Projektarbeit

Fahrsimulator

Sandro Ropelato (ropelsan)
Christof Würmli (wurmlchr)

11. Dezember 2011

Studiengang: Systeminformatik SI
Betreuender Dozent: Peter Früh (frup)

Inhaltsverzeichnis

1. Einleitung

1.1. Ausgangslage

Im Gebiet der Fahrsimulatoren gibt es bereits eine Vielzahl von verschiedenen Lösungen. Einige davon bestehen aus Filmmaterial, das abgespielt wird und der Fahrer muss auf die Bremse drücken sobald ein bestimmtes Ereigniss eintritt. Andere Fahrsimulationen bringen bereits eine virtuelle Welt mit, in der man sich mehr oder weniger frei bewegen bzw. frei fahren kann. Jedoch sind bei den meisten von diesen Fahrsimulatoren bereits feste Szenarien implementiert die nicht geändert werden können.

Die Grenzen eines Fahrsimulators liegen vor allem in der Leistungsfähigkeit des Rechners auf dem die Simulation installiert werden soll.

Das Projekt wird in Zusammenarbeit mit der ETH Zürich durchgeführt. Es ist bereits eine LabView Schnittstelle für das Steuerrad vorhanden.

1.2. Aufgabenstellung

1.2.1. Formulierung

Das Ziel der Arbeit besteht darin, einen Fahrsimulator für die bestehende Simulatinsumgebung zu erstellen. Diese besteht aus einem Fahrercockpit, einer Leinwand, einem Projektor und einem Computerterminal, von dem aus die Simulation gesteuert werden kann. Das Fahrercockpit enthält ein Steuerrad, drei Pedalen, einen Schaltknüppel und einen Autositz mit Sicherheitsgurt.

Die Fahrsimulation sollte dem Benutzer die Illusion des Autofahren möglichts realistisch vermitteln. Die soll durch einen Einsatz von Karteninfomationen von Google Maps oder Google Street View unterstützt werden.

Zudem sollen alle Betriebszustände und Benutzereingaben registriert und aufgezeichnet werden um eine genaue Analyse zu ermöglichen.

1.2.2. Aufteilung der Arbeit

Wir teilten die Arbeit im Wesentlichen in zwei Teile auf. Im ersten Teil legten wir den Fokus auf die korrekte Ansteuerung des Cockpits. Um dies zu testen, setzten wir uns das Ziel, ein Video abzuspielen und mit Gas- und Bremspedal die Geschwindigkeit kontrollieren zu können. Im zweiten Teil folgte die Implementation und Installation der Fahrsimulation.

1.3. Zeitplan und Arbeitsteilung

2. Aufbau des Systems

2.1. Systembeschreibung

Der Aufbau des Systems für den Fahrsimulator wird anhand der Abbildung 1 ilustriert. Die blau markierten Komponenten werden im Rahmen dieser Projekt Arbeit entwickelt.

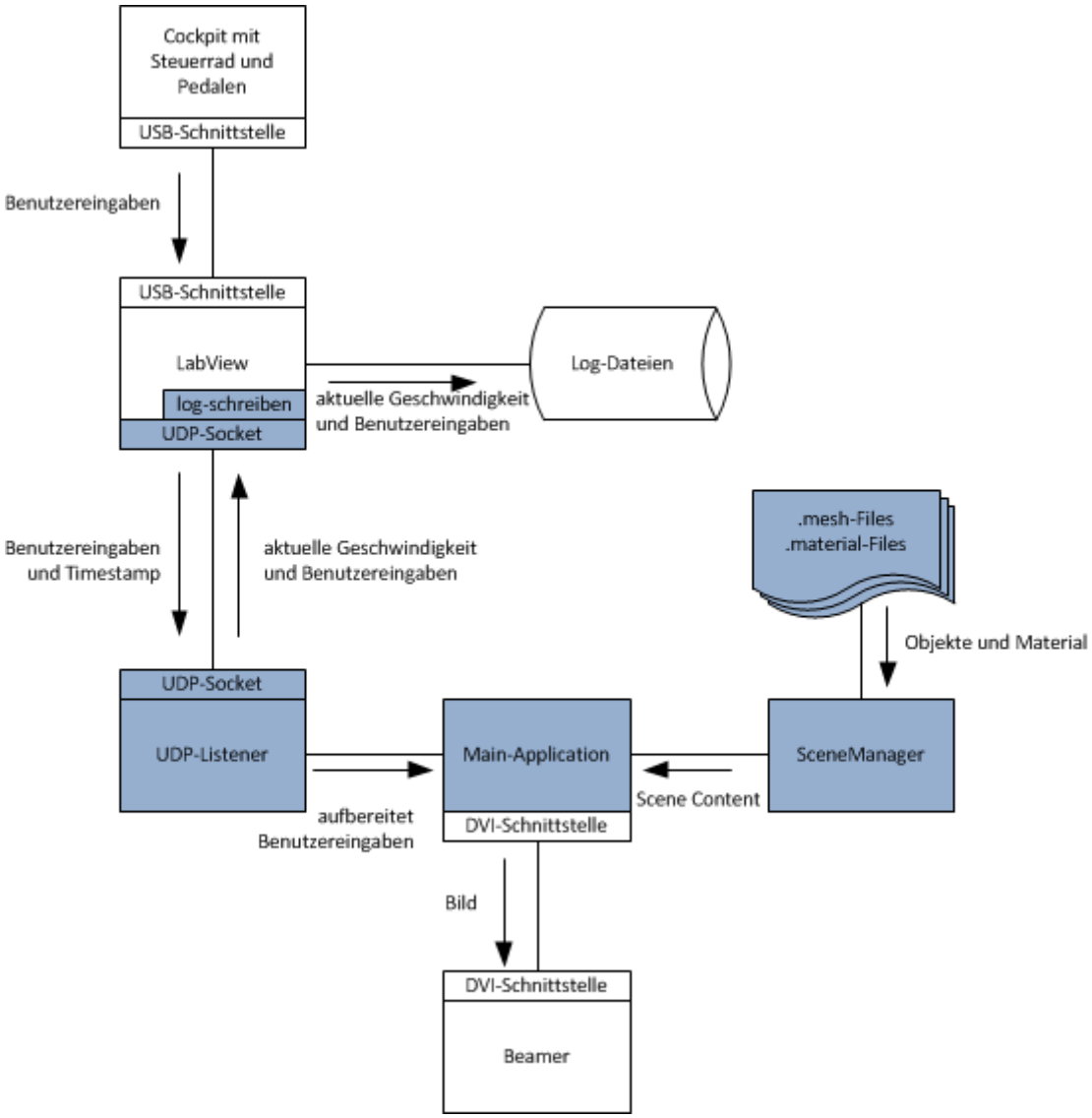

Abbildung 1: Systembeschreibung

Alle übrigen sind bereits vorbestehend. Benutzereingaben, die im Cockpit gemacht werden, werden von einem LabView Programm eingelesen. Nun benötigt es einen UDP-Port, über den verschiedenen Eingaben an das Programm weitergeleitet werden. Es handelt sich hierbei um Werte, die das Drehen des Steuerrades und den Druck auf Gas- oder Bremspedal quantifizieren. Zusätzlich wird der UDP-Port auch für das Empfangen diverser Log-Daten, die von unserem Programm gesendet werden, verwendet. Damit die Empfangenen Daten sauber in ein Log-File geschrieben werden, wird das Lab-View Programm erweitert. Weiter muss in Programmiersprache C einen UDP-Socket mit entsprechendem UDP-Listener implementieren werden, um die Benutzereingaben zu empfangen. Gleichzeitig wird der UDP-Listener dazu verwendet die Geschwindigkeit des Fahrzeugs sowie Timestamps und weitere Daten an das Lab-View Programm zurück zu schicken. Damit können die Daten gespeichert und später ausgewertet werden.
Diese Aufteilung durch eine Netzwerkschnittstelle ermöglicht es, das System, wenn notwendig, zu dezentralisieren. Einfahheitshalber wurde der UDP-Listener erst in einem Video-Beispiel implementier und getestet (Siehe Anhang B). Nachfolgend wird dieser UDP-Listener auch in das Programm des Fahrsimulator transferiert. Sind die Daten vom UDP-Listener empfangen und aufbereitet, werden sie im Hauptprogramm (Main Application) weiter verwendet. Während die Position des Steuerrades, des Gas und Bremspedales vom UDP-Listener permanent an das Hauptprogramm übertragen werden, wertet dieses die Positionen aus und veranlass die entspechenden Aktionen in der geladenen Szene. Die Szene selbst wird von einem Szenen-Manager geladen. Dieser benötig für die zu ladenden Objekte ein Meshfiles und mindestens ein Materialfile. Die Form jedes Objektes in der Szene wird durch ein separates Meshfile definiert. Texturen und Materialien werden durch ein oder mehrere Materialfiles beschrieben. Ein Materialfile kann zur Beschreibung unterschiedlicher Objekte verwendet werden. Die berechnete Szene wird schlussendlich in einem Fenster von Hauptprogramm angezeigt und über eine DVI-Schnittstelle an den Beamer übertragen. Der Beamer projeziert das Bild an die Wand, die sich direkt vor dem Cockpit befindet.

2.2. Anforderungen

2.2.1. Funktionelle Anforderungen

2.2.2. Nicht Funktionale Anforderungen

a) Zuverlässigkeit

b) Benutzbarkeit

c) Aussehen und Handhabung

d) Wartbarkeit

e) Zeitliche Anforderungen

Lab-View Lab-View Programm Log-Datei Cockpit