Spherical Maps

Bearbeitet von
Andreas Petermann MNr.: 162586
Danilo Gulamhussene MNr.: 162795
Sascha Baldt MNr.: 162773

als Bildinformationstechik-Praktikum im Wintersemester 2005/2006

1 Selbst gestellte Aufgabe

Implementierung einer Anwendung, die es ermöglicht aus einer Reihe von Einzelbildern, eine spherical map zu erstellen.

Die zu implementierende Anwendung soll es dem Anwender ermöglichen, die Einzelbilder in geeigneter Weise anzuorden. Anhand der Anordnung sollen die Einzelbilder verzerrt und in einer einzelnen Spherical Map zusammengefasst werden.

Annahme:

Die Einzelbilder wurden von einem Punkt aufgenommen und decken die komplette Umgebung ab.

2 Umsetzung

Das Projekt wurde in Squeak, einer auf Smalltalk basierenden Entwicklungsumgebung umgesetzt.

2.1 Programmstruktur

Um für eine gute Erweiterbarkeit des Programmes und Wiederverwendbarkeit einzelner Bestandteile zu sorgen, wurde auf eine Objektorientierte Programmierweise geachtet. Die Struktur des Programms wurde vor Beginn der Umsetzung in einem UML-Diagramm entworfen, um die Zusammenarbeit an dem Projekt zu erleichtern.

Das geschriebene Programm gliedert sich in zwei Hauptbestandteile. Diese sind ein graphisches User Interface und ein Renderer. Die Gesamtstruktur

des Programms ist in Abbildung 1 dargestellt, wobei graue Bestandteile nicht implementiert sind, aber mögliche Erweiterungen wären.

2.2 Graphisches User Interface

Das Graphische User Interface dient dazu, die vielen Einzelbilder wieder korrekt anzuordnen. Das heist, für jedes Bild wieder den korrekten Breitenwinkel, Höhenwinkel, Drehwinke und die Öffnungswinkel anzugeben. Diese Werte können alle numerisch eingegeben werden. Zusätzlich kann der Breiten und Höhenwinkel auch durch verschieben der Bilder mit der Maus geändert werden.

Um das Verfahren zu beschleunigen, wurde eine Multiselektion der Bilder implementiert, wodurch sich Einstellungen gleichzeitig für mehrere Bilder vornehmen lassen.

Um die Genauigkeit der Anordung zu erhöhen, wurde eine Zoomfunktion und ein halbautomatisches Matching integriert. Beim Matching kann der Nutzer in zwei Bildern übereinstimmende Punkte angeben, worauf diese beiden Bilder von dem Programm so angeordnet werden, daß die angegebenen Punkte übereinander liegen (siehe rote Quadrate in Abbildung 2). Die Zoomfunktion ermöglicht es dem Nutzer, diese Punkte genauer anzugeben.

2.3 Renderer

Auf Grund der im User Interface gewonnenen Daten können die Einzelbilder in einer dreidimensionalen Scene angeordnet werden. Es ergibt sich eine kreisförmige Anordnung.

Es wurde ein Raytracer implementiert, um diese Scene abzutasten und in einem einzelnen Bild zusammenzufassen. Die Abtastung erfolgt vom Mittelpunkt der Scene, entsprechend eines vorgegebenen Mappings. Bisher wurde das Spherical Mapping verwendet. Der Renderer ist aber um weitere Mappings erweiterbar.

Für den Raytracer war die Implementierung von Schnitttests, und Texturmapping notwendig. Um die allgemeine Qualität des Renderers zu verbessern, wurde lineare Texturfilterung und adaptives Antialiasing implementiert.

Eine spezielle Optimierung für das Projekt Spherical Maps, war die Implemetierung eines Blendings zwischen benachbarten Einzelbildern (siehe Ab-

bildung 6).

3 Ergebnisse

Die Aufgabenstellung konnte im gegebenen Zeitrahmen umgesetzt werden. Die erzielten Ergebnisse nach einzelnen Entwicklunstufen des Programms sind in den Abbildungen 3 bis 6 dargestellt.

Die schwarzen Stellen, die selbst in der letzten Entwicklungstufe noch zu sehen sind stammen von einer ungenügenden Abdeckung der Umgebung durch Photos, und sind nicht durch einen Fehler im Programm entstanden.

Dennoch lässt die bisherige Implementierung viel Platz für Verbesserungen, vor allem bei der Geschwindigkeit des Verfahrens. So wird für das Anordnen der Bilder noch ca. eine Stunde benötigt und das Rendering einer Spherical Map in einer Auflösung von 2048x1024 benötigt auf einem 1,5 GHz-Prozessor ebenfalls ca. eine Stunde. Auf die möglichen Verbesserungen des Programms soll noch einmal genauer im folgenden Kapitel 4 eingegangen werden.

4 Ausblick

Das Matching sollte weiter verbessert werden. Es beruht bis jetzt nur auf einem einzelnen Matchingpunkt pro Bild. Für ein Eindeutiges Matching mit allen Freiheitsgraden sind aber mindestens zwei solcher Punkte pro Bild notwendig.

Um das Anordnen der Bilder zu beschleunigen könnte das finden von Matchingpunkten automatisiert werden.

Eine weitere Verbesserung wäre die Beschleunigung des Renderers. Zum einen könnten Algorithmen zur Optimierung der Schnitttests eingesetzt werden (z.B. Octree Scenenunterteilung). Die größte Beschleunigung (Faktor 10-50) sehen wir aber in einer hardwarenäheren Implementierung rechenintensiver Programmteile in C++.

Weitere Features wären die Unterstützung zusätzlicher Mappings (z.B. Cubemapping), eine 3D-Visualisierung zur besseren Anordnung der Bilder durch den Nutzer, da in der 2D-Ansicht die Verzerrungen nicht dargestellt werden und eine Unterstützung zur Generierung von High-Dynamic-Range-Images.

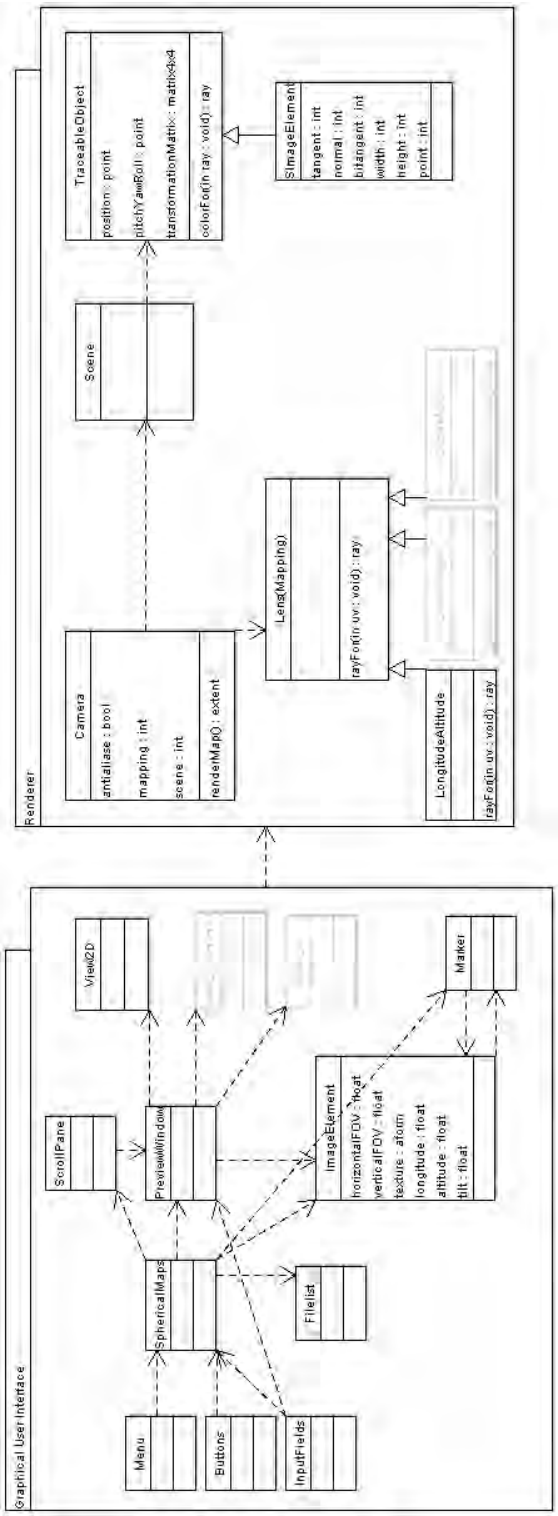

Fig. 1: UML Diagramm zu Spherical Maps

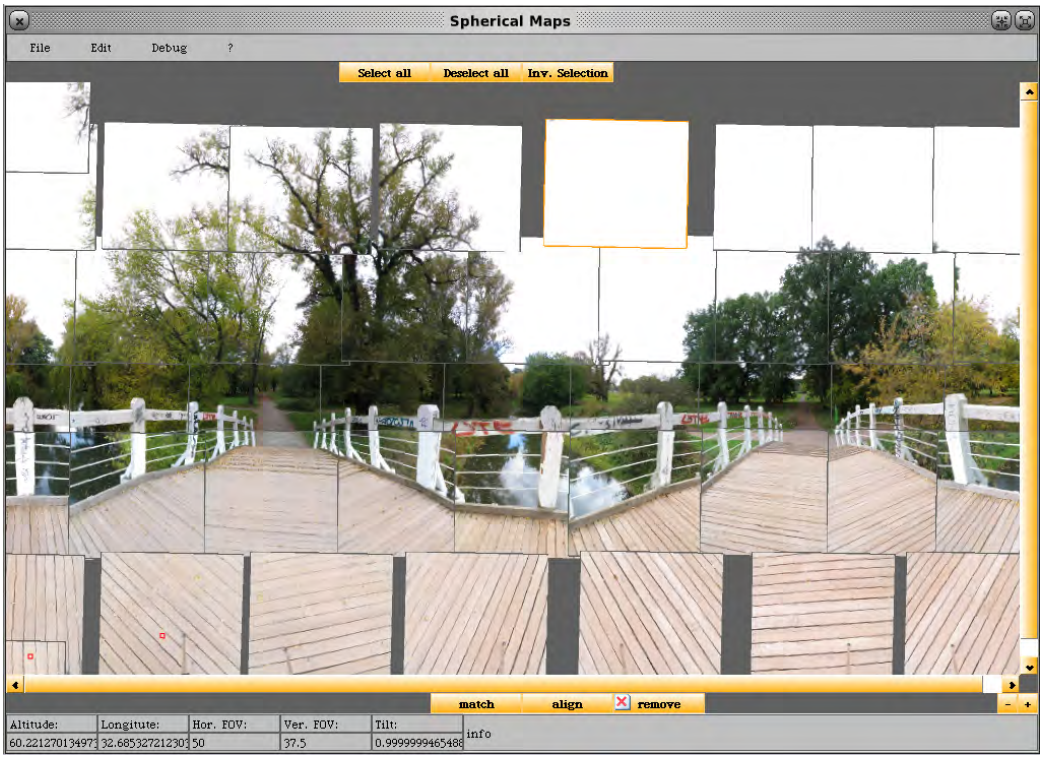

Fig. 2: Screenshot des graphischen User Interface mit bereits angeordneten Einzelbildern

Fig. 3: normal

Fig. 4: matching

Fig. 5: matching + zoom

Fig. 6: matching + zoom + blending