfenster zurück bewegt und das Verlassen durch Drücken der entsprechenden Taste anzeigt (falls die Figur die Begegnung mit dem Wachpersonal nicht übersteht, wird das Spiel natürlich auch beendet werden). Weiters kann der Spieler bereits entwendete Gegenstände auch bei diesem Einstiegsfenster durch Drücken einer weiteren Taste sichern, so dass die Figur in der Folge weiter Kunstgegenstände rauben kann, da das Gewicht der aktuell gesicherten Objekte nicht mehr in die allg. Gewichtsbeschränkung eingerechnet wird.
Dies ist deshalb interessant, da Jack McNamara natürlich nur eine bestimmte Menge von Gegenständen, abhängig von deren Gewicht, auf einmal tragen kann und danach keine weiteren Gegenstände mehr entwendet werden könnten.

Um die Entdeckung durch das Wachpersonal zu entgehen, sollte der Spieler seine Figur immer mit einem gewissen Abstand zu diesen bewegen, da sonst eine Entdeckung und eine Alarmauslösung unumgänglich sind, und die Sicherheitskräfte mittels Schusswaffengebrauch die Unzufriedenheit mit ihrer Anwesenheit zum Ausdruck bringen. Natürlich wird der Alarm auch automatisch ausgelöst, falls der Spieler zuerst auf einen Wächter schießt, da nach einem erfolgten Treffer der Wächter schmerzlich von dem uneingeladenen Besucher Kenntnis nehmen wird.
Jedoch hat der Spieler auch die Möglichkeit manuell den Alarm wieder zu deaktivieren, wobei man in jedem Raum einen entsprechenden Schalter finden wird, um tatsächlich den Alarm auszuschalten.

Der Alarm wird aber auch nach einem gewissen Zeitintervall automatische wieder deaktiviert.

2.1 Erläuterungen zu den Spielelementen

Während des Spieles werden einige Spielelemente am Bildschirm angezeigt, die dem Spieler Auskunft über wesentliche Spielinformationen geben:

- Lebensenergie:
  Der obere Balken am rechten, unteren Bildschirmrand zeigt dem Spieler die aktuelle Lebensenergie an, wobei die Farbe des Balken von rot (Normalzustand) bis hin zu blau (Lebensgefahr) sein kann.

- Gewichtsanzeige:
  Der untere Balken am rechten, unteren Bildschirmrand zeigt dem Spieler das aktuelle Gewicht an, welches die Figur aufgrund der entwendeten Gegenstände mit sich herum trägt. Falls der Balken einmal voll ausgefüllt ist, so ist die maximale Gewichtsbeschränkung erreicht und die Figur kann keine weiteren Kunstgegenstände mehr entwenden.

- Fadenkreuz:
  In der Mitte des Bildschirmes wird ein Fadenkreuz angezeigt, welches der Spieler zum Zielen seiner Schusswaffe verwenden kann.

- Score-Anzeige
  Am linken, oberen Bildschirmrand wird der aktuelle Punktestand angezeigt, der vom Wert der entwendeten Kunstgegenstände abhängig ist.

- Energieanzeige der Wächter
  Falls der Alarm durch den Spieler ausgelöst wird (sei durch zu große Annäherung oder durch gezielten Schuss auf Wächter), werden die Energiewerte der Wächter in der rechten, oberen Bildschirmecke angezeigt.

3 Wie wird das Spiel gestartet?

Durch Anklicken des Files GrandTheft.exe im Verzeichnis „\bin“ wird das Spiel geladen, wobei aufgrund des Generierens der benötigten Daten der Ladeprozess ein paar Sekunden dauern kann.
Nach Beendigung des Ladeprozesses wird das Menü von “The Grand Theft” angezeigt, wobei durch Auswahl (Cursor-Tasten + Return) des Punktes “Spielstart” das Spiel endgültig gestartet wird.

4 Spielende

Das Spiel kann im Allgemeinen auf drei unterschiedlichen Wegen beendet werden, wobei nur eine dieser drei Möglichkeiten als eine erfolgreiche Beendigung des Spiels anzusehen ist.

Diese erfolgreiche Spielbeendigung wird durch das Drücken der Taste ’q’ im Bereich des Einstiegsfenster (Ort des Spielbeginnes) aktiviert und bedeutet die Beendigung des

aktuellen Raubzuges und Rückkehr in das Hauptmenü.
Weiters kann de Spieler auch jederzeit mittels ESC in das Hauptmenü zurückkehren.
Die letzte Beendigungsmöglichkeit ist wohl der nicht zu wünschende Weg des Spielendes, nämlich falls einer der Museumswächter den Spieler tödlich mit seiner Waffe trifft (d.h. ausreichende Trefferzahl des Wächters, so dass die Lebensenergie nicht ausreichend ist),

5 Spielsteuerung

Die folgende Übersicht gibt Auskunft über alle möglichen Aktionen, der der Spieler mittels Maus- und Tastatursteuerung durchführen kann.

5.1 Aktionen mittels Maussteuerung

Wie auch bei üblichen Spielen dieser Kategorie wird durch die Bewegung der Maus die Blickrichtung bzw. Ausrichtung der Figur entsprechend verändert. D.h. durch das Bewegen der Maus kann der Spieler einerseits die Laufrichtung der Figur bestimmen und andererseits das Zielen mit der Schusswaffe durchführen.

Der linke Mausbutton hat weiters die Funktion des Abzuges der Schusswaffe, so dass nach Drücken dieser Maustaste ein Schuss abgegeben wird.
Die rechte Maustaste hat zu diesem Zeitpunkt der Spielentwicklung noch keine ausgezeichnete Bedeutung.

Während der Menüanzeige ist die Funktionsweise der Maussteuerung nicht aktiv.

5.2 Aktionen mittels Tastatursteuerung

- <u>während der Menüanzeige:</u>

Taste	***Aktion***
Tasten 'w' und 's' bzw. Cursortasten '↑' und '↓'	Menüauswahl wird entsprechen hinauf oder hinunter bewegt.
Enter im Hauptmenü	Die aktuelle Menüauswahl wird durchgeführt.
Enter im Submenü	Rückkehr in das Hauptmenü.

- <u>während des eigentlichen Spieles:</u>

Taste	***Aktion***
'w' oder '↑'	Figur wird vorwärts bewegt.
's' oder '↓'	Figur wird rückwärts bewegt.
'a' oder '←'	Figur wird seitlich nach links bewegt.
'd' oder '→'	Figur wird seitlich nach rechts bewegt.
'q'	Spiel wird beendet, falls Figur im Bereich

	des Einstiegsfensters ist.
'e'	Laufgeschwindigkeit der Figur wird verändert.
't'	Objekt, wenn nahe genug, wird entwendet. Der Alarm-Switch wird auch mittels dieser Taste gesteuert.
'g'	Alle momentan entwendeten Gegenstände werden gesichert, falls Figur in der Nähe des Einstiegsfensters ist.
'm'	Hintergrundmusik ein/aus.
F1	Anzeige der Hilfe während des Spieles (nicht im Menümodus!).
F2	Anzeige der Framerate (nicht im Menümodus!).
F3	Wireframe-Modus aktivieren/deaktivieren (nicht im Menümodus!).
F4	Umschalten zwischen bilinear/trilinear texturing
ESC	Spielabbruch und Rückkehr in das Hauptmenü, oder wenn bereits im Menümodus, Spielende.

6 Besonderheiten bei der Implementierung des Spieles

In diesem Abschnitt werden wir kurz auf die verwendeten Loader (respektive über die verwendeten File-Formate für Models), zusätzliche C++ Bibliotheken, Collision- und Shot Detection und verwendete Modeling-Tools näher eingehen.

Grundsätzlich wurde neben der graphischen Library OpenGL für die Visualisierung der graphischen Szenen als Interface zu dem Betriebssystem (das Spiel ist ausschließlich für das Windows-Betriebsystem erstellt worden) die C++ Library GLUT verwendet. Mittels GLUT kann somit vor allem der Windows-Manager konfiguriert werden (Auflösung: 1024x768, Bildgröße: Vollbild) bzw. die Event-Steuerung der Tastatur und Maus, welche beide als Inputgeräte für dieses Spiel konzipiert sind, integriert werden.

6.1 Bemerkungen zu den implementierten Features des Spiels

Im Bezug zu dem Spieldesign wurden einige Features wie Collision Detection mit den Levelwänden und den Objekten, Shot Detection für den Museumswächter und eine begrenzte Animationssteuerung integriert. Weiters werden auch gültige Reichweiten der zu stehlenden Objekte berechnet, so dass tatsächlich nur in einer bestimmten Entfernung das Objekt entwendbar ist. Ein zusätzliches Feature ist die Alarmsteuerung, die einerseits

automatisch nach einem gewissen Zeitintervall deaktiviert wird, jedoch auch entsprechende Schalter in jedem Ausstellungsraum zwecks manueller Steuerung existieren.
Natürlich wird auch darauf geachtet, dass sich der Wächter auch entsprechend wehren kann, das heißt er hat auch die Möglichkeit den Spieler zu neutralisieren (Trefferquote wird einfach halber mit einem Zufallsgenerator gesteuert).

Das Spiel enthält eine mehrere verschiedene triviale (mittels GLUT erstellt) und nicht triviale Objekte, welche entweder zu stehlen sind oder nur als Dekoration dienen bzw. animierte Objekte wie der Wächter. Zu den zu stehlenden Objekten, welche auch nicht trivial sind, zählen die Bilder (Bilderrahmen) und Schwerter. Nicht triviale, dekorative Objekte sind die Sockel, Tische und Sofas.

Hinsichtlich der Beschleunigung der Sichtbarkeitsberechnung wurde ein einfaches *View-Frustum-Culling* mittels *bounding spheres* für alle Objekte(außer Wächter) implementiert, wobei die Funktion *isSeeAble()* in *Objects.cpp* dafür verantwortlich ist.

Als Unterstützung des Gameplays wurde auch nicht auf die Soundausgabe vergessen, wobei einerseits Effekte aber auch eine Hintergrundmusikstücke ausgegeben werden. Dazu wird die C++ Library FMOD verwendet.

Zum Thema Experimentierung mit OpenGL wurden die Qualitätseinstellungen für die Texturberechung implementiert, weiters kann man in den WireFrame-Modus wechseln bzw. Framerate und Hilfe anzeigen lassen.
Zwecks Qualitätseinstellungen von Texturen werden entweder die Parameter *GL_LINEAR_MIPMAP_LINEAR* (triliniear) oder *GL_LINEAR_MIPMAP_NEAREST* (bilinear) für die Funktion *glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE _MIN_FILTER, …)* verwendet.

Leider konnten aufgrund des Zeitmangels, bedingt durch andere Lehrveranstaltungen, keine eigentlichen Spezialeffekte implementiert werden.

6.2 Verwendete Loader

Zwecks des Einlesens der Modeldaten in entsprechende Datenstrukturen werden insbesondere zwei unterschiedliche Loader verwendet, welche zum einen Daten im MilkShape ASCII Format *(mlkLoad.cpp)* und andererseits Daten aus dem md2-Format *(Md2.cpp)* lesen können.
Das gesamte Level und alle nicht animierten Objekte (exklusive den einfachen Objekten, welche mittels entsprechenden GLUT-Befehlen erstellt werden) werden im MilkShape ASCII Format verarbeitet, welches im Prinzip ein einfaches ASCII Textfile mit entsprechender Strukturierung der einzelnen Meshes mit dazugehörigen Texturkoordinaten/Texturen bzw. den Materials ist. Das animierte Wächter-Objekt wird aufgrund der besseren Unterstützung der Bones-Werte und der Gliederung von Keyframes im md2-Format bearbeitet.
Weiters ist noch zu erwähnen, dass beide Loader eine Datenstruktur verwenden, welche standardmäßig für die Verwendung von Vertex Arrays ausgerichtet sind.

Schließlich wurde noch ein eigener Loader für Texturen im *tga-Format* implementiert, so dass Texturen ohne Alpha-Werte mittels *auxDIBImageLoad* eingelesen werden, und Texturen mit zu verwendende Alpha-Werte unter Verwendung des *tga-Bildformates* mit dem entsprechenden Loader *(tgaLoad.cpp)* verarbeitet.

6.3 Zusätzliche verwendete C++ Libraries

Wie bereits oben erwähnt wird vor allem die GLUT Library 3.76 verwendet, um mit dem Betriebssystem kommunizieren zu können (zwecks Windows-Manager und Event-Handling). Die entsprechende GLUT-Version ist über die Seite http://www.xmission.com/~nate/glut.html downloadbar.

Weiters wird noch die Sound Library FMOD 3.71 für Windows integriert, um das Abspielen von *mp3*- und *wave-Audiofiles* zu ermöglichen.
Insbesondere die Hintergrundmusik (vollständige Sound Clips) wird als ein *Stream-Object* verarbeitet und wird nach Erreichen des Endes wieder von Vorne neu abgespielt *(loop playback)*. Der Link zu der entsprechenden Download-Seite von FMOD ist über die Webpage http://www.fmod.org/ zu erreichen.

Sonstige Funktionen des Spieles wurden tatsächlich in eigenen Source-Files implementiert, und verwenden außer den Standard Libraries aus C++ keine weiteren zusätzlichen Tools.

6.4 Bemerkungen zu einigen Funktionen des Spiels

Um das Gameplay des Spiels zu verbessern soll vor allem auf die integrierte Collision Detection und die Shot Detection bezüglich des Wächters erwähnt werden.

Die Collision Detection wird in Verbindung mit den Wänden des Levels und mit allen Objekten (insbesondere den Sockel-Objekten) angewendet. Im Kontext der Levelwände werden bezüglich der aktuellen Position der Kamera die Ebenengleichungen der Polygone, welche die Wände darstellen, berechnet und festgestellt, ob eben aufgrund der Kameraposition man sich vor oder hinter der entsprechenden Wand befindet (in *Camera.cpp*). Bei den Objekten wird dies wesentlich einfacher implementiert, wobei die Sockeln mit Hilfe eines b*ounding parallelepiped* geprüft werden und bei dem Wächter-Objekt eine *bounding sphere* verwendet wird.

Auch bei der Shot Detection umgeben mehrere kleinere *bounding spheres* in einer vertikalen Ausrichtung den Wächter und mittels der Kameraausrichtung *(camera view)* werden mögliche Schnittpunkte auf den *bounding spheres* errechnet.

Schließlich sei noch hinzuweisen, dass die Frage, ob der Spieler in Reichweite eines zu stehlenden Objektes oder des Alarm-Switches beim Drücken der Taste *'T'* ist, auch mittels *bounding spheres* beantwortet wird. Ist der Spieler innerhalb der entsprechenden *bounding sphere*, so kann er den Gegenstand auch tatsächlich stehlen kann.

6.5 Verwendete Modeling Tools

Als einziges Modellierungsprogramm wurde MilkShape verwendet, wobei sowohl alle

nicht animierte Objekte (wie Sockeln, Schwerter) als auch die Animation des Wächters modelliert worden sind.
Zu erwähnen sei noch, dass alle Animationen des Wächters aus eigener Produktion stammen, d.h. das die Bones-Werte bzw. die Erstellung der Keyframes selbst gemacht wurden. Der Wächter selbst stammt jedoch aus fremder Hand.
Weiters wurde die Animation in ein md2-Format abgespeichert, alle sonstigen Objekte wurden in einem MilkShape ASCII Format gespeichert.

6.6 Implementierte Spezialeffekte

Aufgrund des Zeitmangels, welcher durch andere Lehrveranstaltungen bedingt wurde, konnten wir LEIDER keine Spezialeffekte implementieren.