Oliver Kayser-Herold, Andreas Keese, Markus Krosche,
Martin Krosche, Oliver Pajonk

Teilprojekt Achterbahnsimulator

Aufgabenbeschreibung für das Softwareentwicklungspraktikum am Institut für Wissenschaftliches Rechnen

Institut für Wissenschaftliches Rechnen
Carl-Friedrich-Gauss-Fakultät
Technische Universität Braunschweig
Version: 1.0.3 für das Sommersemester 2009
Revision: 2284
Erstellt am 30. März 2009

Braunschweig, 2009

Inhaltsverzeichnis

1 Einleitung

Das Institut für Wissenschaftliches Rechnen stellt zwei gekoppelte Projektthemen für das Softwareentwicklungspraktikum (SEP) 2009. Zum einen soll ein CAE-Simulator[1] entwickelt werden, der eine Achterbahn vereinfacht, aber mathematisch korrekt simuliert und dreidimensional visualisiert (Teilprojekt „Achterbahnsimulator“). Zum anderen soll ein CAE-Editor zur komfortablen, ingenieursmäßigen Konstruktion von Achterbahnkurven implementiert werden (Teilprojekt „Achterbahneditor“).

Dieses Dokument enthält die Aufgabenbeschreibung und Anforderungsspezifikation für den *Achterbahnsimulator*. Die Grundlagen der mathematischen Modellierung finden sich in [1]. Zudem ist es unerlässlich, dass sich beide Gruppen zusätzlich mit der jeweils anderen Aufgabenbeschreibung vertraut machen, um das Gesamtprojekt zu verstehen und auf ein gemeinsames Ziel hinarbeiten zu können.

Zusätzliche Informationen zur Organisation finden sich auf `http://www.wire.tu-bs.de` und der entsprechenden Informationsseite des Insituts für Software Systems Engineering (SSE), `http://www.sse-tubs.de/teaching/ss09/sep`.

2 Über das Dokument

Dieses Dokument enthält viele, im Normalfall blau gekennzeichnete, *Hyperlinks*. Benutzen Sie zuerst diese, um bei Fragen zusätzliche Hilfen oder einfach weiterführende und vertiefende Informationen zu erhalten.

3 Aufgabenstellung

Die Aufgabe dieses Teilprojektes ist der softwareingenieursmäßige Entwurf und die Implementierung eines CAE-Visualisierers und -Simulators für Achterbahnen.

In Abbildung 1 ist ein eher auf Unterhaltung ausgerichteter Simulator[2] beispielhaft abgebildet, der allerdings die wesentlichen Elemente der Visualisierung beinhaltet. Bei der **Visualisierung** soll der Streckenverlauf (die Geometrie) in einer dreidimensionalen Umgebung dargestellt werden. Dies soll dem Konstruktionsingenieur ermöglichen, Fehler oder Probleme in der Streckenführung frühzeitig zu erkennen und mit Hilfe des Editors [2] zu beheben. Bei der **Simulation** soll in dieser dreidimensionalen Darstellung des Streckenverlaufs eine Simulation einer Achterbahnfahrt aus der Zugperspektive erfolgen. Die 3D-Visualisierung und -Simulation ist genauer in Abschnitt 4.1 beschrieben.

Ein wichtiger Aspekt beim Entwurf einer Achterbahn ist die maximale Belastung der Passagiere, die sich aus dem pyhsikalischen Modell während der Simulation leicht berechnen lässt. Um diese parallel zum Fahrtverlauf in der Zugperspektive analysieren zu können, soll der **Betrag der Beschleunigung** in einem einfachen Grafikfenster während der Simulation geplottet werden. Eine genauere Beschreibung findet sich in Abschnitt 4.2.

4 Beschreibung der Visualisierung

Im Folgenden werden die Anforderungen an die Visualisierung beschrieben.

[1] CAE = *Computer Aided Engineering*

[2] Natürlich muss die in diesem Praktikum entstehende Visualisierung nicht die hier gezeigte Komplexität in der grafischen Darstellung aufweisen.

Abbildung 1: *Beispiel eines Achterbahnsimulators (Screenshot des NoLimits Rollercoaster Simulators, http://www.nolimitscoaster.com)*

4.1 Dreidimensionale Visualisierung

Die Streckendarstellung der Bahn soll halbwegs realitätsgetreu erfolgen. Das heißt nicht, dass Stützen o.ä. visualisiert werden müssen. Mindestanforderung sind jedoch zwei Schienen, die in gleichem Abstand zueinander liegen und der Winkel/die Verdrehung zwischen diesen. Weiterhin müssen die Schienen in jedem Abschnitt durch mindestens zwei Querbalken verbunden sein, da sonst kein Gefühl für die Geschwindigkeit aufkommt. Ebenso sollte die x-y-Ebene auf Höhe $z = 0$ visualisiert werden; das entspricht dem Boden, auf dem die Bahn aufgebaut ist. Diese Ebene sollte einen Horizont bereitstellen und damit die Orientierung im Raum erleichtern.

4.1.1 Ein Darstellungsproblem

Die Verdrehung der Schienen entspricht in der Simulation der Orientierung der mitfahrenden Kamera. Da wir, wie in [1] erläutert, die Achterbahn als Punktmasse auffassen, fehlt in dem beschriebenen Modell eine solche Verdrehinformation. Ein einzelner Wert, wie beispielsweise ein einzelner Winkel für die Kameraorientierung relativ zur Bahn, reicht nicht aus: in Loopings gäbe es Probleme im punktuellen 90 Grad Anstieg/Abstieg, welche typischerweise zu einem spontanen Springen der Kameraorientierung um 180 Grad führen. Zudem könnte man einen Looping von einem kontinuierlichen 90 Grad Anstieg ohne Überschlag nicht unterscheiden. Dies muss aber möglich sein, da bei einem Looping der Wagen zeitweise auf dem Kopf ist – bei dem angesprochenen 90 Grad Anstieg aber nicht.

4.1.2 Die Lösung: ein zusätzlicher Vektor

Um also alle Kurvenelemente auch visuell darstellbar zu machen, benötigt man demnach mehr als einen Wert. Im Dreidimensionalen sind es gleich drei zusätzliche Werte pro Stützstelle. Dies kann man sich so erklären: führt man ein *lokales Koordinatensystem*, bestehend aus drei Vektoren, mit der Bewegung der Punktmasse mit, so ist die Orientierung stets eindeutig. Diese Vektoren sind der Geschwindigkeitsvektor (*Rollachse*, tangential zur Bahn), der Schwellenvektor (*Nickachse*, in

Richtung der Querbalken) und der sich durch das *Kreuzprodukt* ergebende Vektor (*Gierachse*). Der Geschwindigkeitsvektor wird durch die Simulation berechnet und ist somit bekannt. Da einer der drei Vektoren aus dem Kreuzprodukt der anderen beiden resultiert, reicht die Vorgabe eines zusätzlichen Vektors aus: entweder dem Schwellenvektor oder dem Giervektor. Diesen muss der Bahnkonstrukteur für einen Stützpunkt während des Erstellens der Bahn stets mit angeben. In diesem SEP verwenden wir den Giervektor, da dieser vom Benutzer in einem Editor intuitiv gesetzt werden kann.

Eine Interpolation zwischen den Giervektoren der Stützstellen erfolgt *analog zu den Stützstellen selbst*. Mit Hilfe von [1] sollten Sie in der Lage sein die nötigen Erweiterungen zum mathematischen Modell zu designen.

Die Einführung des Giervektors hat Auswirkungen auf *beide Teilprojekte* und auf das XML-Dateiformat. Diese Auswirkungen, welche alle Phasen des Projekts betreffen, müssen von den SEP-Gruppen bedacht werden.

4.2 Visualisierung der Beschleunigung

Die auf die Passagiere während der Achterbahnfahrt wirkende Beschleunigung soll während der Simulation in einem kleinen 2D-Fenster angezeigt werden. In vertikaler Richtung soll der Betrag der Beschleunigung, in horizontaler Richtung die Zeit aufgetragen werden. Hinweis: eventuell kann diese Anzeige auch für das Debugging des Modells nützlich sein.

5 Technische Produktumgebung

Im Folgenden ist die technische Produktumgebung gegeben. Es stehen im wesentlichen zwei grundsätzlich unterschiedliche Varianten zur Auswahl.

Programmiersprache Zur Wahl stehen *Java* oder *ISO/IEC C++*. Aufgrund der Erfahrungsgemäß kürzeren Entwicklungszeit wird Java empfohlen. Die Referenz für Java ist das *Sun JDK 6*, die Referenz für C++ ist die *GNU Compiler Collection 4.3*. Für C++ ist zusätzlich Kompatibilität zum *Intel C++ Compiler 11.0* wünschenswert (sprich: optional).

Betriebssystem Eine aktuelle 32bit Linux Distribution (z.B. *OpenSuSE*, *Debian*, *Ubuntu*,...) mit *Kernel 2.6*. Wünschenswert ist 32- *und* 64-Bit Kompatibiliät.

Hardware Aktueller, *x86* oder *Intel 64* / *AMD64*-basierter Standard-PC (z.B. *Intel Core 2*) mit hinreichend leistungsfähiger 3D-Grafikkarte

Testwerkzeuge *JUnit* oder *CppUnit*

Tools/Bibliotheken *Java3D* oder *OpenInventor* und eventuell weitere - hier ist eine Analyse notwendig

6 Arbeitsumgebung

Die Entwicklungs-, Dokumentations und Projektmanagementwerkzeuge, welche benutzt werden sollen, sind die folgenden:

Dokumentenerstellung *OASIS Open Document Format*, *Microsoft Word* oder *LaTeX 2ε*, wobei die Empfehlung aufgrund der weitaus besseren Teamarbeitsmöglichkeiten in Verbindung mit Subversion klar bei LaTeX 2ε liegt. Die Abgabe der Dokumente erfolgt im *PDF Format*.

Entwicklungsumgebung *Eclipse* oder *KDevelop* (je nach Geschmack und Programmiersprache)

Versionsverwaltung *Subversion* (ausschließlich, siehe auch Abschnitt 10)

Build-Werkzeug *Apache Ant*/*Apache Maven* oder *GNU Make*

Projektverwaltung *Trac*, eventuell mit zusätzlichen *Plugins*

7 Einsatz

Die zu entwickelnde Software soll in vollem Umfang auf dem Betriebssystem Linux (Details siehe Abschnitt 5) lauffähig sein und mit dem Editorwerkzeug des anderen Projektes problemlos zusammenarbeiten.

8 Qualitätsanforderungen

Auf Exaktheit in der Anzeige wird Wert gelegt, da es sich um ein Designwerkzeug für Ingenieure handelt. Grafische Opulenz ist nur da wichtig, wo die Visualisierung der Ergebnisse unterstützt wird. Eine möglichst einfache Installation mit ausführlicher Beschreibung ist wichtig.

Der Simulator/Visualisierer soll ausreichend stabil laufen. Datenverlust und somit Verlust von Arbeit ist in jedem Fall zu vermeiden. Eventuell sind besondere Optimierungen bezüglich der Laufzeiteffizienz nötig. Das Produkt muss auf ausreichend leistungsfähigen, „normalen“ Rechnern in vollem Umfang lauffähig und benutzbar sein (siehe Abschnitt 5).

Das Programmdesign soll bedingt für eine spätere Erweiterung ausgelegt sein. Die Programmstruktur soll mit wenig Zeitaufwand für Außenstehende verständlich sein. Die Arbeit im (Projekt-)Team soll durch die Architektur der Software ermöglicht und unterstützt werden. Nötige Kommentare sollen nach Javadoc/Doxygen Vorgaben erfolgen, um eine möglichst automatische und somit konsistente Dokumentation zu erzeugen. Programmcode und Kommentare sollen, wie in Industrie und Forschung üblich, in Englisch verfasst werden.

Zum Testen der Klassen und Module sollen Unit-Tests, unter der Verwendung einer geeigneten Unit-Test-Bibliothek, herangezogen werden. Bei der Wahl der zu verwendenen Tools/Bibliotheken soll darauf geachtet werden, dass diese mit hoher Wahrscheinlichkeit auch in Zukunft von den Herstellern gepflegt werden.

9 Dokumentation

Zugehörige schriftliche Ausarbeitungen können in deutsch oder englisch verfasst werden. Hierzu wird ausdrücklich LaTeX2e empfohlen, es sind aber auch andere Formate möglich (siehe Abschnitt 6). Vom SSE werden Vorlagen für alle Dokumente bereitgestellt, welche auch zu benutzen sind.

10 Ergänzungen

- Der Einsatz von Subversion als Versionsverwaltungswerkzeug ist *bindend* und muss in der folgenden Weise erfolgen: alle primären (d.h. nicht automatisiert erstellbaren) Artefakte sind in dem bereitgestellten Repository zu verwalten. Dies schließt die folgenden Dinge ein (ist aber nicht beschränkt auf diese): Programmcode; Dokumente bzw. Quelltexte für Dokumente; Grafiken; Prototypen; Diskussionspaper; Präsentationen. Automatisiert erstellte Artefakte dürfen nur aus besonderem Grund ins Repository eingespielt werden. Dies schließt die folgenden Dinge ein (ist aber nicht beschränkt auf diese): Ausführbare Programme und Bibliotheken, welche aus dem Quelltext des Projekts erstellt werden; Temporäre Dateien und Zwischendateien; generierte PDFs. Ein besonderer Grund *für* das Einspielen von sekundären Artefakten ist z.B. das Ablegen des PDFs einer Abgabeversion eines Dokuments.
- Der Inhalt und die zeitliche Entwicklung des Subversion Repositories werden mit zur Bewertung des SEPs herangezogen.

- Die in diesem SEP entstehende Software und Dokumentation soll unter GPL oder LGPL Lizenz (je nach Programmierumgebung) gestellt werden um für spätere Präsentationen zur Verfügung zu stehen.

Schließlich möchten wir anregen, dass Sie, falls Sie Spaß daran haben und die Zeit reicht, selbstständig die Aufgabenstellung erweitern – hierzu ein paar Vorschläge:

- Visualisierung von Stützen
- Eine Warnung (grafisch oder über ein Tonsignal), wenn die auf die Passagiere wirkende Beschleunigung einen Schwellenwert überschreitet
- Ausgabe von weiteren technischen Daten, wie z.B. momentane Geschwindigkeit, Höhe über Null, Simulationszeit, ...
- Möglichkeit, Bildschirmfotos anzufertigen
- Möglichkeit, einen Film einer Achterbahnfahrt zu erstellen
- Erweiterungen des dynamischen Modells, siehe auch [1]
- Visualisierung aus Perspektive einer dritten Person (z.B. Verfolger des Zuges)
- Hinzufügen von weiteren Typen von Streckenelementen (z.B. Tunnel, Aufzüge)
- Mehrere Wagen auf der Strecke, die kollidieren können
- Visualisierung eines Himmels mit Texturen
- Visualisierung eines nicht ebenen Bodens
- Schönere Schienen

Falls Ihre Simulation hierfür weitere Parameter benötigt, können Sie diese aus weiteren XML-Dateien auslesen.

Literatur

[1] Kayser-Herold, Oliver, Andreas Keese, Markus Krosche, Martin Krosche und Oliver Pajonk: *Achterbahn-Editor/-Simulator - Dokumentation für das Softwareentwicklungspraktikum am Institut für Wissenschaftliches Rechnen*, März 2009.

[2] Krosche, Martin und Oliver Pajonk: *Teilprojekt Achterbahneditor - Aufgabenbeschreibung für das Softwareentwicklungspraktikum am Institut für Wissenschaftliches Rechnen*, März 2009.