urn:nbn:de:gbv:519-thesis 2008-0112-7

...meinem Vati

Hochschule Neubrandenburg
University of Applied Sciences

Studiengang Geoinformatik

Diplomarbeit zum Erlangen des akademischen Grades Diplomingenieur (FH)

3D Modellierung
mit der Open Source Software Blender

urn:nbn:de:gbv:519-thesis2008-0112-7

vorgelegt von:
Marco Lerm Mat.Nr. 281504
Neubrandenburg, den 20.10.2008

Betreuer: Prof. Dr.-Ing. Hillmann
Zweitprüfer: Dipl.-Ing. Winck

Erklärung der Selbstständigkeit

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Neubrandenburg, den 20.10.2008 .....

Kurzfassung

Diese Diplomarbeit beschreibt die Möglichkeiten der Modellierung eines Gebäudes mit der Open Source Software Blender. Weiterhin beschreibt sie die Verwendung des Blender internen Partikelsystems, die Verwendung von Audioeffekten für Animationen sowie die Nutzung der Blender internen Game-Engine.
Vorweg werden in Kapitel 2 allgemeine Grundlagen der 3D Modellierung wiedergegeben. Ferner gibt Kapitel 2 Aufschluss über weitere Modellierungsprogramme, welche zu Animationsarbeiten herangezogen werden können. Des Weiteren werden Softwarelösungen vorgestellt, welche ganz speziell zur Auswertung, Planung und Entwicklung eines 3D Stadtmodells entwickelt wurden. In Kapitel 3 wird die Blender interne Game-Engine diskutiert. Zusätzlich erfolgt eine Gegenüberstellung des Modellierungsprogramm Cinema4D gegenüber Blender, mit dem Hintergrund 3D Grafiksoftware vs. Game-Engine. Kapitel 4 gibt Aufschluss über die Modellierung des benannten Gebäudes mit Darstellung aufgetretener Mängel. In Kapitel 5 werden Bilder des Modells mit Bildern aus der Realität gegenübergestellt. Abschließend wird in Kapitel 6 Fazit und Ausblick wiedergegeben.

Abstract

This thesis describes the possibilities of modeling of a building with the Open Source Software Blender. Furthermore this work describes the use of the Blender internal particle system, the use of audio effects for animation and the use of the Blender internal game engine.
Beforehand general bases of 3D modelling are returned in chapter 2. Further chapter 2 gives explanation about other modelling programmes which can be pulled up for animation work. Besides the software solutions which were developed quite especially for the evaluation, planning and development of 3D town model are introduced. In chapter 3 the Blender internal game engine is discussed. In addition, a confrontation of the modelling programme Cinema4D occurs compared with Blender, with the background 3D graphic software vs. game engine. Chapter 4 gives explanation about the modelling of the named building with representation of appeared defects. In chapter 5 are pictures of the model with images from the reality compared. In the end, chapter 6 describes the conclusions and outlook.

Inhaltsverzeichnis

1 Einleitung

Seit der Bereitstellung des virtuellen Stadtmodells von Berlin im März 2007 stieg der Bekanntheitsgrad von virtuellen Stadtmodellen. Nahezu jede Stadt strebt seitdem eine virtuelle Präsentation im Internet an. Oftmals wird den fachunkundigen Nutzer suggeriert virtuelle Stadtmodelle sind eine Erfindung der vergangenen 2 Jahre. Jedoch fanden Stadtmodelle bereits im 16. Jahrhundert ihre erste Anwendung, als Holzschnitte und Kupferstiche, zu finden sind sie oftmals noch in Rathäusern oder bei Architekten, zum Beispiel als Modell für neue Baugebiete. In der Neuzeit ließ Berlin in einem Pilotprojekt ein erstes virtuelles Stadtmodell anfertigen, welches vorerst nur für Wirtschaft und Verwaltung zugänglich war. Erst mit Verbesserung der notwendigen Computertechnik sowie der erforderlichen Software fanden virtuelle Stadtmodelle den Zugang zur breiten Öffentlichkeit. Gleichzeitig verbesserte sich die Qualität der virtuellen Stadtmodelle und damit der wirtschaftliche Nutzen. So werden virtuelle Stadtmodelle verstärkt für Krisenmanagement, Städteplanung, Verwaltungsangelegenheiten, aber auch immer stärker für militärische Zwecke genutzt.

Die Verbesserung der Techniken zum Aufbau eines virtuellen Stadtmodells spiegelte sich auch in der Computerspielewelt wieder. Im März 2007 brachte die Softwarefirma GSC Game World ein Spiel[1] auf dem Markt, welches durch ein virtuelles Stadtmodell vom AKW Tschernobyl beeindruckt. Die Softwarefirma entwickelte anhand von Satellitenaufnahmen und eigenen Fotos das virtuelle Stadtmodell vom AKW Tschernobyl.
Weiterhin fanden freie Softwaresysteme inzwischen großen Zuspruch bei Computeranwendern, welche eine Erstellung eigener virtueller Stadtmodelle in privaten Rahmen ermöglichen. Hierbei spielen nicht nur Programme wie „Google SketchUp“ eine Rolle, sondern auch weitergehende Modellierungsprogramme finden immer mehr ihren Nutzen, wie zum Beispiel „Blender“.

Mit Blender steht ein Open Source Programm zur Verfügung, welches in der Lage ist, beeindruckende Rasterbilder zu erstellen sowie realistische Modelle aufzubauen. Neben der Modellierung bietet Blender die Möglichkeit, eigene Animationen zu erstellen. Des Weiteren profitiert der Anwender von einer programminternen Game-Engine.

[1] Ego-Shooter „S.T.A.L.K.E.R.: Shadow of Chernobyl“

Aufgrund des immensen Funktionsumfanges kann Blender sich mit kommerziellen Modellierungsprogrammen wie Cinema4D ohne Weiteres messen.

Die vorliegende Diplomarbeit beschreibt die Modellierung eines Gebäudes im abgesiedelten Dorf Bonnland, auf dem Truppenübungsplatz Hammelburg, für das Amt für Geoinformationswesen der Bundeswehr[2]. Für die Modellierungsarbeiten wurde das Open Source Programm Blender verwendet. Kapitel 2 erläutert den Begriff 3D-Stadtmodell sowie die daraus resultierenden Forderungen. Des Weiteren werden theoretische Grundlagen vermittelt, wie Standards und Modellierungsarbeiten. Abschließend werden 3D-Softwarelösungen vorgestellt.

Kapitel 3 gibt einen Vergleich zwischen der kommerziellen Software Cinema4D und dem Open Source Programm Blender wieder. Weiterhin wird auf ein Projekt des Zentrums für Graphische Datenverarbeitung e. V. eingegangen. Darüber hinaus werden Alternativen benannt, um Projekte nach Modellierungsarbeiten für Interaktionen zu verwenden.

Kapitel 4 beschäftigt sich mit dem Softwarepaket Blender. Es werden Datenformate näher gebracht sowie Standardschnittstellen von Blender, welche die Programm übergreifende Verwendung von Blender-Objekten sicherstellen. Weiterhin wird das Programm ausgiebig beleuchtet, mit Vorteilen und Nachteilen. Des Weiteren erfolgt ein Überblick über die Modellierungsarbeiten zum Gebäude 23. Hierbei werden Arbeitsschritte wiedergegeben sowie auftretende Schwierigkeiten, mit entsprechenden Lösungsvorschlägen.

Kapitel 5 stellt Abbildungen des Gebäudes 23, aus dem Modell, mit Abbildungen des Gebäudes 23 in der Realität gegenüber.

Abschließend gibt Kapitel 6 das Fazit sowie den Ausblick wieder.

[2] AGeoBw

2 Dreidimensionales Stadtmodell

Das nachfolgende Kapitel befasst sich mit der Definition des 3D-Stadtmodells, beleuchtet den Hintergrund und gibt im Anschluss Forderungen für ein 3D-Stadtmodell wieder. Weiterhin werden Standards wiedergegeben, wie zum Beispiel CityGML. Es werden die populärsten Modellierungsarten dargelegt, sowie abschließend die bekanntesten Softwarelösungen, für 3D-Modellierungen.

2.1 Definition

3D-Stadtmodelle belebten in der Vergangenheit immer mehr die IT-Branche und den Wissensdrang in der Wirtschaft sowie im privaten Leben. Dennoch existiert derzeit keine eindeutige Definition von 3D-Stadtmodellen. Dem ungeachtet soll nachfolgend eine Definition aufgestellt werden.

> Grundlegend definiert sich das 3D-Stadtmodell über eine digitale Abbildung der Geländeoberfläche, den dazugehörigen Gebäuden, den Straßen, der Vegetation sowie sämtlicher Stadtmöblierungen der betreffenden Szenerie, zusätzlich sollten 3D-Stadtmodelle den Forderungen nach Interoperabilität, Generalisierung, Multimedia und Semantik erfüllen.

Für die vielfältigsten Aufgaben wie Katastrophenmanagement, Standortplanung oder Simulationen, setzt es zusätzlich nicht nur ein reines 3D-Stadtmodell aus Geometriedaten voraus, sondern ebenso deren semantischen Eigenschaften, von der Ebene ganzer Objekte wie Gebäuden bis zu den kleinsten Bestandteilen und Teilobjekten wie Balkonen und Fenstern. Weiterhin ist für die effektive Nutzung eine weitere Voraussetzung, dass 3D-Stadtmodelle eine vielfältige Verbreitung finden und problemlos zwischen verschiedenen Systemen, Formaten und Organisationen ausgetauscht werden können. [Gru05]

Nach dem zuvor genannten Absatz bilden Grundrisspläne bzw. Luftbilddaten die Grundlage für ein 3D Modell, weiterführend ist dagegen eine Erfassung von wichtigen Elementen wie Fenster, Mauervorsprünge, Mauerstärken, Türöffnungen, künstliche Lichtquellen u.a. von wesentlicher Bedeutung. Die Erfassung der zuvor genannten Elemente setzt der Modellierungsarbeit eine örtliche Begehung voraus bzw. für Planungsmodellierungen eine Festlegung der jeweiligen Elemente. Für eine effektive Weiterverwendung modellierter Objekte unter

anderen IT-Systemen bilden programminterne Speicherformate sowie Exportformate einen wichtigen Punkt. Somit ist eine vorausschauende Verwendung modellierter Objekte unumgänglich.

Für das Gebäude 23 in Bonnland ergeben sich unter Beachtung der zwei eingangs genannten Absätze folgende Forderungen:

- **Interoperabilität:** Aufgrund der Weiterverwendung des Modells in einem anderen Anwendungssystem, innerhalb der Bundeswehr, ist die Prüfung der Konvertierung unumgänglich. Dank der Modellierung mit Blender ist eine grundlegende Konvertierung in andere Formate sichergestellt. Die Standardschnittstellen sind in Tabelle 4.1 aufgelistet. Das Format CityGML, welches zukünftig das Standardformat für 3D-Stadtmodelle sein wird, geht Abschnitt 2.2.2 ein.

- **Generalisierung:** Für das 3D-Modell des Gebäudes 23 ergibt sich die Forderung nach einem Basismodell und einem Anwendungsmodell sowie auch für jedes andere 3D-Stadtmodell. Während das Basismodell aus Primitiven besteht, welche sich zu Aggregaten zusammensetzt und Materialeigenschaften annehmen, wie zum Beispiel Farbe oder eine Textur. Erfolgt im Anwendungsmodell die semantische Ausgestaltung des Objektes in mehreren Stufen. Abschnitt 2.2.2 geht hierbei noch gesondert auf das Anwendungsmodell ein.

- **Multimedia:** Zur Einweisung von militärischen Operationen nutzt die Bundeswehr bereits heute multimediale Elemente. Der gleiche Verwendungszweck ist auch für das Modell Bonnland angedacht, zur reinen Groborientierung ist hierfür eine sequentielle multimediale Darstellung absolut ausreichend. Für eine verfeinerte Geländeorientierung muss eine nichtsequentielle multimediale Darstellung im Projekt aufgebaut werden. Die nichtsequentielle multimediale Darstellung muss hierbei Fragen beantworten können wie: Himmelsrichtungen, Geländemerkmale, Lage Zielobjekt, Zufahrtswege, Eingangsbereiche, Freiflächen etc.

- **Semantik:** Für eine effektive Nutzung ist die Verarbeitung und Weitergabe von semantischen Eigenschaften im Modell erforderlich. Hierzu sollte das Modell des Gebäudes 23 zum Beispiel über Mauerwerk, Zugangswege oder auch verbaute Materialen weiterführende Informationen enthalten. Wie im Vorfeld erwähnt wird das Format CityGML durch Blender nicht unterstützt. Aufgrund dieser Tatsache erfährt die Semantik eine große Einschränkung. Um dem Fehlen des Formates CityGML entgegenzuwirken, können semantische Informationen über die Anbindung einer Datenbank mittels Python realisiert werden.

Zusammenfassend ist die Modellierung des Gebäudes 23 von vornherein mittels der Software Blender realisierbar. Die Forderung der Interoperabilität kann erfüllt werden. Damit ist eine Fortführung der Modellierungsarbeiten bzw. Weiterverwendung des Modells unter einem anderen System realisierbar. Die Einschränkung durch das Fehlen des Formates CityGML fällt im Interoperabilitätsbereich nicht gravierend aus. Eine Lösung ist zukünftig denkbar, durch Studienarbeiten der Studenten des Fachbereiches Geoinformatik der Hochschule Neubrandenburg. Die Forderung der Generalisierung ist durch das Basismodell sowie dem dazugehörigen Anwendungsmodell mit seinen 5 Abstraktionsstufen realisierbar. Grundlegend ist hierbei die Forderung des Auftraggebers. Die Multimediale Forderung ist im Bereich sequentielle multimediale Darstellung durchführbar. Der nichtsequentielle Bereich ist mittels Blender ebenso möglich, durch die interne Game-Engine, hierbei ist jedoch mit Einschränkungen im grafischen Sektor zu rechnen. Zur Verbesserung des grafischen Eindrucks ist auf externe Game-Engines zurück zugreifen, Abschnitt 3 geht gesondert nochmal auf die Game-Engine ein.
Die Forderung der Semantik ist kritischer zu betrachten. Grundlegend ersetzt ein 3D-Stadtmodell kein Geoinformationssystem. Daher können keine hohen Erwartungshaltungen in diesem Bereich gesetzt werden. Ein Geoinformationssystem bietet nicht nur einen grafischen Betrachtungsteil, sondern ermöglicht darüber hinaus auch Analyseaufgaben. Im Analysebereich liegt die Schwachstelle eines 3D-Stadtmodells, um dieser Schwachstelle in einem gewissen Grad zu begegnen, wäre eine Kombination zwischen einer interaktiven Anwendung, mit einem semantischen Anteil, basierend auf CityGML, denkbar. Diese Idee könnte mit Blender und einer eingebundenen Datenbank realisiert werden.

2.2 Standards

Standards tragen zur Vereinheitlichung von Schnittstellen und Produkten bei, was zu einer geringeren Marktsegmentierung führt. Zudem verkürzt die Bezugnahme auf Standards die Entwicklungszeit von Produkten, sodass selbst komplexe Systeme durch Standardisierung keine wesentlichen Kosten als andere Produkte haben. Erst die Schaffung von Standards für Schnittstellen zwischen Teilsystemen ermöglichen überhaupt den effizienten Aufbau von daraus bestehenden komplexen Systemen. Daher werden nachfolgend die Standards GML und CityGML beschrieben. Beide Standards sind ausschlaggebend für den effizienten Aufbau eines 3D-Stadtmodells.

2.2.1 GML

Geography Markup Language ➠ GML ist ein Austauschformat von Geobezogenen Objekten, basierend auf das XML Schema. GML erlaubt dem Nutzer im Bereich von Geodaten die Übermittlung von Attributen, Relationen und Geometrie. In Kooperation mit der ISO TC 211 und des OGC wird GML definiert und liegt derzeit in der Version 3.2.1 vor. Allgemein

ausgedrückt ist GML, eine auf Text basierende Programmiersprache und dient der Erzeugung von geographischen Objekten.

GML arbeitet auf der Grundlage von OGC Abstract Specification, die OGC hat für geografische Objekte folgende Eigenschaften definiert:

- Geografische Objekte sind Abstraktionen der realen Welt
- Geografische Objekte werden durch geometrische Werte beschrieben
- Geografische Objekte sind innerhalb eines Koordinatensystems festgelegt [Lak08]

Zusammenfassend bietet GML folgende Vorteile:

- standardisierte Beschreibung geometrischer und nichtgeometrischer Objekteigenschaften der realen Welt
- einfache Bearbeitung mit einem Text-Editor
- Übermittlung von Objekten mit Geometrien, Relationen und Attributen
- Kombination verschiedener Daten möglich
- einfache Transformation in ein anderes Datenformat
- Daten- und Darstellungsinformationen werden voneinander getrennt [Zür08]

Weiterhin weist GML eine reichhaltige Menge von Primitiven auf, die zum Aufbau anwendungsbezogener Schemata oder Anwendungssprachen dienen. Im einzelnen sind das:

- Objekt
- Geometrie
- Koordinatensystem
- Zeit
- dynamisches Objekt
- Überdeckung unter Einschluss von geographischen Abbildungen
- Maßeinheit
- Gestaltungsregeln der Kartendarstellung

Trotz der vergangenen Erfolge von GML ist GML für zukünftige 3D-Stadtmodelle nicht geeignet. Semantische Eigenschaften werden mit GML nicht ausreichend weitergegeben, hier bietet CityGML sich als Alternativlösung an, nachfolgender Abschnitt geht besonders auf CityGML ein.

2.2.2 CityGML

Die Darstellung von 3D-Daten werden vorrangig mittels der OGC Schnittstelle WMS (Web Map Service) oder auch WFS (Web Feature Service) dargestellt. Diese Vorgehensweise hat sich in der Praxis sehr bewährt. Jedoch liefert der WMS auf dem Client lediglich eine grafische Wiedergabe, der zuvor erteilten Anfrage. Dadurch können keine semantischen Eigenschaften von Objekten an den Nutzer weiter gegeben werden. Abhilfe bietet hierfür der Web 3D Service (W3DS). Mittels W3DS werden dreidimensionale Abbildungen erzeugt, wobei auch semantische Eigenschaften an dem Nutzer wiedergegeben werden. Grundlage für die Nutzung von W3DS ist die Verwendung des Datenformat CityGML.

CityGML ist das zukunftsweisende Datenmodell für die Darstellung von 3D-Stadtmodellen. CityGML definiert die Klassen und die Beziehungen zu den wichtigsten topographischen Objekten im Stadtmodell, hinsichtlich der geometrischen, topologischen, semantischen Eigenschaften sowie deren Aussehen. Die thematischen Informationen gehen über das grafische Erscheinungsbild hinaus und erlauben somit das 3D-Stadtmodell für weitergehende Aufgaben zu nutzen, wie Simulationen oder auch thematische Abfragen. [Kol08c, „What is CityGML?“] Das Format CityGML wird als offenes Datenmodell definiert, zur Speicherung und zum Austausch von 3D-Stadtmodellen, aufbauend auf dem XML-Format. Grundlage für CityGML ist das Datenformat GML.

Wie bereits erwähnt, lassen sich mittels CityGML ganze zusammenhängende Erscheinungsformen von räumlichen und semantischen Komponenten in einem 3D-Modell realisieren. So ist es möglich ein ganzes Gebäude in seinen einzelnen Elementen zu zerlegen (Nr. 1; Abbildung 2.1), dabei können Gebäudeflächen ab LOD 2 thematisch aufgeteilt werden. Die Aufteilung erfolgt entsprechend ihren Eigenschaften, wie zum Beispiel Dachflächen, Wandflächen, Decken-/Fußbodenflächen (Nr. 2; Abbildung 2.1). Weiterhin besteht die Möglichkeit ab LOD 2 Gebäudemerkmale bzw. Gebäudefunktionen festzulegen, wie zum Beispiel Balkon, Gauben und Treppen (Nr. 3; Abbildung 2.1). Ab der Abstraktionsstufe LOD 3 existiert die Möglichkeit Öffnungen, Türbereiche und Fensteröffnungen mitzuerfassen (Nr. 4; Abbildung 2.1). Im Bereich der höchsten Abstraktionsstufe können Details wie Räume und Inventar erfasst werden (Nr. 5; Abbildung 2.1). Zusätzlich können zu den vorweggenannten Elementen Attribute definiert werden, wie zum Beispiel Name, Funktion, Adresse (Nr. 6; Abbildung 2.1), Nutzungsart, Baujahr, Geschosshöhen, Geschossanzahl. Natürlich werden Charakteristika zu den einzelnen Oberflächen ebenfalls gespeichert, wie zum Beispiel Texturen und Farben. [Kol08a]

Zur Sicherstellung der zuvor genannten Informationen, in den einzelnen Abstraktionsstufen, kann jedes Objekt bzw. Objektteil auf Informationen in extern gelegenen Quellen zurückgreifen, wie zum Beispiel Datenbanken des Katasterwesens. Durch das Einbinden

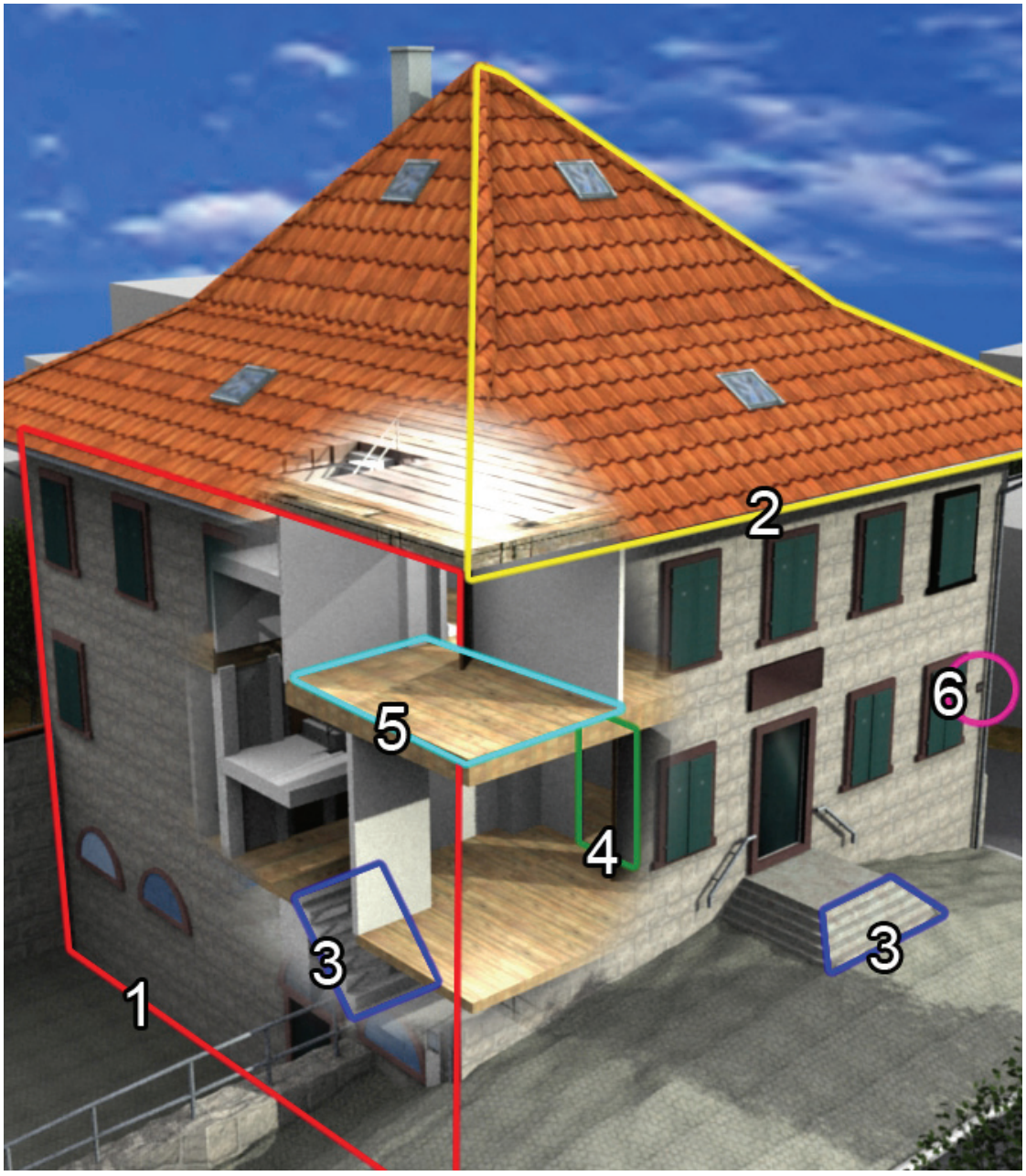

Abb. 2.1: Übersicht einiger Elemente des Gebäude 23 (Nr. 1 Gebäudeelement Außenwand; Nr. 2 Gebäudefläche Dachfläche; Nr. 3 Gebäudemerkmal Treppen; Nr. 4 Gebäudeöffnung Tür; Nr. 5 Gebäudedetail ab LoD 4 Raum; Nr. 6 Adressenattribut Hausnummer)

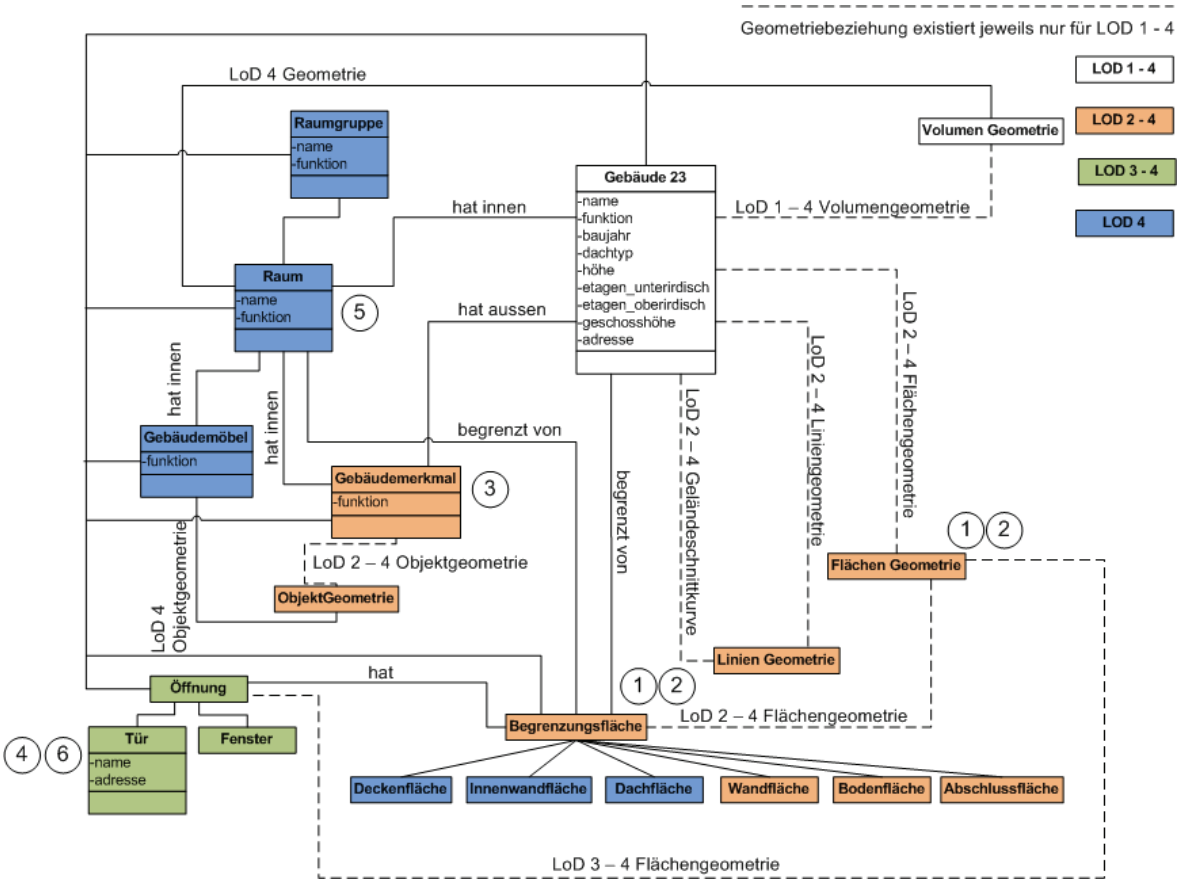

Abb. 2.2: Vereinfachtes UML-Diagramm des Gebäudes 23 in LoD 1 - 4

von Datenbanken ist es möglich 3D-Modelle für die verschiedenartigsten Verwendungen zu nutzen, wie zum Beispiel in der Disziplin Facility Management. Abbildung 2.2 gibt in einem vereinfachten UML-Diagramm die Darstellung des Gebäudes 23 in den verschiedenen Abstraktionsstufen wieder, die angegebenen Zahlen spiegeln die Elemente aus Abbildung 2.1 wieder.

Abstraktionsmöglichkeiten ➠ Level of Detail

Bei der Erstellung eines 3D-Stadtmodell kann zwischen mehrere Detailstufen unterschieden werden. Ausschlaggebend hierfür sind die Erfassungsmethoden, der Grundlagendaten sowie die erforderliche Genauigkeit und Auflösung. Für eine effiziente Visualisierung werden für betrachternahe Standorte hochaufgelöste Darstellungen verwendet. Für entferntere und im Bildhintergrund befindliche Objekte gröbere Darstellungen. Weiterhin sollen sich 3D-Stadtmodelle erst in besonderen Anwendungsfällen in der geforderten Detailstufe präsentieren.
Daher wird das 3D-Modell in Basis- und Anwendungsmodell unterschieden. Das Basismodell stellt die universelle Grundlage für verschiedene thematische Modellierungen dar und ist in verschiedenen Zusammenhänge wiederverwendbar. Das Anwendungsmodell bildet die thematischen Eigenschaften sowie Strukturen der für eine Anwendung relevanten 3D-Geoobjektes ab. Je nach Verwendungszweck werden verschiedene Detailstufen (Level of Detail) für ein 3D-Modell benötigt. Wie bereits im vorherigen Abschnitt 2.2.2 erwähnt, wird das Anwendermodell in verschiedene Abstraktionsstufen „Level of Detail“ (LoD) unterschieden und gliedert sich in 5 verschieden Detailstufen. Eingesetzt wird dieses Verfahren für eine effizientere Visualisierung, wie bereits zu Beginn des Absatzes erwähnt.

Die nachfolgenden kursiv angegebenen Genauigkeiten sind Richtwerte der SIG 3D der GDI NRW [Kra04, S.3] [Plü04].

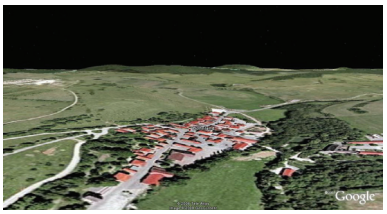

Abb. 2.3: Level of Detail 0
Quelle: Google Earth

LOD 0 - Regionalmodell
Level of Detail 0 definiert das Regionalmodell, welches aus einem dreidimensionalen digitalen Geländemodell besteht sowie zusätzlich mit einem Luftbild texturiert wird. Gebäude werden im Modell nicht dargestellt.
Punktgenauigkeit (Lage/Höhe): >5m/>5m

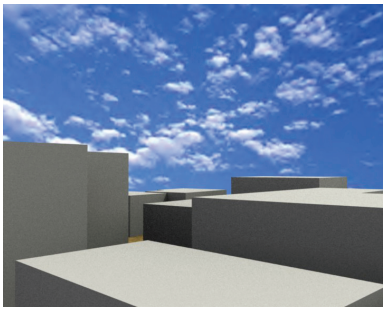

Abb. 2.4: Level of Detail 1

LOD 1 - Klötzchenmodell

In Level of Detail 1 präsentiert sich ein Gebäude als simpler Körper, ohne Dachform. Das Modell in LOD 1 wird nicht weiter verfeinert und stellt aus Sicht der Semantik lediglich eine Einheit dar. Zusätzlich können in LOD 1 mehrere Gebäude zu einem Gebäudekomplex zusammengefasst werden. Darüber hinaus können Modelle ab LOD 1 zu Visualisierungszwecke mit Materialeigenschaften ausgestattet sein.

Punktgenauigkeit (Lage/Höhe): 5m/5m

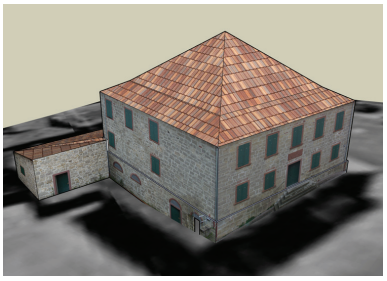

Abb. 2.5: Level of Detail 2

LOD 2 - 3D-Modell der Außenhülle, mit Dachstrukturen und einfachen Texturen

Level of Detail 2 stellt eine Entfaltung des LOD 1 Modells dar, es werden einfache Dachstrukturen und Gebäudeaußenseiten dargestellt. Im Modell wird zwischen Bestandteilen des im Fokus stehenden Modells differenziert, wie zum Beispiel Geländefläche, Wandfläche und Dachfläche.

Punktgenauigkeit (Lage/Höhe): 2m/1m

Abb. 2.6: Level of Detail 3

LOD 3 - Architekturmodell - 3D-Modell der Außenhülle mit Textur

Im Vergleich zu LOD 2 werden in Level of Detail 3 Wand- und Dachstrukturen detaillierter dargestellt. Weiterhin wird das Modell durch Vegetation und Verkehrsflächen vervollständigt, womit der Grad der realistischen Visuallisierung angehoben wird.

Punktgenauigkeit (Lage/Höhe): 0,5m/0,5m

LOD 4 - Innenraummodell - 3D-Modell des Gebäudes mit Etagen, Innenräumen, etc. und Texturen
In Level of Detail 4 wird das gesamte Modell noch differenzierter dargestellt und um seine Innenräume ergänzt. Innenräume werden texturiert und durch vorhandene Möblierung ergänzt.
Punktgenauigkeit (Lage/Höhe): 0,2m/0,2m

Abb. 2.7: Level of Detail 4

2.3 Modellierungsarten

Neben Standards gibt es zusätzlich verschiedene Modellierungsarten. Auf den kommenden Seiten werden verschiedene Modellierungstechniken vorgestellt, wobei das Volumenmodell zum Standard avanciert.

2.3.1 Drahtgittermodell

Das Drahtgittermodel, auch Wire-Frame-Modell genannt, ist ein geometrisches 3D-Modell, welches sich lediglich durch Kanten darstellt. Gleichzeitig wird auch eine Darstellungsform in der 3D-Modellierung Wire-Frame genannt. Weiterhin kann eine Darstellungsform Wire-Frame genannt werden, obwohl das Modell in einer anderen Modellierungsart aufgebaut wurde. Bei der Modellierung in Wire-Frame wird das Modell durch mit Kanten verbundene Knotenpunkte präsentiert. Vorteil des Wire-Frame ist die Sichtbarkeit von verdeckten Kanten bzw. Objektabschnitte. Des Weiteren erfordert die Berechnung der Wire-Frame Darstellung eine kurze Berechnungszeit.
Demzufolge ist die Darstellung eines komplexen 3D-Modells eine komfortable Möglichkeit der schnellen Übersichtsdarstellung. Nachteil der Wire-Frame Darstellung ist eine untreue Abbildung des Modellobjektes.

2.3.2 Flächenmodell

Das Flächenmodell ist eine Modellierungsart in der 3D-Modellierung, dabei dient ein vorhandenes Drahtgittermodell als Grundlage für das Flächenmodell. Alle Polygone des Drahtgittermodells werden im Flächenmodell einer bestimmten Farbe zugeordnet. In einem Flächenmodell wird die Flächendarstellung lediglich durch die Körperoberfläche präsentiert. Demzufolge sind dahinterliegende bzw. verdeckte Flächen nicht sichtbar.
Vorteil des Flächenmodells ist eine realistischere Darstellung, gegenüber dem Drahtgittermodell. Nachteil des Flächenmodells liegt in der längeren Berechnungszeit, im Vergleich mit dem

Drahtgittermodell. Des Weiteren präsentieren sich Farben im Flächenmodell ohne Farbübergänge. Zugehörige Schattierungseffekte können lediglich nur mit einem erhöhten Aufwand in der Bearbeitung realisiert werden.

2.3.3 Volumenmodell

Das Volumenmodell ist eine auf das Flächenmodell aufbauende Modellierungsart von 3D-Modellen, hierbei werden verschiedene Volumenmodelle unterschieden:

- Boundary Representation Modell (BRep)
- Constructive Solids Geometry Modell (CSG)
- Swept Area Solid Modell

Beim Volumenmodell werden beim Rendering vorhandene Objekteigenschaften des eigentlichen Modells berücksichtigt. Das bedeutet, während des Renderingsprozesses werden Strukturen, Oberflächeneigenschaften (z. Bsp. Textur, Reflexionsgrad), Lichtquellen, Schattenwirkung u.v.m. im Renderingergebnis einzeln für jedes Pixel extra berechnet. Farbübergänge und Polygonübergänge werden kontinuierlich berechnet.
Vorteil des Volumenmodells ist die realistischere Darstellung. Nachteil des Volumenmodells ist die erhöhte Berechnungszeit, im Vergleich mit dem Drahtgitter- und Flächenmodell.

Nachfolgend werden die Boundary Representation und Constructive Soilds Geometry Methoden erläutert. Auf die Erklärung von Swept Area Solid wird verzichtet, da diese verstärkt ihre Anwendung in CAD Programmen wiederfindet.

2.3.3.1 Boundary Representation (BRep)

➠ ist eine Volumenmodellbeschreibungsmethode und indirekte Modellierung. Hierbei werden Objekte durch ihre eigene Oberfläche dargestellt. Die Oberflächenbeschreibung eines Objektes setzt sich aus den Primitiven

- Knoten (vertex)
- Kante (edge)
- Fläche (face)

zusammen. Dabei besitzt jedes dieser Elemente eigene Attribute, welche das Primitiv geometrisch beschreiben (Koordinaten, Krümmung, Steigung) sowie die Beziehung zu angrenzenden Primitiven. Das BRep Modell unterscheidet sich vom Flächenmodell dadurch, dass es in der Datenbasis Materialvektoren zur Erfassung der Volumeninformation einfügt. BRep Modelle finden ihre häufigste Anwendung nicht nur in der 3D Computergrafik sondern auch in CAD Programmen, weil sie aufgrund ihres Algorithmus schnell zu berechnen sind.

2.3.3.2 Constructive Solid Geometry (CSG)

➠ verfolgt im Vergleich zu BRep ein anderes Verfahren. Bei CSG kommen boolesche Operatoren zum tragen. Mittels der booleschen Operatoren werden einfache geometrische Formen (Würfel, Kugel, Zylinder, Ring, Kegel etc.) miteinander zu einem komplexen Objekt kombiniert. Dafür stehen folgende Operatoren zur Verfügung:

- Union (Addition) $\cup$
- Difference (Differenz) $-$
- Intersect (Verschneidung) $\cap$

Voraussetzung der booleschen Operationen sind geschlossene Körper. Aufgrund dessen, dass boolesche Operationen im Allgemeinen nicht kommutativ sind, können diese hierarchisch geordnet werden. Diese Ordnung wird als CSG Baum bezeichnet. Die Blätter geben dabei die einzelnen geometrischen Primitiven wieder und die Knoten geben die boolesche Operation wieder. Das Endergebnis des Baumes ist seine Wurzel.

2.4 Bekannte 3D Softwarelösungen

Wie bereits in der Einleitung erwähnt, gibt es eine Vielzahl von Programmen, welche sich für die 3D-Modellierung eignen. Ob jedoch jedes 3D-Modellierungsprogramm zur Erstellung von 3D-Stadtmodellen bzw. ganze virtuelle Welten prädestiniert ist, sei dahin gestellt. Voraussetzungen sind neben dem eigentlichen Programm, dem ganzem technischen Zubehör, vor allem geistige Kreativität.

2.4.1 Google SketchUp

Seitdem Aufstieg von Google Earth und der damit verbundenen Möglichkeit, Stadtszenen dreidimensional betrachten zu können, steht Google SketchUp immer öfter im Mittelpunkt der 3D-Modellierung. Mit Google SketchUp kann jeder Computeranwender eigene 3D-Modelle schnell und ohne Kostenaufwand selbst erstellen. Google SketchUp ist so aufgebaut, dass die Lernphase und Anwendung intuitiv erfolgt. Die Modellierungsmöglichkeiten reichen von Entwurfsmodellen über Architekturdarstellungen bis hin zu genauen technischen Zeichnungen. [Inc08] SketchUp wurde 1999 von @Last Software entwickelt. Dabei entwickelte sich das Motto „3D für alle“. Im März 2006 übernahm Google @Last Software, dabei wurde auch das Motto „3D für alle“ übernommen und stellt seitdem für Google den Leitspruch für SketchUp dar. Kurz nach der Übernahme stellte Google die erste kostenlose Version von SketchUp zur Verfügung, für die nicht kommerzielle Verwendung. Parallel zur kostenlosen Version von Google SketchUp existiert auch eine kostenpflichtige Version, Google SketchUp Pro. Die kostenpflichtige Version unterscheidet sich hauptsächlich in den Schnittstellen und kostet

ca. 500 Euro. Zusammenfassend kann Google SketchUp als ein Muss für jeden Neuling in der 3D-Modellierung angesehen werden. Die Arbeit verfolgt dabei folgende Prinzipien: ➟ Zeichnen, wie man es möchte ➟ Spaß ➟ Einfachheit ➟ Ausprobieren [Inc08]

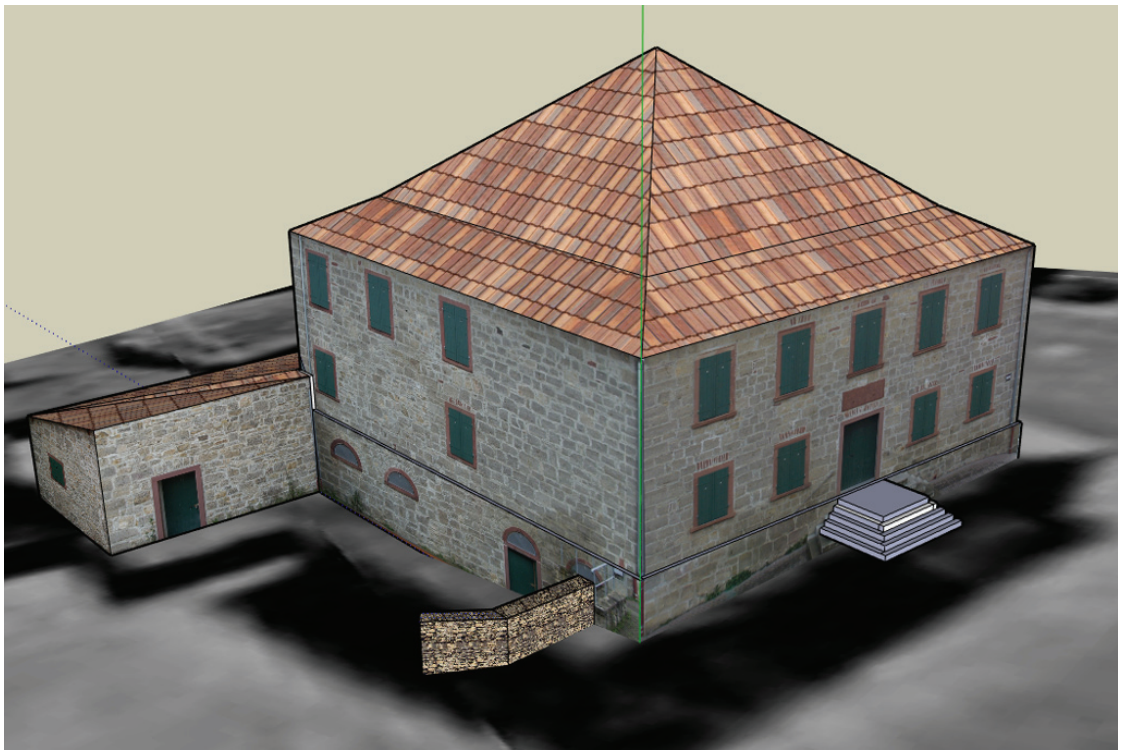

Abb. 2.8: Gebäude 23 in Google SketchUp in LoD 2

Die Anforderungen, welche für das Gebäude 23 erwartet werden, können mit Google SketchUp nicht erfüllt werden. Insbesondere die semantische Forderung wird nicht realisierbar sein. Weiterhin ist die Anbindung einer Datenbank nicht gegeben. Darüber hinaus unterstützt Google SketchUp derzeit nicht die Schnittstellen GML sowie CityGML. Aufgrund der schnellen Modellierung eignet sich Google SketchUp zur schnellen Situationsdarstellung in LoD2. Zur detaillierten Darstellung eines 3D-Objektes eignet sich Google SketchUp nicht. Abbildung 2.8 zeigt das Gebäude 23 in LoD2, konstruiert mit Google SketchUp.

Unter http://de.sketchup.com/ (Juli 2008) kann das Programm in der Free sowie in der Pro Version bezogen werden, darüber hinaus enthält die Webseite weitergehende Informationen zu Google SketchUp.

2.4.2 Cinema 4D

Cinema 4D ist ein Produkt des Softwarehauses MAXON. 1993 entwickelte MAXON die erste Version von Cinema 4D. Derzeit liegt Cinema 4D in der Version 10.5 vor. Grundlegend enthält Cinema 4D alle Funktionen um photorealistische Bilder und Animationen zu erstellen. Die

Ergebnisse können in alle gängigen Formate exportiert werden. Vorteil von Cinema 4D ist der modulare Aufbau, somit kann der Nutzer selbst entscheiden um welche Funktionen er das Basisprogramm erweitern will und dadurch Kosten sparen. Das Basisprogramm ist bereits ab 775 Euro unter http://www.cycot.de/cinema4d (Juli 2008) erhältlich. Zurückblickend kann Cinema 4D auf eine erfolgreiche Anwendergeschichte zurückblicken. So wurden zum Beispiel Filme wie „Der Polar Express“, „Krieg der Welten“ oder auch „Jagdfieber“ mit Hilfe von Cinema 4D realisiert. Weiterhin nutzen Werbefirmen Cinema 4D, wie zum Beispiel für die Firmen „Warsteiner“ und „Coca Cola“.
Zusammenfassend ist Cinema 4D ein Programm welches sein Geld wert ist. Als enormen Pluspunkt bei Cinema4D ist die Architecture Edition[1], zu der Basisversion von Cinema4D. Aufgrund der enormen Kosten ist jedoch Cinema 4D nicht für den alltäglichen Hausgebrauch zu empfehlen. Weiterhin steht der Anwender einer großen Lernphase gegenüber.

Weitere Informationen zu Cinema 4D sind unter http://www.maxon.net (Juli 2008) erhältlich.

2.4.3 3ds Max

3ds Max ist ähnlich wie Cinema 4D ein gestalterisches 3D Computergrafik- und Animationsprogramm, welches Anwendung in die Erstellung von photorealistischen Bildern, Animation, Design und Architektur findet. 3ds Max entwickelte sich aus 3d Studio Max und wurde vormals durch verschiedene Softwarehäuser hervorgebracht, wie Kinetix und Discreet. Seit der Version 8 wird die Software durch Autodesk entwickelt und ist seitdem unter den Namen 3ds Max bekannt. Ähnlich wie bei Cinema 4D kann auch 3ds Max auf eine erfolgreiche Anwendergeschichte zurückblicken. Unter anderem entstanden Filme wie „Mr. & Mrs. Smith“, „Die Truman Show“ oder auch „Black Hawk Down“ mit 3ds Max. Des Weiteren wird die Software auch von der Spieleindustrie intensiv genutzt, so entstanden die Spiele „Medal of Honor“ oder auch „Die Siedler: Das Erbe der Könige“ unter der Verwendung von 3ds Max.
Analog wie bei Cinema 4D geht 3ds Max mit einem hohen Anschaffungspreis einher. Eine Einzelplatzlizenz der Version 3ds Max 2009 kostet 4.641 Euro, zu beziehen unter http://www.lichtblick4d.com/shop (Juli 2008).

Weitergehende Informationen zu 3ds Max gibt es unter http://www.autodesk.de (Juli 2008).

2.4.4 LandXplorer

Im Vergleich zu Cinema 4D oder den anderen vorhergehenden Programmen ist LandXplorer eine Anwendung, welche speziell für das Management und der Visualisierung geovirtueller 3D-Stadtmodelle und 3D-Landschaftsmodelle erschaffen wurde. LandXplorer bietet zum Aufbau

[1] http://www.maxon.net/pages/products/editions/architecture/architecture_d.html (August 2008)

der eben genannten Modelle eine umfangreiche Import- und Konvertierungsfunktionalität, die unterschiedliche Typen und Quellen von 2D- und 3D-Geodaten nahtlos integriert. Zusätzlich unterstützt LandXplorer die Kommunikation komplexer 3D-Geoinformationen und stellt eine umfangreiche Funktionalität zum Export des Geo-Content auf unterschiedliche Medien und Formate bereit. [Gmb08a] LandXplorer präsentiert sich derzeit in zwei Versionen, zum einem als LandXplorer Studio, zum anderem als LandXplorer Studio Professional. LandXplorer Studio beinhaltet Basisfunktionen zur 3D-Visualisierung von Geodaten. LandXplorer Studio Professionell bietet dagegen ein erweitertes Funktionsfeld. Die LandXplorer Studio Version kostet zur Zeit 1.950 Euro. Der Preis der professionellen Version ist auf Anfrage bei 3D Geo GmbH zu erfahren.
LandXplorer wird durch die Firma 3D Geo GmbH aus Potsdam vertrieben. 3D Geo GmbH versteht sich als Pionier im Bereich der Geovisualisierung, so entwickelte 3D Geo GmbH unter anderem das 3D Modell zu Berlin sowie von Dresden, beide Modelle können mittels Google Earth betrachtet werden.

Weitergehende Informationen zu LandXplorer gibt es unter http://www.3dgeo.de (Juli 2008).

2.4.5 tridicon™ Produkte

Unter tridicon™ Produkte versteht sich eine ganze Produktpalette von Softwarelösungen, zum Auswerten, Planen, Entwickeln und Präsentieren von urbanen Gebieten. Die einzelnen Softwarepakete sind Bestandteil eines Gesamtkonzeptes zur Generierung, Nutzung und Pflege von 3D Stadtmodellen. Im einzelnen befinden sich folgende Produkte derzeit im Vertrieb [Gmb08b]:

- tridicon™ 2D (digitale 2D Kartierung)
- tridicon™ 3D (digitale 3D Modellierung)
- tridicon™ SCRIPT (automatische Verarbeitung von 3D-Geodaten)
- tridicon™ CAPTURE (Gewinnung orientierter Digitalfotos)
- tridicon™ TEXTURE (photorealistische Texturierung von digitalen 3D Modellen)
- tridicon™ CityDiscoverer (Analyse und Visualisierung von digitalen 3D Stadtmodellen)
- tridicon™ ARCHITECTURE (Aufmaß und 3D Modellierung aus Digitalphotos)
- tridicon™ MANAGE (Verwaltung von digitalen 3D Stadtmodellen)
- tridicon™ GIPSY (Gesamtsystem zur photorealistischen Texturierung)

Die tridicon™ Produkte werden durch die Firma GTA Geoinformatik GmbH aus Neubrandenburg realisiert. Die zuvor genannten Softwarelösungen sind lediglich über die Firma GTA Geoinformatik GmbH zu erwerben.

Weitergehende Informationen zu tridicon™ Produkte gibt es unter http://www.gta-geo.de (Juli 2008).

2.4.6 Blender

Mit Blender steht ein 3D Modellierungsprogramm zur Verfügung, welches die Vorzüge eines Open Source Programmes bietet sowie die Professionalität kommerzielle 3D Modellierungsprogramme. Blender wurde vom niederländischen Animationshaus NeoGeo entwickelt. Seit dem 13. Oktober 2002 steht Blender als GNU General Puplic License der Welt offen. Durch die offene Lizenz wurde Blender seitdem durch eine Weltweite Entwicklergemeinschaft immer weiter entwickelt. Derzeit steht die Version 2.47 von Blender zur Verfügung, zu beziehen unter http://www.blender.org/download/ (Juli 2008). Blender steht für eine Vielzahl der gängigsten Betriebssysteme zur Verfügung, darunter Linux, Apples Mac OS X und Windows.

Abb. 2.9: Blender Logo

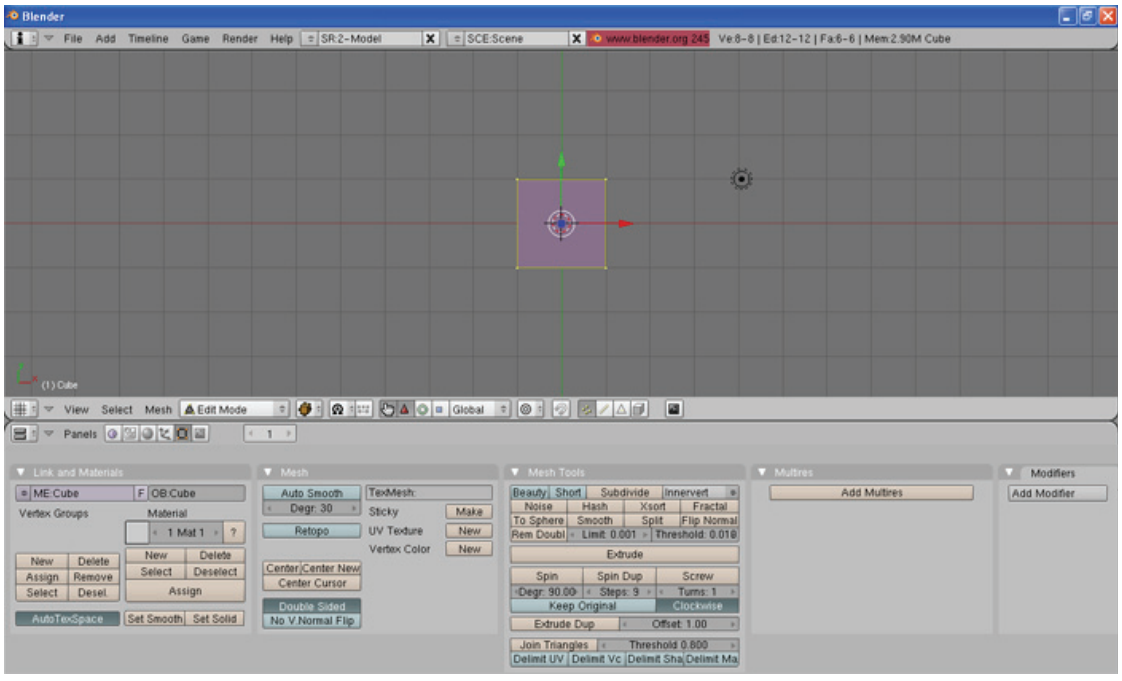

Abb. 2.10: Blenderoberfläche im Editormodus

Ergebnisse der Game-Engine von Blender erreichen sicherlich nicht die Qualität, wie sie aus Spielen wie „Call of Duty“ bekannt sind. Jedoch bei der Erstellung einer Szenerie, welche

zum Rasterbild verarbeitet werden soll, kann Blender sich mit der kommerziellen Konkurrenz messen. Blender schafft mit seinen internen Funktionen einen Standard, welcher für Grafiker sehr interessant ist. So lassen sich photorealistische Bilder und Animationen sowie 3D-Welten erschaffen. Mit Blender kann jeder problemlos und kostengünstig den Einstieg in die Welt der 3D-Animation und 3D-Programmierung finden.

Dank eines raffinierten Aufbaus des originalen Blenderformat .blend, ist Blender in der Lage eigene Dateien schnell und zuverlässig zu speichern und zu laden. Weiterhin sind die Blender eigene Dateien auf- und abwärts kompatibel, somit ist gesichert, dass Dateien von aktuellen Blender-Versionen auch in älteren Versionen gelesen und bearbeitet werden können. Die Auf- und Abwärtskompatibilität ist eines der Hauptziele, welche sich die Entwickler von Blender gesteckt haben. [War07, S. 6]

Fazit: Leider ist Blender sehr eigen, der Anwender sieht sich einer nicht ganz einfachen Bedienung und einem riesigen Funktionsumfang konfrontiert. Wer sich aber lange genug mit Blender beschäftigt, wird dennoch zu ausdrucksvollen Ergebnissen kommen.

3 Game-Engine Blender

Im Zeitalter der althergebrachten Computeranimation wurden die einzelnen Bilder aufwändig berechnet. Die zuvor gerenderten Bilder wurden dann vorgehalten und an entscheidender Situation im Spiel bzw. Programm eingebracht. Mit einer Game-Engine dagegen sind sogenannte Echtzeitgrafiken möglich, dass heißt innerhalb weniger Momente muss das Programm mindestens 15 Bilder pro Sekunde erstellen, um dem Anwender eine flüssige Bewegung auszugeben. Damit liegt klar auf der Hand welche enorme Leistung eine Game-Engine zu leisten hat, um grafische Änderungen im Programm sofort anzuzeigen. Diese Vorgehensart ist jedoch alleine nicht durch eine komplexe Game-Engine realisierbar, sondern zusätzlich durch eine ausgereifte Technik in der Grafikkarte. Die Grafikkarte übernimmt an entscheidender Stelle die Berechnung des einzelnen Bildes. Damit ist auch klar, dass der Prozessor auf der Grafikkarte in der Echtzeitdarstellung einen immer wichtigeren Platz einnimmt.

Tabelle 3.1 gibt einen Überblick über die derzeit am stärksten vertretenen Gamen-Engines wieder, zu einem im kommerziellen Bereich sowie im Open Source Bereich.

Mit der Version 2.0 von Blender wurde erstmals eine Game-Engine in Blender eingesetzt. Die Game-Engine bildet hierbei das eigentliche Grundgerüst für ein Computerspiel, für Echtzeitausgaben oder einfach nur für Architekturvisualisierungen. Grundlegend besteht eine Game-Engine aus folgenden Komponenten:

- Grafik-Engine
- Physik-Engine
- Soundsystem
- Künstliche Intelligenz
- Zustandsspeicherung
- Steuerung

- Netzwerkcode
- Datenmanagement
- Scripting

Inwieweit jede einzelne Komponente zum Einsatz kommt hängt vom Entwickler sowie von der Verwendung des Programmes bzw. des Modells ab.

Name der kommerziellen Game-Engines & Entwickler	**Name der Open Source Game-Engines & Entwickler**
➟Torque Game Engine „Garage Games“	➟OGRE “Steve Streeting“
➟TV3D SDK 6.5 „Truevision3D, LLC“	➟Irrlicht “Nikolaus Gebhardt“
➟3DGameStudio „Conitec Datasystems“	➟Crystal Space “Jorrit Tyberghein“
➟C4 Engine „Terathon Software“	➟Panda3D “The Panda Development Group“
➟OGRE „Steve Streeting“	➟jME “Mark Powell, Joshua Slack“
➟NeoAxis Engine „NeoAxis Group“	➟Reality Factory “Gekido Design Group“
➟DX Studio „Worldweaver Ltd.“	➟Blender Game Engine “Blender Foundation“
➟Unity „Unity Technologies ApS“	➟The Nebula Device 2 “Radon Labs“
➟Leadwerks Engine 2 „Leadwerks Software“	➟RealmForge “Dan Moorehead“
➟Visual3D.NET „Realmware Corporation“	➟OpenSceneGraph “Robert Osfield, Don Burns“

Tabelle 3.1: Überblick von Game-Engines im kommerziellen Bereich sowie im Open Source Bereich

3.1 Vor- und Nachteile der Game-Engine in Blender

Vorteil der integrierten Game-Engine von Blender ist mehr als deutlich. Blender verinnerlicht nicht nur den Modellierungspart, sondern gibt dem Anwender eine mehr oder weniger komplette Game-Engine an die Hand.

Aufgrund dessen, dass die Game-Engine in Blender integriert ist, müssen gefertigt Modelle nicht aufwendig konvertiert werden und der Game-Engine zugänglich gemacht werden. Mittels der grafischen Oberfläche des Logic Bereiches (F4 in Blender) sind Interaktionen und Animationen schnell realisiert. Möglich ist diese Einfachheit des Logic Bereiches durch die „click and pull“ Funktion der Steuerungskomponenten. Darüber hinaus können komplexe Interaktionsvorgänge durch Python Scripte verwirklicht werden.

Nachteilig während der Bearbeitung mit der Game-Engine in Blender ist der sehr komplexe Funktionsumfang. Jeder einzelner Komponententeil der Game-Engine versteckt sich in separaten Funktionsbereichen von Blender. Wie bereits zuvor erwähnt, die Steuerungskomponente im Logic Bereich, die Einstellungsmöglichkeiten der Physik-Engine im Object Bereich, die Einstellungsmöglichkeiten der Grafik-Engine im Shading Bereich, sowie in allen darunter befindlichen Menüs. Weiterhin nachteilig ist die Organisation der Texturen, wenn die Game-Engine zum Einsatz kommen soll. Hierbei muss für jede Fläche ein Texturblock definiert werden. Weiterhin ist eine Weitergabe der Ergebnisse der Game-Engine auf andere Rechner nur dann möglich, wenn das komplette Modell im Vorfeld gepackt wurde.

Der gravierendste Nachteil der Blender Game-Engine ist das grafische Ergebnis (Abbildung 4.21). Die Game-Engine von Blender ist nicht zu vergleichen mit professionellen Game-Engines kommerzieller Software Firmen. Anwender von Blender müssen mehr oder weniger mit einem eingeschränktem Ergebnis der Game-Engine rechnen. Die Materialeffekte sind im Ergebnis der Game-Engine nicht überzeugend, Schatten- und Lichteffekte werden nicht realisiert. Momentan laufen jedoch Entwicklungsbemühungen die Game-Engine von Blender zu verbessern, insbesondere in der Materialdarstellung sowie der Schatteneffekte. Für überzeugendere Ergebnisse ist der Nutzer gezwungen auf alternative Game-Engines zurückzugreifen. Tabelle 3.1 gibt Aufschluss über vorhandene Game-Engines, ob Kommerziell oder auch Open-Source.

3.2 Cinema4D vs. Blender

Nachfolgend soll die Software Cinema4D mit der Software Blender verglichen werden. Cinema4D steht hierbei als ein Vertreter der 3D-Grafiksoftware und Blender als Vertreter einer Game-Engine. Grundlegend besteht der Unterschied darin, dass eine 3d-Grafiksoftware lediglich einzelne gerenderte Bilder bzw. Animationen ausgibt, während eine Game-Engine Echtzeitvisualisierungen ermöglicht und damit verbundene Interaktionen. Im Vergleich flossen Erfahrungen des Zentrums für Graphische Datenverarbeitung e. V.[1] aus Rostock mit ein. Das ZGDV erstellte mit Cinema4D ein virtuelles Modell und Animationsfilm der Hansestadt Rostock[2].

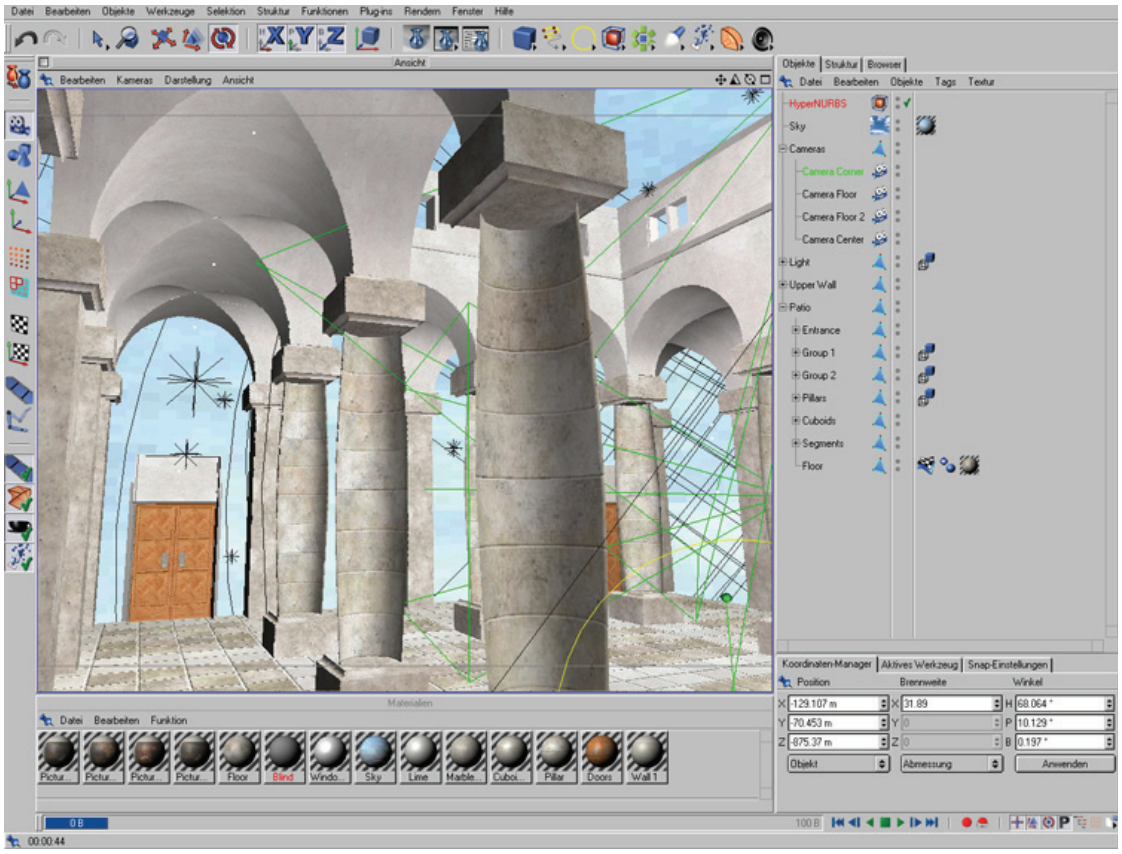

Abb. 3.1: Oberfläche Cinema4D v7

Aufgrund einer intuitiven Oberfläche präsentiert sich Cinema4D beim Anwender freundlicher (Abbildung 3.1), als Blender. Trotz allem präsentiert sich Cinema4D mit einem gleichen Funktionsumfang wie Blender. Erst nach der Modellierungsarbeit werden Unterschiede spürbar. Während Blender mit der integrierten Game-Engine „auftrumpfen“ kann, bietet Cinema4D dem Anwender „lediglich“ Animationstools an, Interaktionen werden nicht realisiert.

[1] ZGDV

[2] http://www.idw-online.de/pages/de/news272208 (Juli 2008)

So nutzt ZGDV zur Echtzeitvisualisierung von Projekten verschiedene externe Systeme wie OpenSG[3], Delta3D[4], bis hin zu komplexen Virtual Reality Systemen wie Instant Reality[5] oder auch IC:IDO[6]. Zu Animationszwecken greift das ZGDV auf Cinema4D Renderserver zurück.

Für anfallende Animationsarbeiten nach der Modellierung erweist sich Cinema4D genauso komplex wie Blender. Der Funktionsumfang in Cinema4D zu Animationszwecken enthält ein ähnlich umfangreiches Angebot wie Blender. Tabelle 3.2 gibt einen Überblick über die Animationstools in Cinema4D.

Tool in Cinema4D	**Toolbeschreibung**
Zeitleiste	Erzeugung, Kontrolle und Wiedergabe von Animationen
Animationstypen	Keyframe-, Parameter-, Deformations-, Boole-, Freiformdeformations-, Point-Level-Animation
Bones und Inverse Kinematik	Animation von organischen Objekten
Raum- und Zeitkontrolle	Kontrollinstrument der Zeitleiste, für räumliche und zeitliche Veränderungen
Motion-Groups	Gruppierung und Überblendung von Sequenzen
Partikelsystem	Rauch-, Feuer-, Flüssigkeits-, Schwarmeffekte
Expressions	Erweiterungen, Plug-Ins, Schnittstellen, Programmierbereich

Tabelle 3.2: Animationswerkzeuge in Cinema4D

Schlussfolgernd beeindrucken Rasterbilder und Animationsfilme aus Cinema4D ebenso wie Bilder aus Blender. Jedoch sind auch Modelle in Cinema4D, welche für Echtzeitvisualisierungen vorgesehen sind, nicht vergleichbar mit der Qualität gerenderter Videos bzw. Rasterbilder. Nachteilig ist die fehlende integrierte Game-Engine in Cinema4D, aber bei einem ausreichendem Budget stehen genügend externe Lösungen zur Verfügung. Darüber hinaus fordert Cinema4D einen gleichen Zeitanspruch beim Rendern wie Blender.

Wie bereits erwähnt, erstellte das ZGDV das Projekt „Hansestadt Rostock 3D“, dieses Projekt wurde mit der Software Cinema4D erstellt. Das ZGDV benötigte hierfür einen Zeitraum von 3 Mannjahre, das Projekt enthält eine Fläche von ca. 200 km^2 wieder. Die Grundlage für das Projekt „Hansestadt Rostock 3D“ war ein CAD-Plan. Dieser CAD-Plan wurde in Cinema4D eingelesen, Gebäude, von denen die Höhe bekannt war, wurden sofort umgesetzt.

[3]http://www.opensg.vrsource.org (August 2008)
[4]http://www.delta3d.org (August 2008)
[5]http://www.instantreality.org (August 2008)
[6]http://www.icido.de (August 2008)

Bei Gebäuden mit unbekannter Höhe wurde diese durch Mitarbeiter vor Ort durch ein vereinfachtes Verfahren bestimmt und im Projekt realisiert. Im Anschluss wurden die einzelnen Texturen eingearbeitet. Zum Rendern des Projektes griff das ZGDV auf eine Renderfarm mit 20 Renderclienten zurück. Hierbei wurden für 90 Sekunden Animation ca. 7 Tage benötigt. Abbildung 3.2 gibt kurz das Ergebnis als Screenshot wieder.

Abb. 3.2: Projekt des ZGDV „Hansestadt Rostock 3D“

4 Blender

Das nachfolgende Kapitel befasst sich speziell mit dem Programm Blender. Zunächst werden Formate näher erläutert, welche sich in der Vergangenheit bewährt haben. Im Anschluss daran wird das hiesige Projekt erörtert, besonders wird auf die Datengrundlage eingegangen sowie auf die Erstellung des eigentlichen 3D-Modells. Abschließend wird auf Softbodies eingegangen, eine Möglichkeit physikalische Faktoren in einem Blender Projekt einfließen zu lassen.

4.1 Dateiformate in Blender

Um Blender über die eigenen Grenzen hinaus nutzen zu können, bietet das Programm eine Vielzahl von Import- und Exportformate. Damit ist gesichert, dass Modelle von anderen Programmen in Blender weitergenutzt werden können. Tabelle 4.1 gibt einen Überblick über die Dateiformate, welche in Blender Im- und Exportiert werden können. Einige der in Tabelle 4.1 genannten Dateiformate dienen lediglich der Darstellung einer Spielewelt in Computerspielen, wie zum Beispiel Quake 3. Aus dem zuvor genannten Grund werden anschließend lediglich die wichtigsten und für Modellierungsarbeiten relevanten Dateiformate näher definiert. Die Reihenfolge der beschriebenen Formate gibt hierbei keine Gewichtung des einzelnen Formates wieder.

4.1.1 VRML

VRML ➠ Virtual Reality Modeling Language ist eine Beschreibungssprache von 3D-Szenen und zugleich eine HTML Erweiterung, um 3D-Szenen im Browser darzustellen. 1995 wurde VRML1.0 vom Computerhersteller Silicon Graphics entwickelt und den Computernutzern zugänglich gemacht. VRML1.0 diente der Erzeugung einer virtuellen Welt mit beschränkten interaktiven Verhalten. 1997 entwickelte Silicon Graphics VRML2.0. Die Erweiterung wurde nach einigen Veränderung als VRML97-ISO 14772 Standard definiert. Im Vergleich zu VRML1.0 ist VRML97 in der Lage virtuelle Welten noch realistischer darzustellen. Hintergründe zur Hauptszenerie können definiert werden sowie Audioeffekte dem Geschehen hinzugefügt werden. Weiterhin kann der Nutzer mit Objekten im Geschehen interagieren, darüber hinaus ist der Animationseffekt um einiges erhöht worden. [Moß08]

Aufgrund des enormen Verbreitungsgrades von VRML ermöglichen die meisten 3D-Modellierungswerkzeuge den Im- und Export des VRML Formates. Aufgrund dieser Tatsache entwickelte sich VRML zu einem Austauschformat von 3D-Modellen. Dadurch erfüllt VRML im Bereich der Interoperabilität die im Abschnitt 2.1 aufgestellte Forderung. Ein ähnliches Bild zeichnet sich in der Forderung der Generalisierung ab. VRML ist in der Lage im Anwendungsmodell mittels LoD Darstellungen zu vereinfachen bzw. zu präzisieren. Im Bereich von Multimedia kann VRML mittels Sensoren auf Benutzeraktionen reagieren, weiterhin sind über Time-Sensors Animationen möglich. Eigenschaften können mit VRML lediglich durch Rasterbilder dargestellt werden, somit ist die Forderung im semantischen Bereich nur eingeschränkt zu erfüllen.

4.1.2 X3D

X3D ➠ Extensible 3D. Ähnlich wie bei VRML können mit X3D 3D-Welten und interaktive Anwendungen in Echtzeit realisiert werden. Im Vergleich zu VRML bietet X3D wesentlich mehr standardisierte Möglichkeiten. Die Syntax von X3D kann wahlweise auf XML oder auch VRML aufbauen. X3D stellt seit 2002 den Nachfolger von VRML dar und ist seit Dezember 2004 durch die ISO Gruppe standardisiert. Dank der Komprimierungstechnik MPEG 4 benötigen X3D-Dateien wenig Speicherplatz. [Ros08] Im Vergleich zu VRML stehen bei X3D mehr standardisierte Möglichkeiten und Schnittstellen bereit. Die wesentliche Änderung ist jedoch der unterschiedliche Aufbau beider Sprachen. X3D ist im Gegensatz zu VRML modular aufgebaut. Die ganze Spezifikation ist dabei in sogenannte Profile unterteilt, die sich in Komplexität und Funktionalität unterscheiden. Insgesamt wird zwischen sechs Profilen unterschieden:

- Core Profile Minimale Datei-Informationen, wobei die verwendeten Komponenten explizit angegeben werden müssen
- Interchange Profile Minimale Implementation, erlaubt beschränkten Lichteinsatz, jedoch keine Interaktion
- Interactive Profile Erweiterte Implementation, die eingeschränkte Interaktion erlaubt
- MPEG-4 interactive Profile Entspricht dem MPEG-4 Standard
- Immersive Profile Entspricht dem VRML Standard
- Full Profile Volle Implementation der X3D Spezifikation

Die im Abschnitt 4.1.1 VRML gemachten Angaben zu den Forderungen lassen sich genauso auf das Format X3D übertragen. Ein Grund, warum sich VRML bei den gleichen Ergebnissen der Forderungen nicht durchgesetzt hat, ist die Tatsache, dass es kein Browser-Plug-In gibt, welcher die komplette Spezifikation implementiert hat. Unterschiedliche Browser können somit eine Szene anders darstellen. Dieses Problem sollte bei X3D aufgrund der Modularität

nicht auftreten. [Kro08]
Zur Nutzung von X3D Dateien in einem Browser, wird standardmäßig ein Plug-In benötigt, diese können unter http://cic.nist.gov/vrml/vbdetect (Juli 2008).html herunter geladen werden.

4.1.3 Collada

Collada ➠ Collaborative Design Activity ist eine offenes Austauschformat in der 3D-Modellierung, es ist ein XML-basiertes Format. Bei dem Colladaformat geht es nicht nur um den reinen Geometriedatenaustausch von 3D-Modellierungen, sondern auch Veränderungsschritte und Programmeinstellungen an einem Colladaobjekt beim Datenaustausch mit zu transferieren. Collada hat sich den vergangenen Jahren zu einem populären Format entwickelt und erlangte durch Sony den Durchbruch. Derzeit unterstützen Firmen wie Google, Intel, Apple und IBM die Entwicklung von Collada. Die berühmteste Anwendung von Collada findet jeder Computer User in der Anwendung von Google Earth. Hierbei trägt die Collada Datei die Textur- und Geometrieinformation des Google Earth Objektes, während ein weiteres Datenformat (KML ➠ Keyhole Markup Language) die Standortdaten des Google Earth Objektes in sich trägt. Aufgrund der enormen Unterstützung aus der Industrie, in der Weiterentwicklung von Collada, ist in den nächsten Jahren ein Durchbruch des Colladaformates wie bei 3DS zu erwarten.

Collada ist ein, in Kooperation mit Sony etablierter und von der Khronos Group kontrollierter 3D-Daten Standard. Collada ermöglicht den 3D-Daten Austausch zwischen einer Menge Dritthersteller-Programme und Software Tools. Durch den zuvor genannten Aspekt wird die Forderung nach Interoperabilität befriedigt. Forderungen der Generalisierung werden in einfacher Weise wie bei X3D realisiert, so wird zum Beispiel durch die Entfernung zwischen Betrachtungsstandort und 3D-Objekt entschieden, ob das 3D-Objekt sichtbar ist oder nicht. Ähnlich wie bei X3D sind sequentielle- und nichtsequentielle multimediale Darstellungen möglich sowie interaktive Anwendungen. Im Bereich der Semantik kommt Collada erst mit Ergänzung von KML den semantischen Forderungen nach. KML definiert einige Geometrieelemente in Anlehnung der Syntax von GML (Point, LineString, LinearRing, Polygon), hat aber ansonsten keinen gemeinsamen Basisstandard mit GML. KML erlaubt zwar die Repräsentation von Sachinformationen und verleitet dazu, KML als Austauschformat von Geodaten zu verwenden. Dies sollte jedoch vermieden werden, da KML im Unterschied zu GML nicht standardisiert ist. [Kol08b]

4.1.4 3DS

3DS ➠ 3D Studio ist das populärste Datenformat in der 3D Modellierung. Publiziert und vertrieben wird das Format 3DS derzeit durch die Softwarefirma Autodesk, in deren Produkt 3D Max. Aufgrund der populären Historie der Software 3D Max (ehemals 3D Studio bzw. 3D Studio Max) hat sich das Format 3DS zu dem weitverbreitestenden Austauschformat entwickelt. Nahezu jedes 3D-Modellierungsprogramm unterstützt in der Bearbeitung 3DS. Grundlegend enthält 3DS sämtliche Geometriedaten eines Objektes, Angaben zu extern gelegenen Texturen, Eigenschaften des 3DS-Objektes sowie Angaben zu einer vorhandenen Beleuchtung. Auf der Grundlage das 3DS kommerziell durch die Software 3D Max vertrieben wird, werden in Entwicklercommunity lediglich Mutmaßungen über den Aufbau des Formates 3DS angestellt. Diese Mutmaßungen wurden bisher nicht durch Autodesk bestätigt. Die bekannteste Ausführung zum Aufbau des 3DS-Formates kann unter http://the-labs.com/Blender/3dsspec.html (Juli 2008) nachgelesen werden.

Durch den Bekanntheitsgrad von 3DS sind die meisten Modellierungswerkzeuge in der Lage 3DS-Modelle zu Im- und Exportieren. Der genannte Bekanntheitsgrad sichert somit die Forderung der Interoperabilität aus dem Abschnitt 4.1.1. In Bereichen der Generalisierung und Multimedia kann aufgrund der kommerziellen Verwendung von 3DS keine Aussage getroffen werden. Die Forderung der Semantik kann durch extern gelegene Rasterbilder minimal befriedigt werden.

4.2 Datengrundlage des Gebäudes 23 in Bonnland

Das im Projekt verarbeitete Objekt, ist ein Haus (Gebäude 23) aus dem abgesiedelten Dorf Bonnland auf dem Truppenübungsplatz Hammelburg. Alle notwendigen Daten zur Erstellung des Modells stammen vom AGeoBw. Zum Einsatz kamen lediglich ein digitaler Grundrissplan sowie digitale Fotos. Die verwendeten Fotos sind durch einen Mitarbeiter des AGeoBw vor Ort im Monat April 2008 gemacht worden. Für das umliegende Gelände im Projekt, ist durch das AGeoBw ein digitales Oberflächenmodell[1] zur Verfügung gestellt worden. Dieses DOM basiert auf ein neuartiges photogrammetrisches Verfahren, welches durch die Bundeswehr entwickelt wurde.

4.3 Datenqualität der Grundlagendaten des Gebäudes 23

Das durch die Bundeswehr zur Verfügung gestellte DOM beinhaltet eine Punktwolke von ganz Bonnland, mit ca. 12 Millionen Punkten. Die Punktwolke liegt in einer Rasterweite von 15cm x 15cm vor und erstreckt sich über eine Fläche von ca. 0,3km^2. Eine Aussage

[1] DOM

Dateiformat	**Export**	**Import**
VRML 1.0 (.wrl)	ja	ja
VRML 97 (.wrl)	ja	nein
3D Studio (.3ds)	ja	ja
AC3D (.ac)	ja	ja
Autodesk DXF (.dxf)	nein	ja
Autodesk FBX (.fbx)	ja	nein
Collada 1.3.1 & 1.4 (.dae)	ja	ja
Cal3D (.cfg .xaf . xsf .xmf .xrf)	ja	nein
DEC Object File Format (.off)	ja	ja
DirectX (.x)	ja	ja
LightWave (.lwo)	ja	ja
LightWave Motion (.mot)	ja	nein
Load MDD to Mesh RVKs	nein	ja
MD2 (.md2)	ja	ja
Motion Capture (.bvh)	nein	ja
Open Flight (.flt)	ja	ja
OpenInventor (.iv)	ja	nein
Paths (.svg; .ps; .eps; .ai; GIMP)	nein	ja
Pro Engineer (.slp)	nein	ja
Quake 3 (.map)	ja	nein
RAW Faces (.raw)	ja	ja
Save Current Theme	ja	nein
SoftImage XSI (.xsi)	ja	nein
Standfort PLY (.ply)	ja	ja
Vertex Keyframe Animation (.mdd)	ja	nein
Wavefront (.obj)	ja	nein
X3D Extensible 3D (.x3d)	ja	nein

Tabelle 4.1: Standardschnittstellen in Blender v2.45

Abb. 4.1: Gebäude 23, umgeben von Gebäuden in LOD 1

über die Genauigkeit des DOM kann nicht getroffen werden, aufgrund der noch laufenden Entwicklungsphase des photogrammetrischen Verfahren. Die Genauigkeit des DOM ist aber im cm Bereich anzusiedeln.

Der zugrunde liegende Grundrissplan des Gebäudes 23 liegt als Rasterdatei vor und stellte sich lediglich als Scan des Originals heraus und ist auf Februar 1985 datiert. Der Scan selbst ist teilweise sehr verzerrt, hat lediglich eine Auflösung von 96 dpi und ein Ausmaß von 3262 x 2306 Pixel. Aufgrund des Alters von 23 Jahren besitzt der Grundrissplan nicht die erforderliche Aktualität. Die nicht gegebene Aktualität des Grundrissplanes belegt sich beispielsweise im veränderten Mauerwerk auf der Südseite des Gebäudes sowie in abgeänderten Wänden innerhalb des Gebäudes.

Die im Vorfeld benannten Fotos, wurden zu Texturzwecken verwendet und zum Vergleich der Örtlichkeit mit dem Modell. Sie besitzen eine Auflösung von 180 dpi.

Höhenunterschied Bw-Modell ↔ Projekt-Modell

Das im Projekt verarbeitete DOM weist um das Gebäude 23 Höhendifferenzen auf. Diese Differenz wird insbesondere auf der Nordseite des Gebäudes 23 deutlich. In Abbildung 4.2 ist zu erkennen, dass der Grund des DOM im Vergleich zum Geländer auf der Nordseite des Gebäude 23 zu hoch erscheint. Die Ursache hierfür liegt in der Konvertierung des Bw-DOM nach Blender. Das originale Bw-DOM erscheint mit seinem Ausmaß mit ca. 12 Millionen Punkte zu groß, aufgrund der genutzten Rechnertechnik ist es Blender nicht möglich das originale DOM zu Händeln. Zur Gewährleistung des Imports wurde das originale DOM über verschiedene Zwischenschritte (Abschnitt 4.4) bearbeitet, weiterhin wurden die Punktanzahl deutlich verringert auf ca. 65.500. Aufgrund des Ausdünnen der Punktanzahl verlor das DOM Stützpunkte, welche die Genauigkeit garantierten. Durch das späteren Aufspannen der Fläche über die noch vorhandenen Stützpunkte erscheint der Grund des DOM in seiner Höhe verkehrt. Durch Nutzung einer stärkeren Rechnertechnik ist es möglich die genutzte Punktanzahl hinauf zu setzen und somit eine bessere Genauigkeit wiederzugeben.

Abb. 4.2: Nord- und Westseite des Gebäude 23, Verschnitt mit dem DOM der Bundeswehr sowie dem modellierten Gebäude

4.4 Import des Bw-DOM in Blender

Wie im Abschnitt 4.2 beschrieben ist im Modell ein DOM der Bundeswehr eingefügt. Das DOM ist so aufgearbeitet, dass es problemlos mit ArcScene oder ArcMap verarbeitet werden kann. Aufgrund der hohen Punktanzahl ist eine Bearbeitung mit einem herkömmlichen Heim-PC nur unter großem Zeitaufwand möglich. Daher ist das DOM durch eine PC-Anlage der Bundeswehr auf einem Ausschnitt von ca. 50m um das Gebäude 23 begrenzt worden.

Für den Import des Bw-DOM nach Blender, wurden 3 externe Programme genutzt sowie ein Python Script.

- ArcMap[2]
- 3DEM[3]
- Terragen[4]
- Python Script „ter2blend“[5]

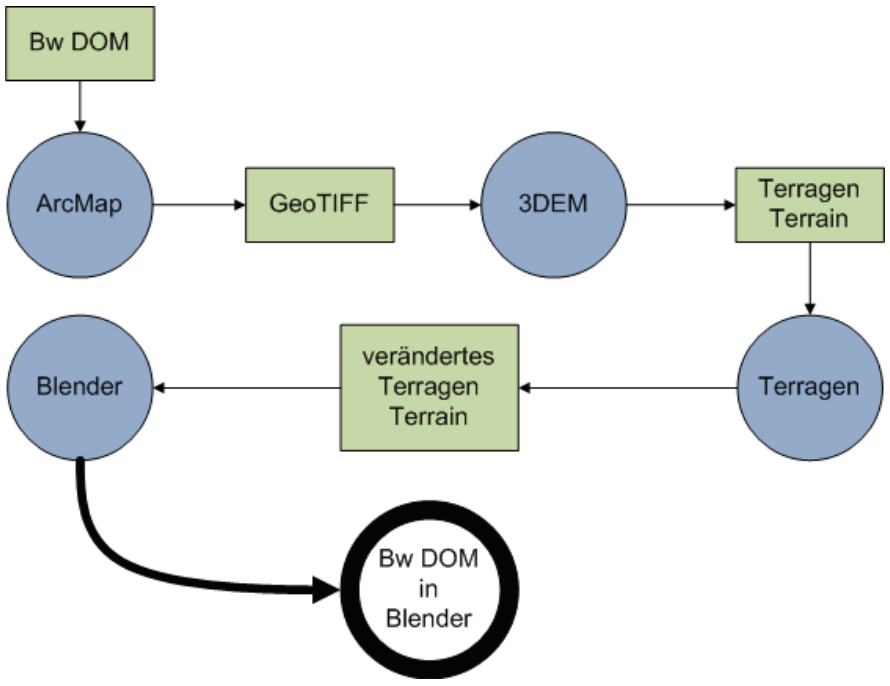

Abb. 4.3: Import Bw-DOM nach Blender

Abbildung 4.3 gibt grob den Import des Bw-DOM nach Blender wieder. Zum Import wurde das Bw-DOM in ArcMap eingeladen und mittels 3D Analystfunktionen ein Raster erstellt,

[2]Informationen unter: http://www.esri-germany.de/products/arcgis/about/arcmap.html (Juli 2008)
[3]Informationen unter: http://www.visualizationsoftware.com/3dem.html (Juli 2008)
[4]Informationen unter: http://www.planetside.co.uk/ (Juli 2008)
[5]Informationen unter: http://projects.blender.org/projects/ter2blend/ (Juli 2008)

hierbei wurden die Funktionen „Create TIN from Feature“ und „TIN to Raster“ benutzt. Das erzeugte Raster wurde mit ArcMap Standardeinstellungen als GeoTIFF exportiert. Das erzeugt GeoTIFF wurde anschließend mit 3DEM geladen und als Terragen Datei gespeichert, über „Save Terragen Terrain“ und „Entire Terrain“. Die erzeugte Terragen-Datei wird mit dem Programm Terragen geladen. Nach dem Laden der Datei wurde das Einheitssystem auf das System von Terragen umgestellt. Erfolgt keine Umstellung des Einheitssystems, besteht die Gefahr, dass das spätere Modell in Blender in eine unüberschaubare Höhe importiert wird. Zusätzlich muss in Terragen die Kameraposition und Kameraorientierung notiert werden. In Blender wird das Python Script „ter2blend“ gestartet und im darauf folgenden Menü die Terragen Datei ausgewählt sowie die Kameraposition und die Kameraorientierung angegeben. Nach dem Import der Terragen Datei in Blender erscheint eine Fehlermeldung[6] über das Konsolenfenster „Blender Lampe ohne Distanz“ diese Meldung kann ignoriert werden. Die durch den Import erzeugte Kamera und Lichtquelle kann bei Bedarf gelöscht werden.
Alternativ kann im letzten Schritt auf Python verzichtet werden, wenn zuvor in Terragen das Terrain als .lwo[7] Datei exportiert wird. Die benannte .lwo Datei kann von Blender problemlos importiert werden. Jedoch kann nur eine Fläche von 256 x 256 Pixel als .lwo Datei exportiert werden, im vorliegenden Projekt wäre damit lediglich das Gebäude 23 erfasst, eine flächendeckende Erfassung ist derzeit mit dem .lwo Format nicht realisierbar.

4.5 Modellierungsarbeiten

Zum Verständnis des Abschnittes Modellierung werden Grundkenntnisse von Blender vorausgesetzt. Es werden Funktionen benannt, die in ihrem Wesen nicht weiter erläutert werden.

Im Vorfeld der Modellierungsarbeit wurde der Grundrissplan zum Gebäude 23 mit der Software Photoshop entzerrt. Im Anschluss der Entzerrung sind die einzelnen Etagen aus dem Grundrissplan heraus geschnitten worden. Zum Entzerren kam die Photoshopfunktion „Transformieren“ und zum Ausschneiden die Funktion „Freistellungswerkzeug“ zum Tragen.

4.5.1 Modellierung der Innenwände

Grundlage der Modellierung der Innenwände bilden die ausgeschnittenen und entzerrten Etagenpläne aus dem Grundrissplan. Des Weiteren erfolgte die nachfolgende Modellierung im X-Y Raum.

[6] bei Blender v2.45
[7] LightWave Objec

Nach Anlegen eines neuen Blenderprojektes ist der Etagenplan des Kellers mittels Background Image... im Projekt hinzugefügt worden. Nach Anlegen eines neuen Mesh, im vorliegenden Fall eine Plane, sind die Innenwände des Kellergeschosses nachdigitalisiert worden. Hierzu ist die erschaffene Plane mittels der Move und Scale Funktion auf eine Wand des Etagenplanes verschoben worden. Abbildung 4.4 zeigt einen Teil der Innenwände des Kellergeschosses. Der rote Kreis in Abbildung 4.4 markiert die neu erzeugte Plane. Von der neu erschaffenden und positionierten Plane wird sukzessive das Mauerwerk abdigitalisiert, in Abbildung 4.4 sind die einzelnen Stufen der Digitalsierung zu erkennen. In der weiteren Digitalsierung kommt die Funktion Extrude ➟ Only Edges entscheidend zum Tragen. Bei Benutzung der Extrude Funktion ist es hilfreich die Richtung (X- oder Y-Richtung) mit anzugeben. Nach Abschluss

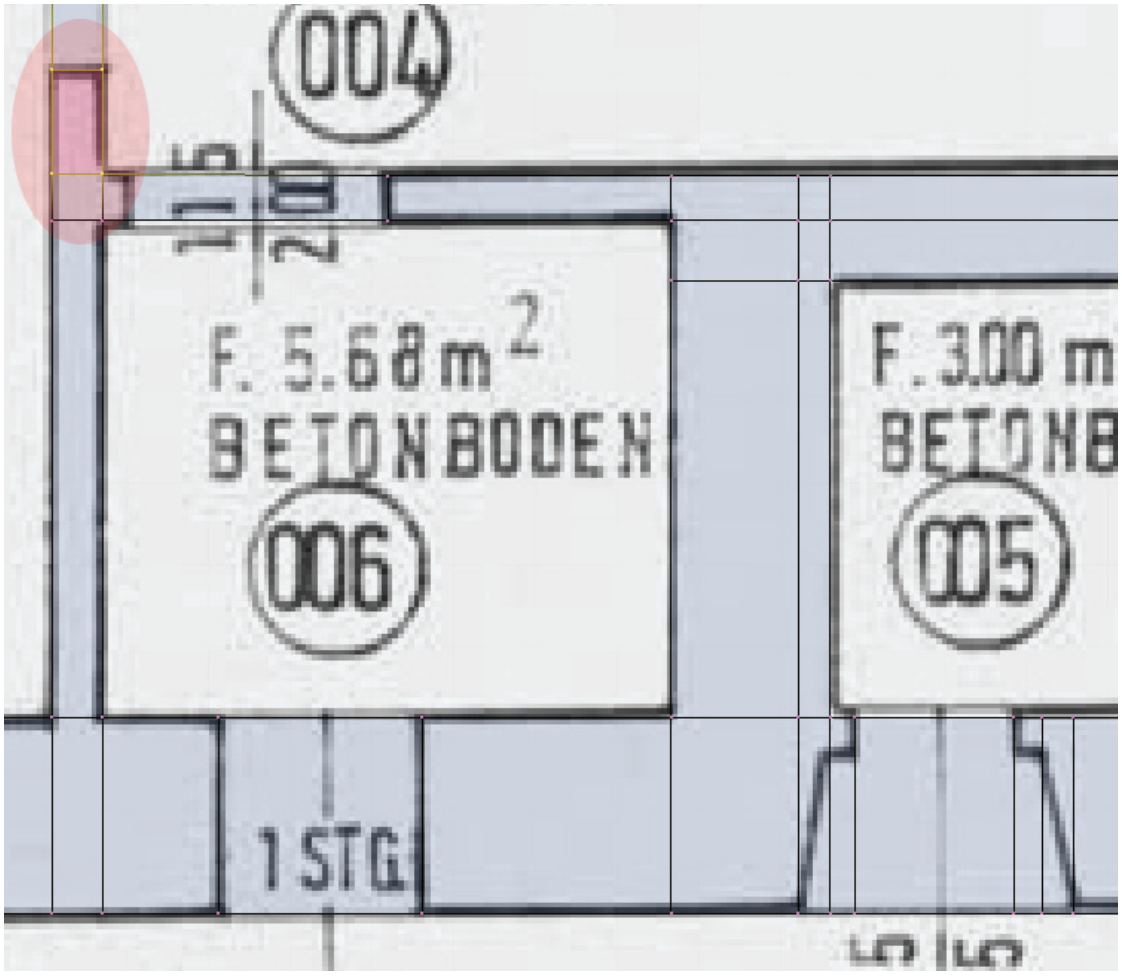

Abb. 4.4: Digitalsierung der Innenwände

der Digitalisierung muss das Ende der Digitalisierungsfläche mit dem Anfang verbunden werden. Zum Verbinden zweier Flächen oder Linien ist die Funktion Use Snap or Grid äußerst effektiv. Nach Aktivierung der Use Snap or Grid Funktion ist diese in Kombination mit der Strg-Taste zu nutzen. Nach Abschluss der Digitalisierungsarbeiten werden alle doppelten

Knoten (Vertices) und Kanten (Edges) gelöscht, mittels der Funktion Remove Doubles über den Menüpunkt Specials.

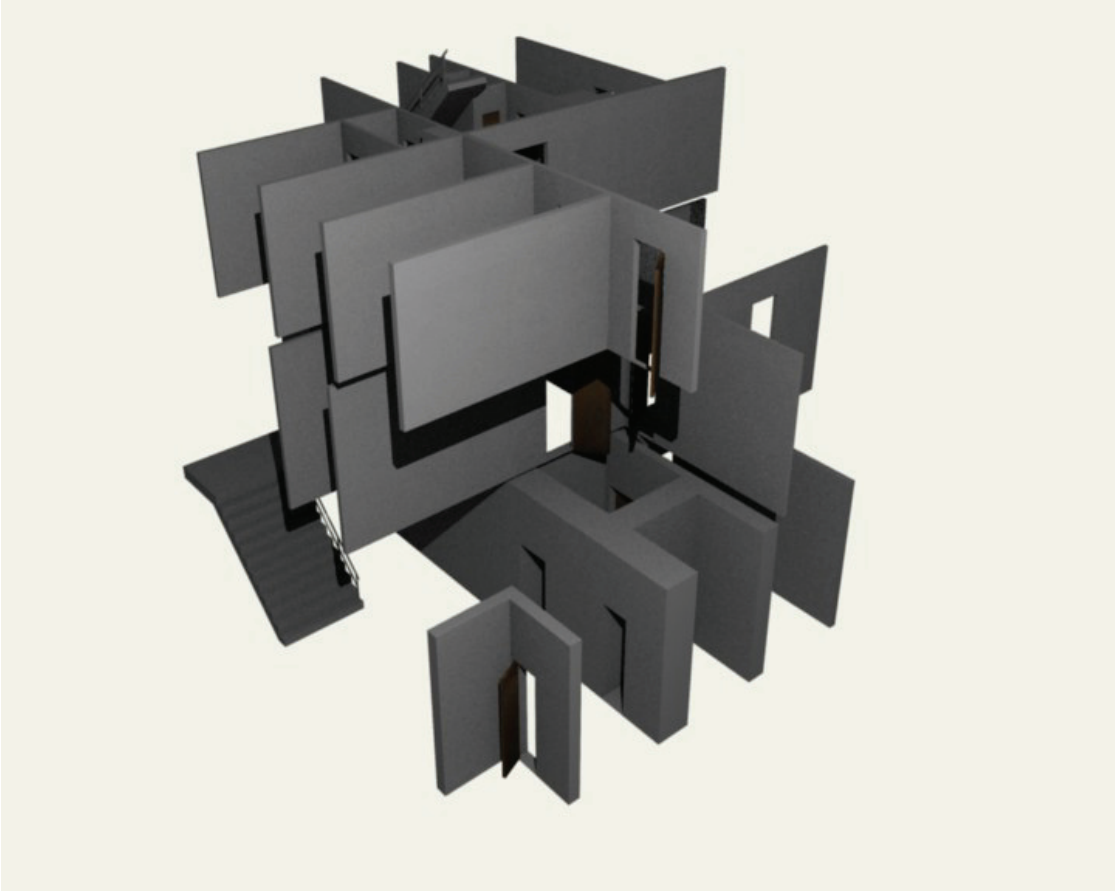

Abb. 4.5: Modellierte Innenwände

Das erstellte 2D Modell liegt derzeit in einem einheitslosen System vor. Ziel ist es dem Modell das richtige Ausmaß zu geben. Hierzu wird in dem Object Mode gewechselt und anschließend das zuvor erstellte Modell des Kellergeschosses ausgewählt. Mittels der Transform Properties ist es möglich dem Modell in der X-Richtung sowie Y-Richtung die richtigen Ausmaße zu geben. Hierzu werden die Maße aus dem Grundrissplan entnommen.

Nach Abschluss der Digitalisierungsarbeiten und der Transformation liegt der Grundrissplan in einer 2D Ebene vor. Nun müssen die Wände in die Y-Richtung extrahiert werden, dazu werden alle Flächen im Edit Mode ausgewählt. Flächen die für Türen vorgesehen sind werden bei der Auswahl ausgelassen. Nach Auswahl aller notwendigen Flächen werden diese mittels der Extrude Funktion in die Y-Richtung extrahiert. Zu beachten ist hierbei der Abschnitt, an denen der Freiraum der Türen nach oben hin aufhört. Aus dem eben genannten Grund werden die zuvor ausgewählten Flächen zweimal extrahiert ➟ vom Boden bis zum Ende

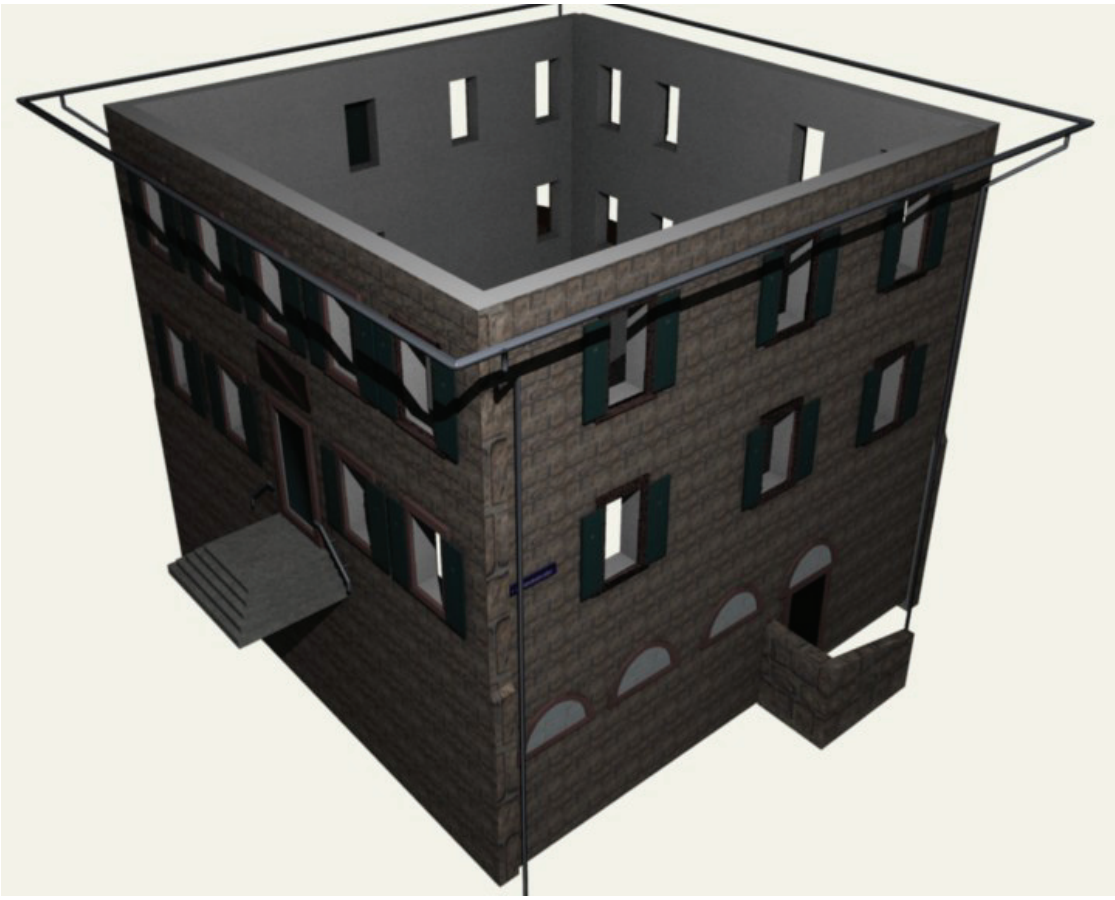

Abb. 4.6: Modellierte Außenwände

des Freiraumes der Türen ➟ vom Ende des Freiraumes der Türen bis zum Beginn der Decke (Abbildung 4.7). Ab der zweiten Extraktion werden die Freiflächen der Türen zuvor geschlossen und zur Extraktion ebenfalls mit ausgewählt.

Abschließend liegt das Kellergeschoss als 3D Modell vor. Auf gleicher Art und Weise werden die verbleibenden Etagen modelliert, zur besseren Kontrolle sollte jede Etage in einem eigenen Layer angeordnet werden. Nach Fertigstellung aller Etagen können diese in einem Layer zusammengefasst werden (Abbildung 4.5).

4.5.2 Modellierung der Außenwände

Im Gegensatz zur Modellierung der Innenwände werden die Außenwände von Fotos ab digitalisiert. Grund für diese Vorgehensart ist die zusammenhängende Texturfläche jeder Hausseite. Voraussetzung für diese Art der Modellierung sind entzerrte Fotos der Hauswände. Die Arbeitsweise der Modellierung der Außenwände ist die gleiche wie im Abschnitt 4.5.1 (Abbildung 4.6).

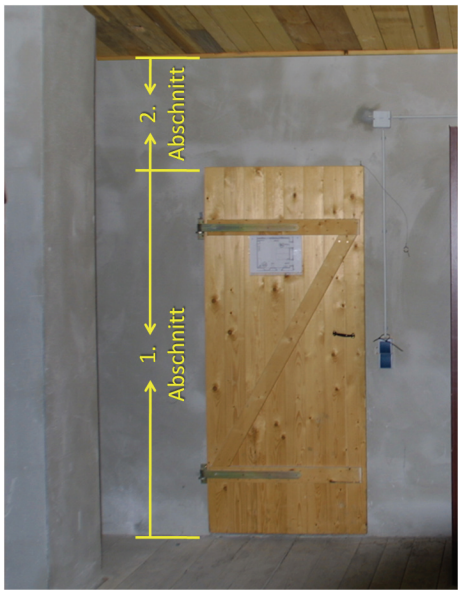

Abb. 4.7: Abschnitte bei der Innenwandmodellierung

Abb. 4.8: Gerenderte Tür, als Vergleichsbild

4.5.3 Lichtquellen im Projekt

Blender benutzt zur Berechnung des Lichteffektes den Normalenvektor von Flächen. Das bedeutet, jedes einzelne Pixel im gerenderten Bild besitzt einen Normalenvektor. Die Ausleuchtung eines gerenderten Pixels richtet sich nach dem Winkel zwischen dem Normalenvektor und der Lichtquelle. Je spitzer der Winkel zwischen Normalenvektor und Lichtquelle desto stärker die Ausleuchtung (Abbildung 4.9).

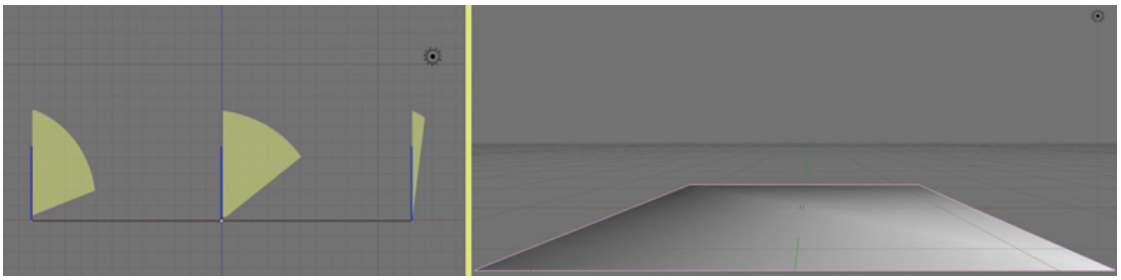

Abb. 4.9: Ausleuchtung einer Fläche, unter Beachtung des Winkel zwischen Flächennormale und Lichtquelle

Auf die Erläuterung der verschiedenen Lichtquellen soll verzichtet werden. Stattdessen wird auf die 3 Punkt Beleuchtung eingegangen werden, um in einer 3D Szene ein spezielles Detail hervorzuheben.

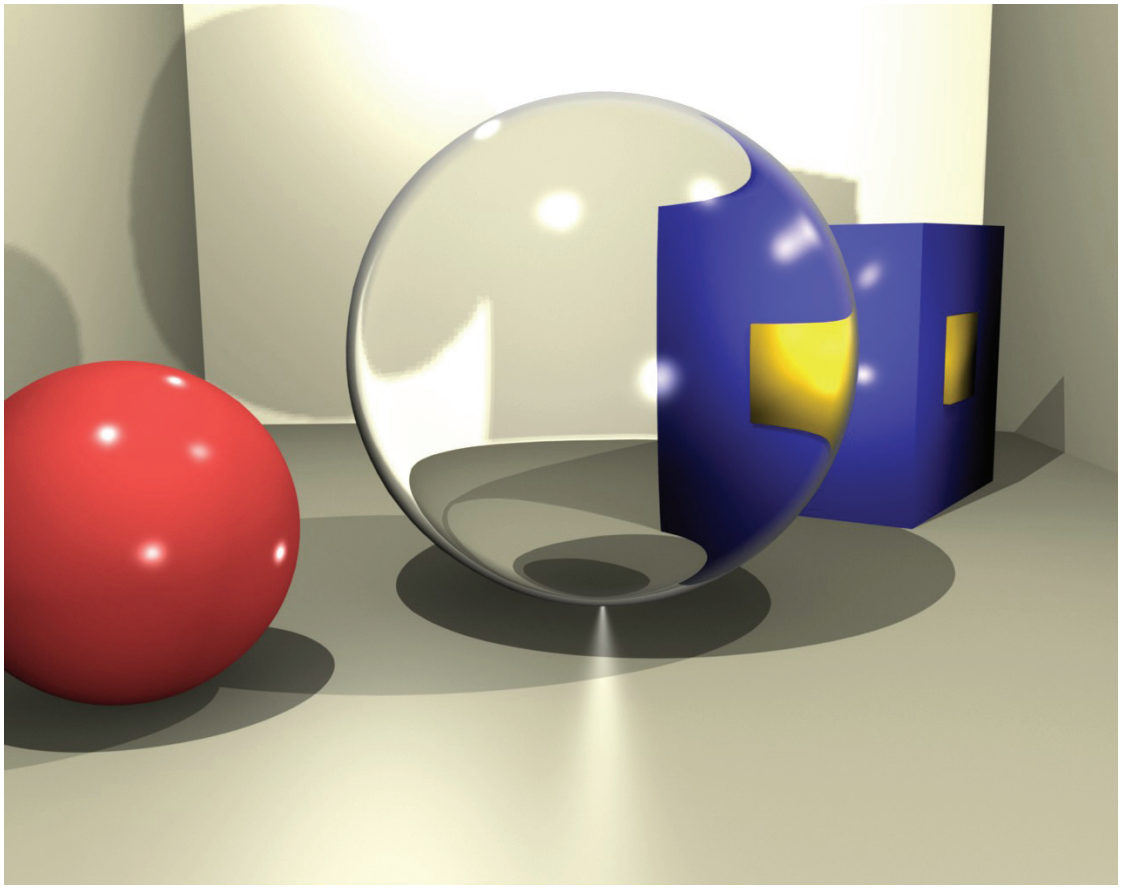

Abb. 4.10: Objekte mittels der 3 Punkt Beleuchtung in Szene gesetzt

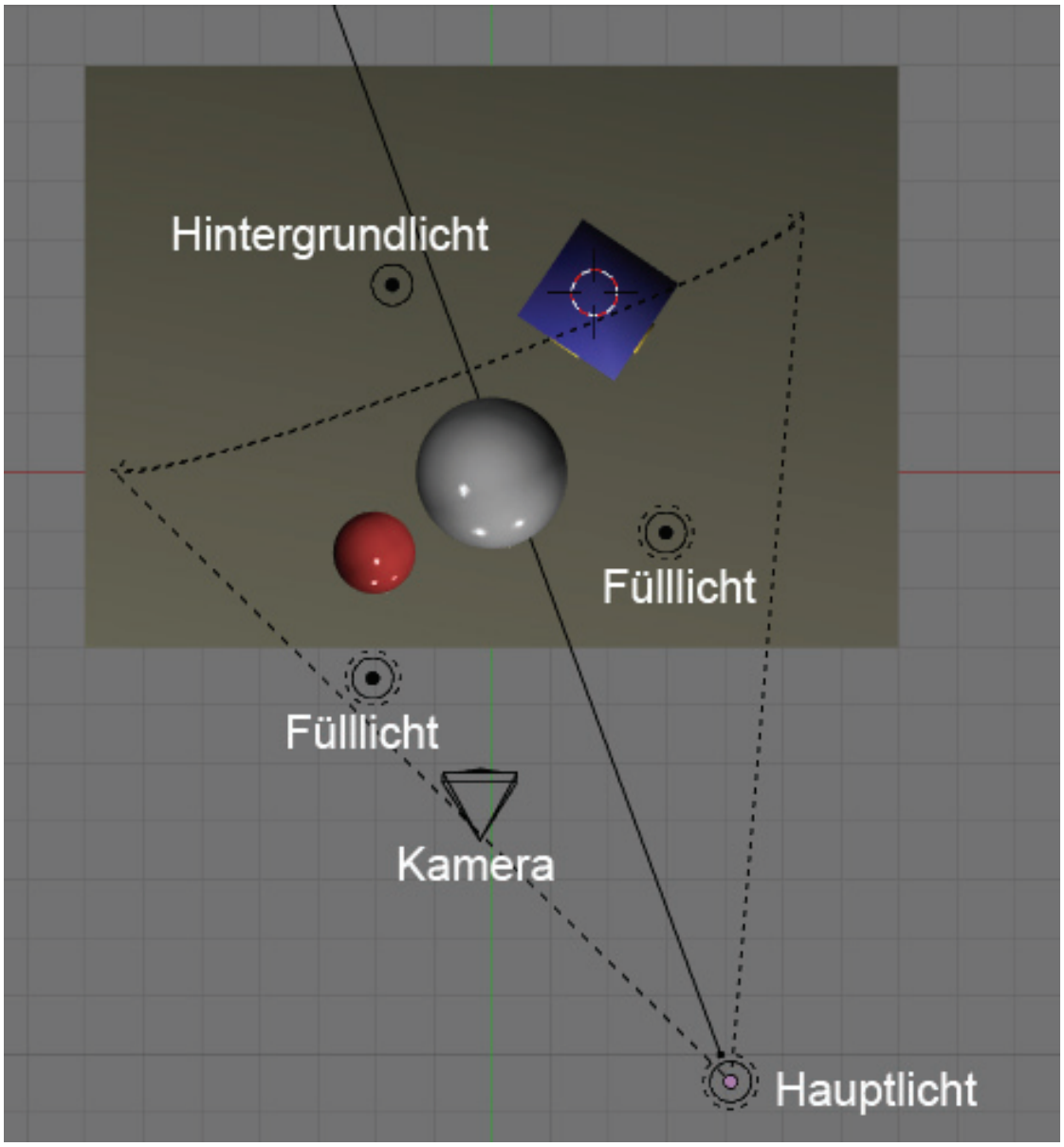

Abb. 4.11: Aufstellung der 3 Punkt Beleuchtung

Am Beispiel einer Glaskugel (Abbildung 4.10) soll die 3 Punkt Beleuchtung (Abbildung 4.11) wiedergegeben werden. Notwendig ist ein Hauptlicht, ein Fülllicht sowie ein Hintergrundlicht. Das Hauptlicht leuchtet das Objekt von einer Seite aus. Je nach Motiv kann das Licht des Hauptlicht eingefärbt sein, oftmals bieten sich hierfür Rottöne an. Aufgrund des Spotlight als Hauptlicht wird die nicht angestrahlte Seite des Objektes in Schatten gehüllt. Ein Fülllicht auf der abgeschatteten Seite des Objektes erhellt den Schatten und die darin liegenden Details. Durch das Hauptlicht und das Fülllicht ist das Objekt ausgeleuchtet, jedoch gehen durch die bisher verwendeten Lichter Konturen bzw. der Kontrast des Objektes verloren. Mittels eines Hintergrundlichts kann der Kontrast angehoben werden.
Ein wichtiger Punkt bei der Benutzung eines Spotlights als Hintergrundlicht ist die Aktivierung des Lichtes auf einen Layer. Wird die zuvor genannte Funktion nicht aktiviert enthält das gerenderte Bild einen zweiten Lichtkegel. Voraussetzung zur Vermeidung eines zweiten Lichtkegels ist die Anordnung des Objektes, des Hintergrundlichtes in einem Layer sowie der Aktivierung von Layer im Shading/Lamp Buttons Bereich.

Im vorliegenden Projekt ist die Lichtquelle Area Light verwendet. Die Lichtquelle Area Light ermöglicht einen weichen Schatten sowie eine bessere Handhabung und bessere Einstellungsmöglichkeiten. Weiterhin vereint Area Light mehrere Lichtquellen in einer. Im Projekt sind für das Area Light folgende Einstellungen vorgenommen worden:

- Rechtwinklige Anordnung der Lichtquellen
- Nutzung von 3 x 3 Lichtquellen
- Distance-Wert von 21
- Energie-Wert von 0.85
- Gamma-Wert von 0.875
- Dither Funktion aktiviert

Die meisten Einstellungsmöglichkeiten in Area Light sind subjektiver Art. So entspricht der Distance-Wert nicht dem Abstand Lichtquelle zum Hauptobjekt, sondern ist ein Kompromiss zwischen Hauptobjekt und den anderen vorhanden Objekten. Weiterhin ist der Gamma-Wert ebenfalls subjektiver Art. Wird dieser erhöht erscheinen Konturen im gerenderten Bild um einiges härter. Die Nutzung von 9 Lichtquellen ist für das Modell-Gebiet als ausreichend empfunden. Die Aktivierung der Dither Funktion sorgt hierbei für einen noch weicheren Schatten.

Weiterhin sorgt das Ambient Occlusion/Umgebungslicht für die Beleuchtung im Projekt. Das Umgebungslicht ist als primäre Lichtquelle im Projekt gedacht, da bei Ambient Occlusion

das Licht von einer imaginären Halbkugel über das komplette Modellobjekt leuchtet. Bei Verwendung von Ambient Occlusion ist der Rechenaufwand immens höher, für die Berechnung der Abbildung 4.1 benötigte der PC[8] 44 Minuten. Darüber hinaus ist bei Verwendung der Ambient Occlusion mit einem körnigen Ergebnis zu rechnen. Der Grund für das körnige Ergebnis ist in den unendlichen vielen Lichtstrahlen zu suchen, welche durch die imaginäre Halbkugel ausgehen. Dem körnigen Ergebnis kann mittels einer cleveren Parametervergabe im Menü World Buttons/Amb Occ entgegengewirkt werden. Zum Beispiel ist durch einen höheren Wert bei Samples mit einem besserem Ergebnis zu rechnen. Leider vervielfacht sich die Berechnungszeit in etwa um den Faktor, welcher als Samples-Wert angegeben wird.

Zur Beleuchtung der Innenräume des Gebäude 23 ist in jedem Raum eine einzelne Lichtquelle angebracht, dabei handelt es sich um 30 einzelne Lampen/Lamps. Der Energie-Wert der Lamps ist bei 0.750 festgelegt. Der Farbwert ist bei 1 1 1 (RGB) belassen worden. Zusätzlich ist die Funktion Layer aktiviert.

4.5.4 Audio im Projekt

Obwohl in der Literatur und im Internet das hinzufügen von Audiosequenzen als sehr trickreich diskutiert wird, ist es relativ einfach eine Audiosequenz zum Beispiel in einem gerenderten Film hinzuzufügen. Zusätzlich ist jedoch ein externer Soundeditor notwendig. Nutzer von Windows können hierbei auf die Software Windows Movie Maker zurückgreifen.

Wenn in Blender eine Filmsequenz zusammengestellt wird, ist diese im Standardformat AVI und ohne Audioeffekte. Um einer Filmsequenz Audioeffekte hinzuzufügen sollten 16 Bit WAV Dateien verwendet werden. Diese WAV Datei wird über dem Audio Window mittels Adds New hinzugefügt. Nachdem Hinzufügen erscheint auch die WAV Datei im Sound Block Buttons Bereich. Im Sound Block Button Bereich kann die geladene WAV Datei konfiguriert werden, prinzipiell sind die Standardeinstellungen ausreichend. Anschließend wird die gleiche WAV Datei im Video Sequence Editor Fenster über die Add Funktion geladen und im Zeitdiagramm entsprechend positioniert, je nachdem an welcher Stelle die Audiosequenz benötigt wird (Abbildung 4.12 obere Hälfte). Nun können im Render Buttons Bereich Einstellungen zum Rendern des erstellten Filmes sowie der eingefügten Audiosequenz erfolgen (Abbildung 4.12). Lediglich das FFMpeg Format unterstützt in der Einstellung Audiosequenzen, dazu muss wie in Abbildung 4.12 FFMpeg eingestellt werden, darauf hin werden die Registerkarten Video und Audio zur Verfügung gestellt. In der Registerkarte Audio muss die Funktion Multiplex Audio aktiviert werden. Die Multiplex Audio Funktion führt beim Rendern nun die Audiosequenz mit Videosequenz zusammen.

[8] Intel Celeron, 3.06 GHz, 1 GB RAM; Grafikkarte ATI Radeon HD 2600 AGP, 512 MB

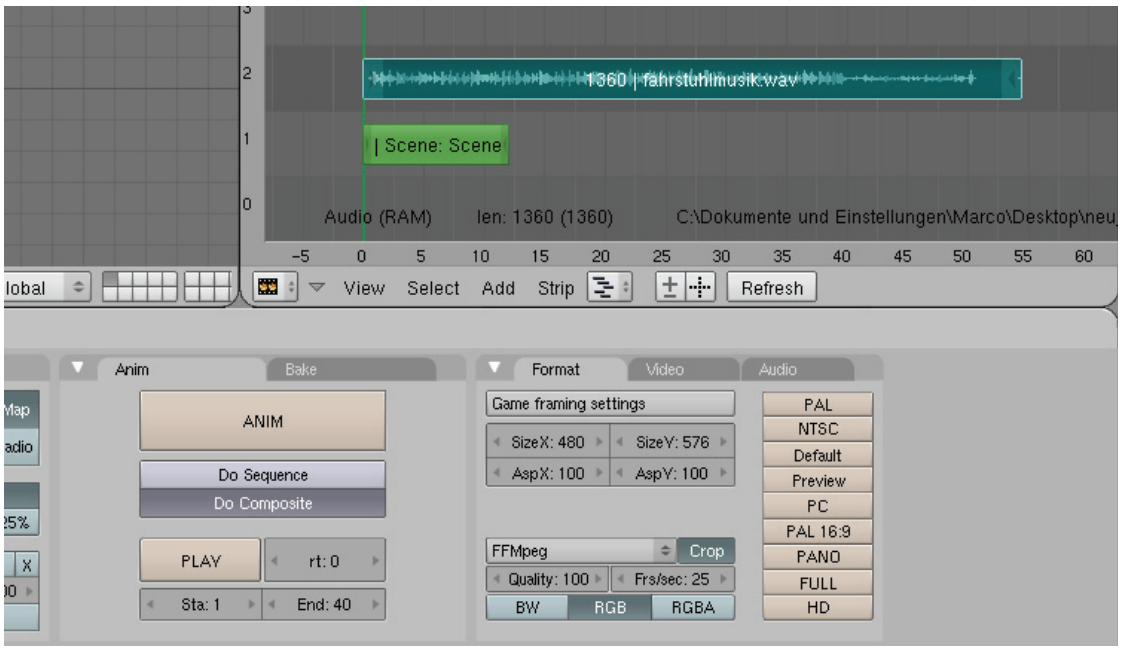

Abb. 4.12: Video Sequence Fenster, darunter Render Einstellungen im Scene Fenster

Das Ergebnis ist leider nicht überzeugend, daher ist es empfehlenswert eine Alternativlösung zu nutzen. In der Alternativlösung wird die Audiosequenz separat abgespeichert sowie die Videosequenz. Beide müssen im Nachhinein manuell zusammengeführt werden. Die Vorgehensweise ist ähnlich der oben genannten Lösung.
Eine WAV Datei wird über das Audio Window hinzugefügt, im Sound Block Buttons Bereich konfiguriert, im Video Sequence Editor geladen und positioniert. Daraufhin wird in den Sound Block Buttons Bereich gewechselt und mittels der MIXDOWN Funktion wird eine Kopie der Audiosequenz als WAV Datei in das Videoverzeichnis gespeichert. Das Videoverzeichnis ist standard das c:\tmp Verzeichnis. Zusätzlich muss der Anwender noch die Videosequenz rendern, hier empfiehlt sich AVI Codec. Die gespeicherte Audiosequenz kann nun problemlos mit der Videosequenz zusammengeführt werden, beide Dateien sind so konform gespeichert, dass sie 1:1 übereinandergelegt werden können.

4.5.5 Regen im Projekt/Partikelsystem

Zum Rendern von Bildern ist im Projekt ein Layer mit einem Regeneffekt vorgehalten. Der Regeneffekt basiert auf dem Partikelsystem und ist verhältnismäßig einfach umzusetzen. Da das Partikelsystem sehr Ressourcenlastig ist, ist der Layer deaktiviert.

Zum Erzeugen des Regeneffektes benötigt Blender zunächst ein Grid (z. Bsp. 5x20), auf dem das Partikelsystem aufbauen kann. Das erzeugte Grid wird an die Stelle verschoben, an die der Regeneffekt gewünscht wird. Nach der Positionierung wird das Grid abschließend in

Y-Richtung verschoben. Nun erfolgt die Erzeugung des Regentropfens, aus der Top Ansicht im 3D Fenster heraus und in der Mitte des zuvor erzeugten Grid. Die Erzeugung wird über Add/Mesh/Cone durchgeführt. Der Regentropfen sollte aufgrund einer zu geringen Rechnerleistung nicht zu viele Vertices besitzen, ideal sind 8 Vertices. Darüber hinaus kommen mehr Vertices aufgrund der Größe des Regentropfens nicht zur Geltung. Der Cone erscheint zunächst überdimensional und muss auf eine passende Größe skaliert werden. Aus harmonischen Gründen sollte die Oberfläche des Regentropfens mittels Subsurf geglättet und verfeinert werden. Abschließend werden Glanz, Reflektion und Transparenz im Shading Bereich eingestellt, für das vorliegende Projekt sind Werte wie in Abbildung 4.13 verwendet worden.

Nach der Konfiguration wird der einzelne Regentropfen und das Grid mit einander verbunden, mittels Make Parent (im Object Mode ➟ Strg+P), wichtig ist das der Regentropfen zu erst markiert wird. Anschließend geht es daran den Regentropfen mit Hilfe des zuvor erstellten Grids zu vermehren. Zur Vermehrung wird das Grid ausgewählt und ins Object Menü (F7) gewechselt, im Object Menü wird die Funktion Dupli Verts aktiviert. Daraufhin werden so viele Regentropfen erzeugt, wie Knoten/Stützpunkte im zuvor erstellten Grid existieren.
Anschließend wird ins Physics Button Menü gewechselt (Objekt (F7) ➟ Physics Button (F7)). Im Physics Button Menü wird für das markierte Grid ein neuer Particle Effect erstellt. Im Partikelsystem werden Einstellung in der Registerkarte Particles und Particle Motion vorgenommen, wie in der Abbildung 4.14 (blaue Registerkarten) angegeben.

Abb. 4.13: Einstellungen im Shading Bereich

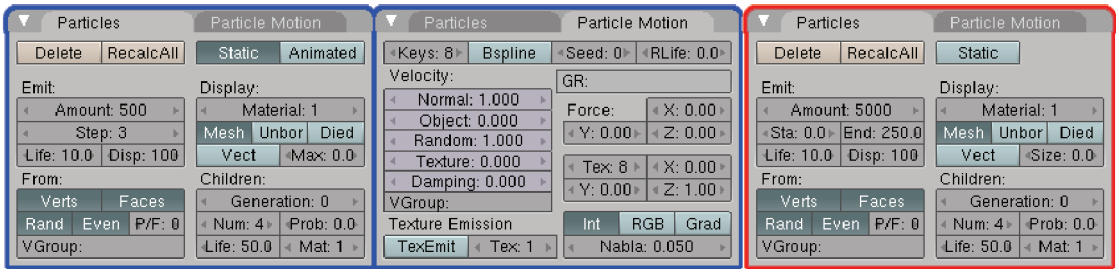

Abb. 4.14: Einstellungen im Physics Button Menü

Die vorgenommenen Einstellungen eignen sich lediglich zum Rendern von Rasterbildern, da in der Registerkarte Particles die Funktion Static ausgewählt wurde. Für Animationszwecke muss die Einstellung wie in Abbildung 4.14 (rote Registerkarte) verändert werden. Ein Test des animierten Regeneffektes kann mittels Alt+P im 3D Fenster durchgeführt werden.

4.5.6 Gras im Projekt/Partikelsystem

Für die Natürlichkeit des Untergrundes, von Grünflächen sind vereinzelt Flächen mit Gras im Projekt eingebracht. Aufgrund des großen Rechenaufwandes, sind die Grasflächen nur vereinzelt dargestellt. Zum Rendern müssen diese entsprechend der Kameraeinstellung verschoben werden. Die Herangehensweise zur Darstellung von Gras ist verhältnismäßig einfach gehalten, zum Einsatz kommt hier das Partikelsystem.

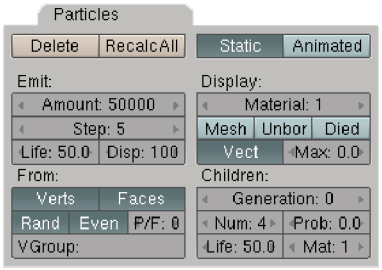

Abb. 4.15: Partikeleinstellungen im Particles Bereich

Grundlage ist eine eingefügte Fläche/Plane, auf diese Fläche werden die Grashalme generiert. Nach Anlegen einer neuen Fläche wird mittels des Physiscs Buttons ein neuer Partikeleffekt erzeugt. Nachfolgend werden entsprechend der Abbildung 4.15 folgende Einstellungen vorgenommen:

- Amount Faktor (Anzahl der Grashalme)
- Life Faktor (Länge der Grashalme)
- Aktivierung von Verts/Faces/Rand/Even/Vect (Grashalme aus Objekte (Vertices) generieren, verteilen der Grashalme auf der Fläche, zufällige Verteilung der Grashalme auf der Fläche)

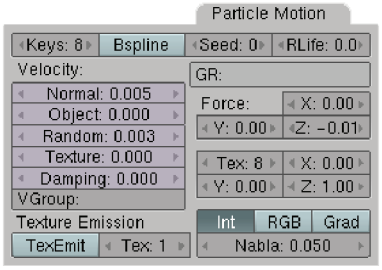

Abb. 4.16: Partikeleinstellungen im Particle Motion Bereich

Nach Einstellung im Particles Bereich, werden noch Faktoren im Particle Motion Bereich vergeben, entsprechend Abbildung 4.16:

- Normal Faktor (Grashalme in positive Z Richtung stehen lassen)
- Random Faktor (Anordnung der Grashalme mit einer gewissen Zufälligkeit)
- Force Z Faktor (Grashalme nach unten einknicken lassen)

Abschließend wird im Shading Bereich die Farbe der Grashalme vergeben, im vorliegenden Projekt sind das 0.837 0.923 0.382 (RGB). Weiterhin werden in der Strands Box, unter Links and Pipeline, die Anfangs- und Enddicke der Grashalme festgelegt, zu beachten ist hierbei das die Enddicke nur minimal 0.25 betragen kann. Die Werte im Strands Bereich sind wie folgt festgelegt worden:

Start ➟ 4.00 End ➟ 0.25 Shape ➟ 0.20.

Abschließend liegt ein Ergebnis wie Abbildung 4.17 vor.

Abb. 4.17: Ergebnis der Grasmodellierung

4.5.7 Zusätzliche Elemente im Projekt

Weiterhin sind im Projekt weitere Elemente eingefügt, dies sind unter anderem Bäume (Abbildung 4.19) und ein Panzer (Abbildung 4.18). Die genannten Elemente stammen von der Webseite http://www.the3dstudio.com (April 2008). Die Webseite stellt für Entwickler eine Plattform zur Verfügung, um selbst entworfene Modelle zu publizieren, zu veräußern sowie um notwendige Modelle zu beziehen.

Die beiden Elemente sind als 3DS Objekte im Projekt eingefügt, nach dem Einfügen müssen diese lediglich der Umgebung skaliert und in der Farbgestaltung angepasst werden.

Abb. 4.18: Panzer, 3DS Objekt im Modell

4.5.8 Interaktives Modell

Wie bereits im Kapitel 3 beschrieben, steht für die interaktive Verarbeitung in Blender eine Game-Engine bereit. Die simpelste Form der interaktiven Verarbeitung ist die räumliche Betrachtung des Modells, ohne an einem Standpunkt gebunden zu sein. Diese Form der Interaktivität soll nachfolgend beschrieben werden.

Die Betrachtung des Modells baut auf eine Kameraführung auf, dazu wird im Projekt zunächst eine Kamera positioniert und entsprechend ausgerichtet. Diese Kameraposition ist der Startpunkt der späteren Betrachtung. Bevor die Kameraführung mittels der Logic Funktion realisiert wird, muss die positionierte Kamera aktiviert bzw. ausgewählt sein. Eine Auswahl zwischen Object Mode oder Edit Mode ist hierbei irrelevant. Nach Auswahl der Kamera erscheint im Logic Bereich im Sensors-, Controllers- und Actuators-Abschnitt die gewählte Kamera. Durch Ausführen der einzelnen Add Funktionen wird jeweils ein neuer Sensor, ein neuer Controller sowie ein neuer Actuator hinzugefügt.

Sensors, Controllers und Actuators, stellen gleichsam die Sinnesorgane, das Gehirn und die Muskeln der Echtzeit-Engine dar. Der Sensors erfühlt zum Beispiel Tastendrücke, der Controllers verarbeitet die Information eines oder mehrerer Sensors und steuert dann den Aktuator, welcher eine Aktion auslöst. [War07, S. 34] Das Zusammenspiel zwischen den Logic-Bricks (Sensor, Controller) erfolgt über die Verbindungslinien (Abbildung 4.20).

Abb. 4.19: Birke, 3DS Objekt im Projekt

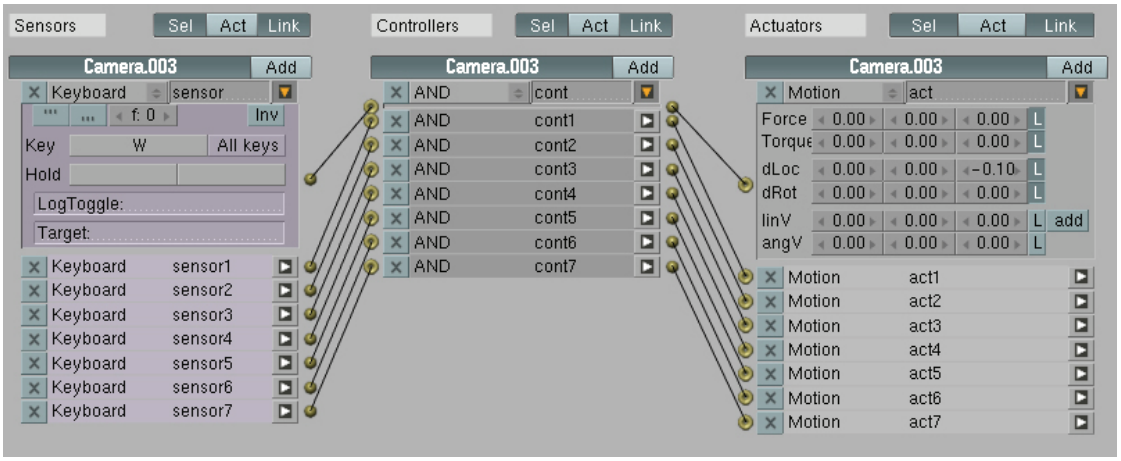

Abb. 4.20: Logic Bereich

Nach Erstellung neuer Blöcke im Sensor-, Controller- sowie Actuator-Abschnitt, werden im Sensor-Abschnitt zu jedem Block die einzelnen Bewegungstasten definiert. Im Controller-Abschnitt wird jeweils die Bedingung und abschließend im Actuator-Abschnitt jeweils der Bewegunswert. Tabelle 4.2 enthält die Tasten für die Bewegungen des Ergebnisses der Echtzeit-Engine sowie die dazugehörigen Bewegungswerte.

Bewegungsrichtung	**Taste**	**Bewegungswert mit Achsangabe**
vorwärts	W	dLoc -0.10 (Y-Achse)
rückwärts	S	dLoc +0.10 (Y-Achse)
links	A	dLoc -0.10 (X-Achse)
rechts	D	dLoc +0.10 (X-Achse)
hoch	↑	dLoc +0.10 (Z-Achse)
runter	↓	dLoc -0.10 (Z-Achse)
links dreh	→	dRot +0.03 (Z-Achse)
rechts dreh	←	dRot -0.03 (Z-Achse)

Tabelle 4.2: Überblick der Bewegungstasten sowie der Bewegungswerte im interaktiven Modell des Projektes

Nach Vergabe aller Werte kann bereits das Ergebnis über die Kameraansicht und der Taste P betrachtet werden. Wenn alles optimal eingestellt ist sollte das laufende Projekt gepackt werden, mittels File/Pack Data, somit werden alle notwendigen Daten in eine Datei verpackt und folglich ist das interaktive Ergebnis transportabel, nach Speicherung als Save Runtime. Jedoch ist die erzeugte EXE Datei alleine noch nicht lauffähig, im gleichen Verzeichnis in der sich die EXE Datei befindet werden für die Blender Version 2.45 noch folgende Bibliotheken benötigt:

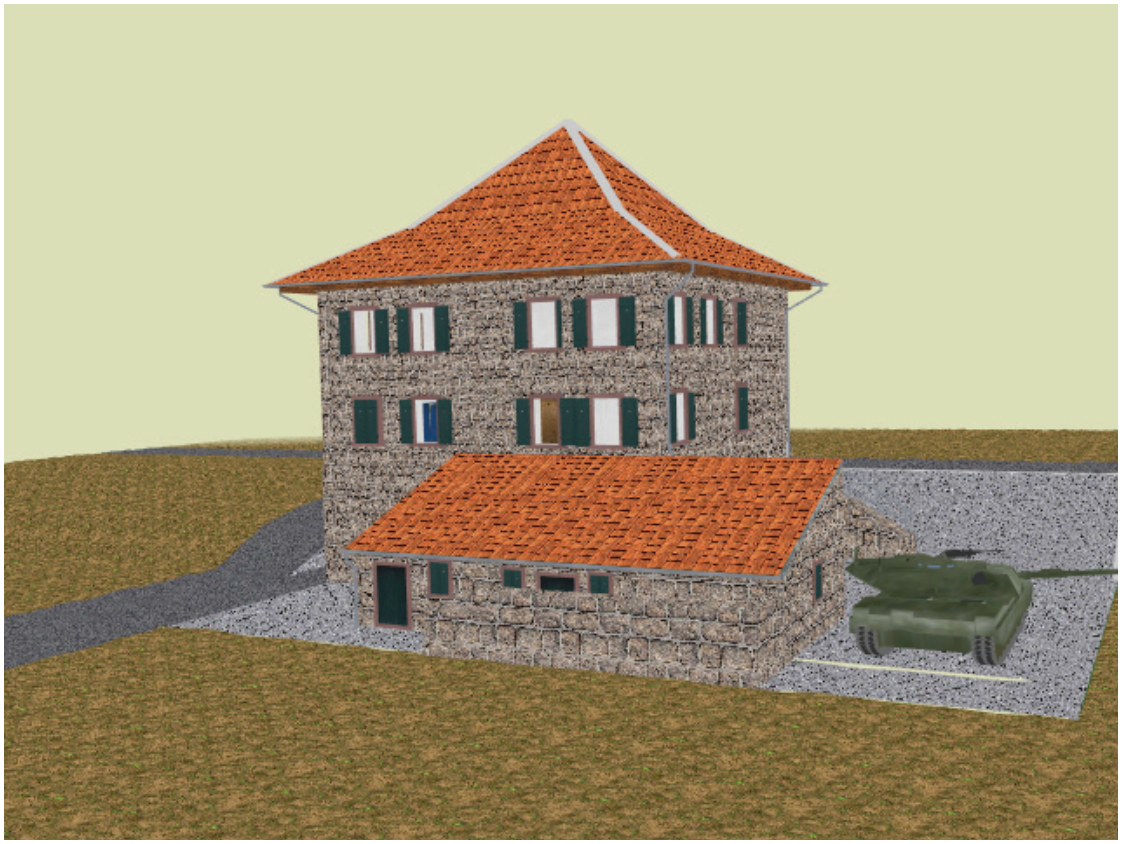

Abb. 4.21: Ansicht im interaktiven Ergebnis

- avcodec-51.dll
- avformat-51.dll
- avutil-49.dll
- gnu_gettext.dll
- iconv.dll
- libpng.dll
- libtiff.dll
- pthreadVC2.dll
- python25.dll
- SDL.dll
- zlib.dll

Die genannten Bibliotheken können ganz einfach aus dem Installationsverzeichnis von Blender heraus kopiert werden. Zusammen mit den oben genannten Bibliotheken kann die EXE Datei ausgeführt werden.

4.5.9 Schattierungsproblem im Rendering-Prozess

Die komplette Bearbeitung des Modells erfolgte in der Blender-Version 2.45. Bei verschiedenen Render-Test fielen Bereiche im Rasterbild auf, welche schwarze Schattierungen aufwiesen. Wie zum Beispiel in Abbildung 5.6, die Fensterrahmen in Parterre sowie im ersten Obergeschoss. Der Grund für diese Schattierungen ist für den Anwender unschlüssig.
Die betreffenden Flächen wurden nach dem Rendern auf ihre korrekte Lage, korrekte Texturierung sowie der Einstellung im Shading Bereich hin kontrolliert. Die genannten Kontrollen ließen jedoch keine Lösungen zum genannten Schattierungsproblem vermuten. Auffällig war jedoch, dass nach einem Neustart der genutzten Rechneranlage die Flächen nach einem erneuten Render-Prozess korrekt dargestellt wurden. Ab dem zweiten Render-Prozess stellten sich dennoch wiederholt die schattierten Flächen im Ergebnis des Render-Prozesses ein. Das genannte Problem ließe ein Konflikt im genutzten Rechner-System vermuten, da nach einem Neustart korrekte Rasterbilder angezeigt wurden. Weiterhin könnte ein Konflikt zwischen der Blender-Version sowie dem Rechnersystem vermutet werden.

Da selbst nach verschiedenen Lösungsansätzen keine Verbesserung im Rasterbild zu erkennen war, wurde auf die inzwischen veröffentlichte Blender-Version 2.46 zurück gegriffen. Das Ergebnis des Render-Prozesses mit der neueren Blender-Version wies jedoch auch das Schattierungsproblem im Rasterbild auf. Die Render-Ergebnisse mit der Blender-Version 2.46 überzeugten jedoch im Ergebnis bei Verwendung Ambient Occlusion Funktion. Bereits ab der 8. Stufe der Samples Einstellung, im Ambient Occlusion Bereich, wies das Ergebnis nicht mehr die erhöhte Körnigkeit auf, wie im Render-Ergebnis der Blender-Version 2.45 zu erkennen ist (Abbildung 4.22; 4.23). Insgesamt betrachtet erscheint das Render-Ergebnis mit Blender 2.46 stärker geglättet.

Abb. 4.22: Bild mit Blender v2.45 gerendert

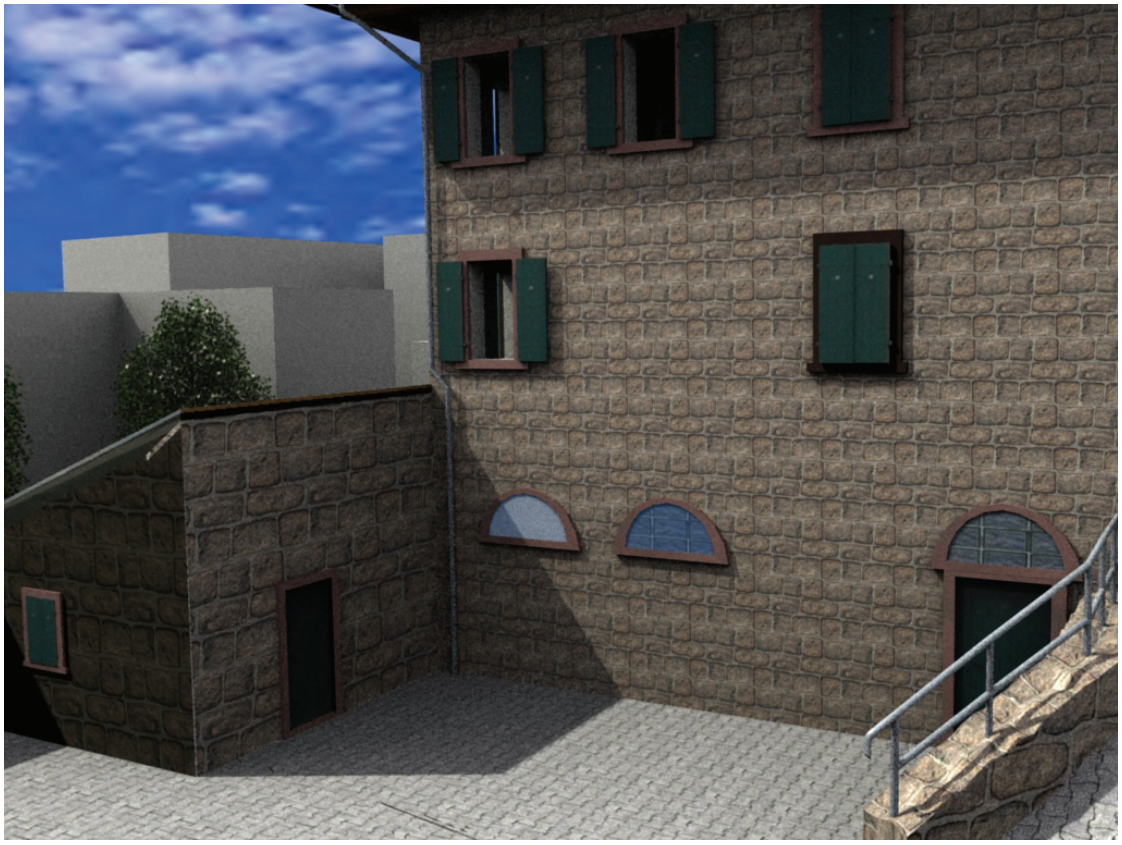

Abb. 4.23: Bild mit Blender v2.46 gerendert

4.5.10 Softbodies

Mit Softbodies hat der Blenderanwender die Möglichkeit einfache physikalische Kräfte im Projekt einfließen zu lassen, wie zum Beispiel eine im Wind wehende Fahne. Ein Softbody ist ein Objekt, dessen Vertices sich so verhalten wie reale Gegenstände. Zum Beispiel:

- lässt man einen Gegenstand los, fällt er nach unten
- weht ein Wind oder wirkt eine andere Kraft auf ein Objekt, bewegen sich die Vertices
- die Beschleunigung der Vertices hängt ab vom Gewicht, so beschleunigen schwere Objekte langsamer, als leichte
- die Bewegung wird durch Reibung verlangsamt

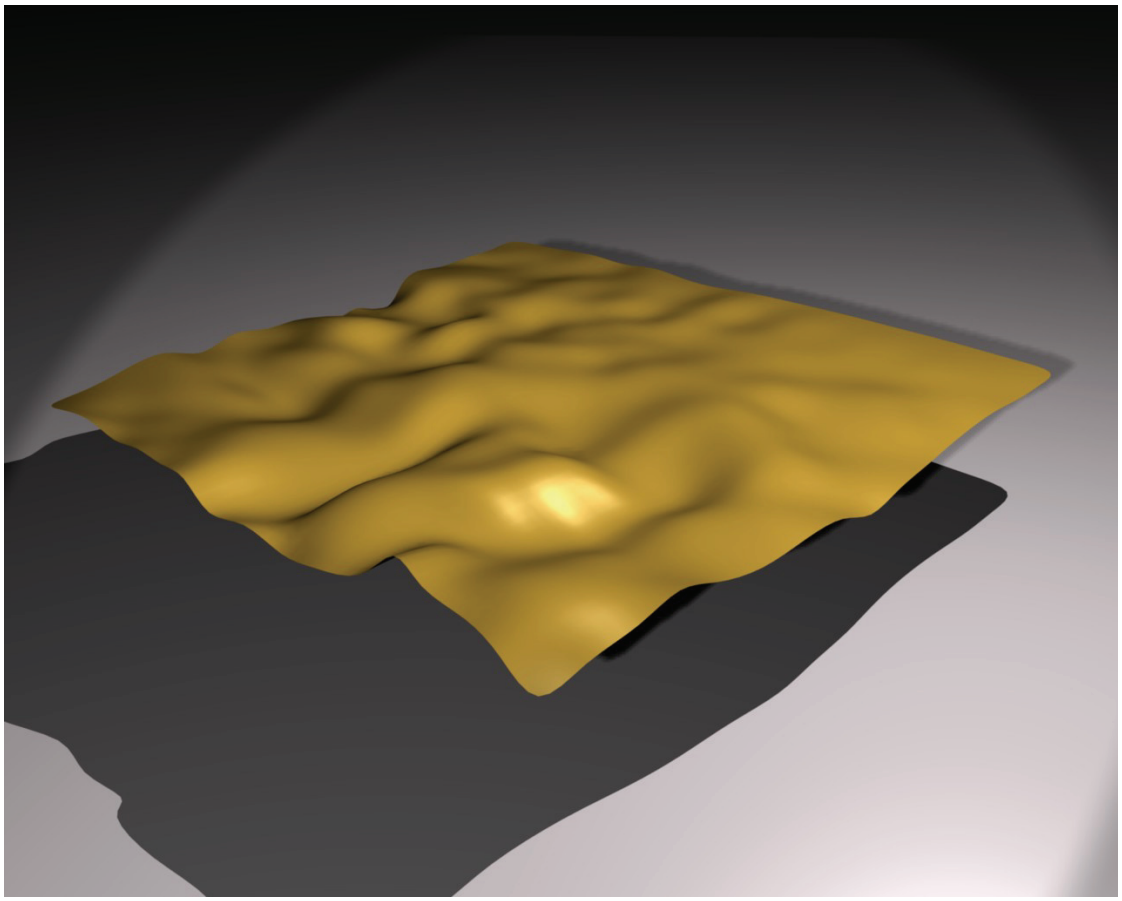

Abb. 4.24: Tuch nach 1 Sekunde freien Fall

Die Herangehensweise hierzu ist für schlichte Vorhaben sehr einfach gehalten. In den Abbildungen 4.24 und 4.25 ist zum Beispiel ein Stück Stoff erstellt worden, welches aus einer Distanz von 5 Blendereinheiten über den Boden fallen gelassen wurde. Als physikalische Kräfte wurden, die Erdanziehungskraft, seitlicher Windeinfluss sowie eine gewisse Festigkeit

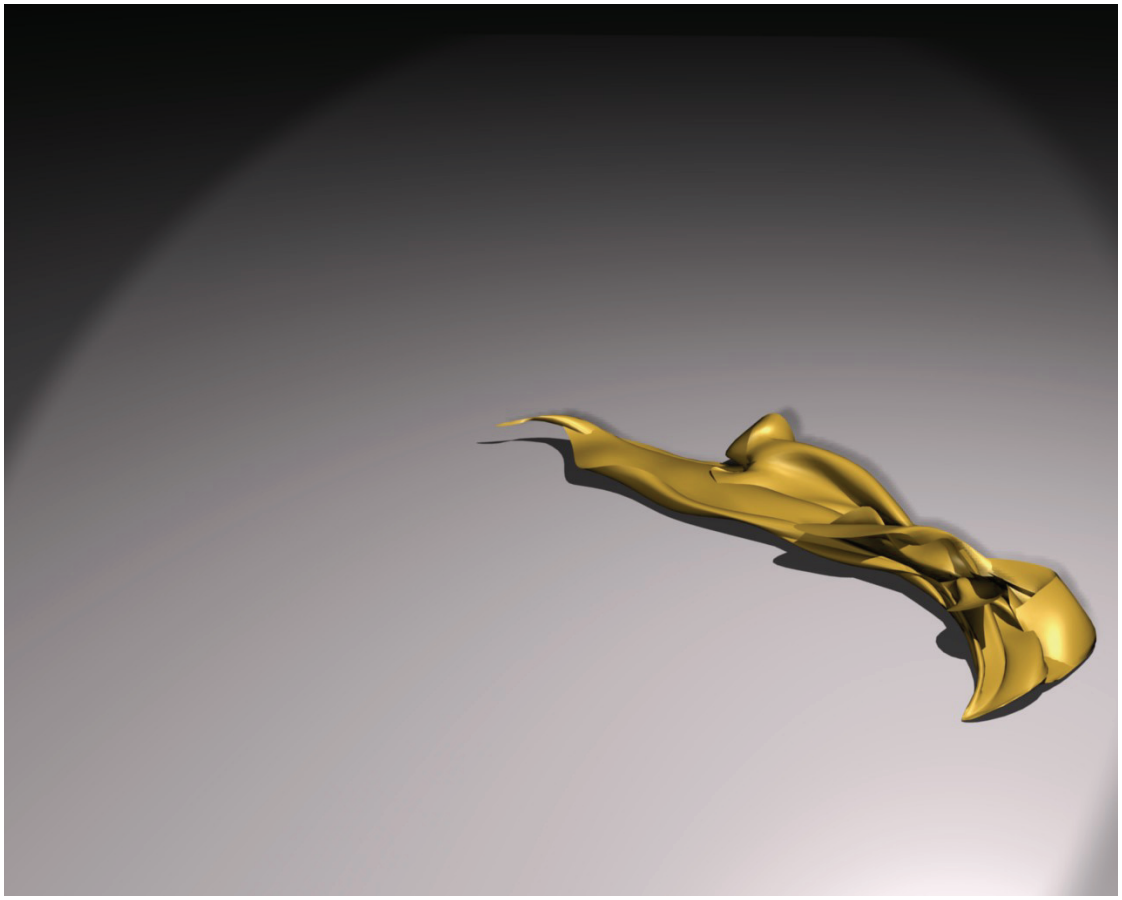

Abb. 4.25: Tuch nach 2 Sekunden freien Fall

des Stoffes eingebunden. Die notwendigen Einstellungen für die oben genannten Faktoren findet der Anwender unter Soft Body - Physics buttons bei Object. Weiterhin ist dem Untergrund im Projekt über Deflection eine Kollisionsmöglichkeit verliehen worden.
Abbildung 4.24 zeigt den Stoff 1 Sekunden nach Fallen lassen, in die negative z-Richtung, Abbildung 4.25 gibt den Stoff nach 2 Sekunden wieder, weiterhin ist in Abbildung 4.25 der Einfluss des Windes erkennbar. Die Reaktion des Stoffes mit dem Wind, dem Widerstand des Untergrundes sowie mit sich selbst wird Self Collision genannt, diese Funktion ist notwendig, da sich das Objekt sonst selbst durchdringen würde. Der Algorithmus zur Berechnung der Softbodies ist nicht für alle Objekte gleich gut geeignet, daher müssen einige Parameter immer wieder neu vergeben werden, wie zum Beispiel: der Parameter der Festigkeit [Wik08].

Grundlegend ist für die Benutzung von Softbodies ein größerer Zeitaufwand einzuplanen, für die Erstellung sowie für die Ausführung der physikalischen Funktionen. Im vorliegenden Projekt ist auf Softbodies verzichtet worden, die Verarbeitung durch Blender von Softbodies hätte schlichtweg den Zeit-, Kosten- und Nutzenaufwand überschritten.

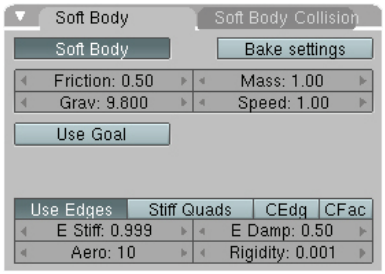

Abb. 4.26: Soft Body Einstellung für das Tuch

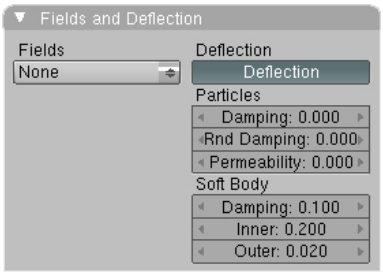

Abb. 4.27: Deflection Einstellung für den Untergrund

Für eine komplexere Anwendung von Soft Body ist ein leistungsstarkes Rechnersystem notwendig. Zu prüfen wäre hierfür die Reaktionszeit und die damit verbundene Verwendbarkeit eines Apple System.

5 Vergleichsbilder

Nachfolgend werdem 4 Vergleichsbilder aufgeführt, Abbildung 5.1 soll bei der Orientierung behilflich sein.

Abb. 5.1: Übersichtsbild mit Windrose

Abb. 5.2: Ostansicht im Modell

Abb. 5.3: Ostansicht in der Realität

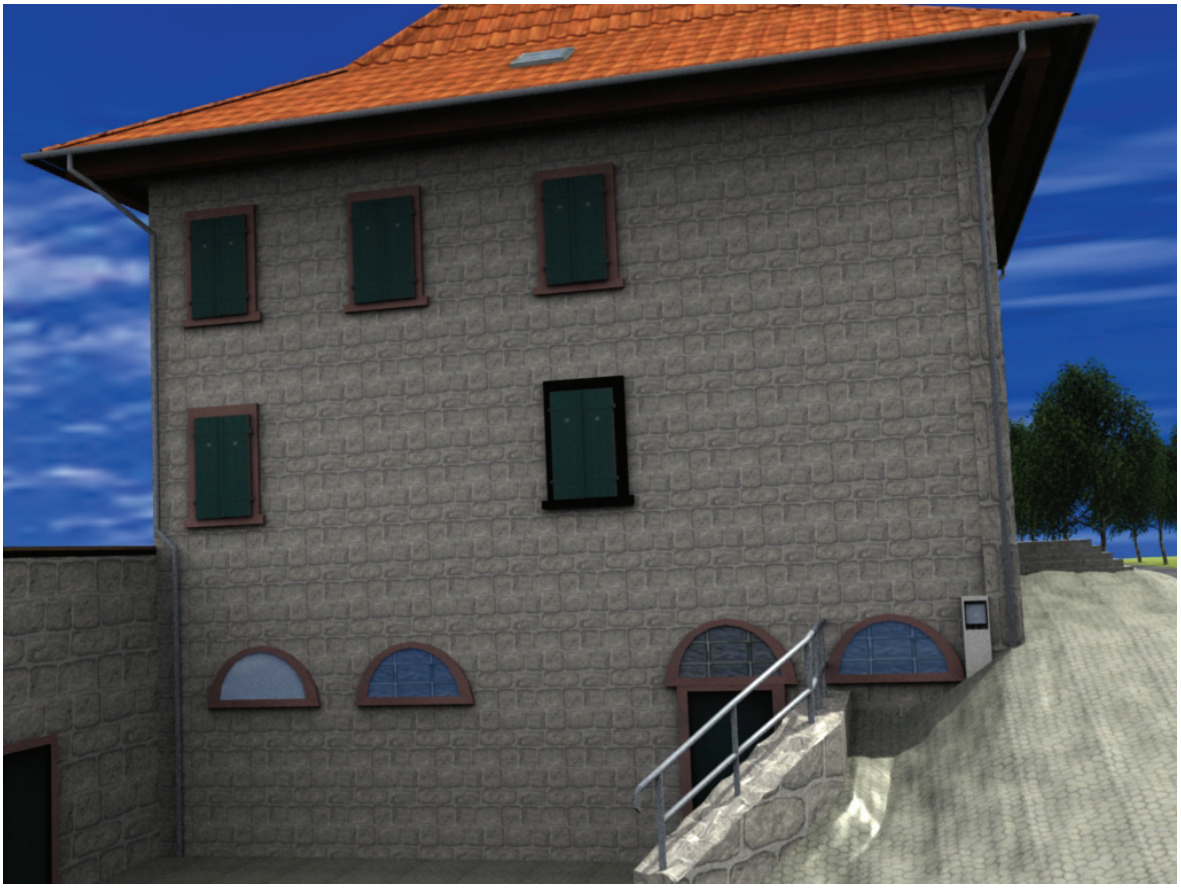

Abb. 5.4: Nordansicht im Modell

Abb. 5.5: Nordansicht in der Realität

Abb. 5.6: Südansicht im Modell

Abb. 5.7: Südansicht in der Realität

Abb. 5.8: Westansicht im Modell

Abb. 5.9: Westansicht in der Realität

6 Fazit und Ausblick

Wie bereits in der Einleitung beschrieben, wird der Bedarf von 3D Stadtmodellen in naher Zukunft steigen, insbesondere öffentliche Einrichtungen streben ein eigenes virtuelles Stadtmodell an. Aufgrund der zukünftigen Nachfrage, ist die Suche nach einem kostengünstigen Modellierungsprogramm berechtigt. Die vorliegende Arbeit beleuchtete Modellierungsarbeiten mit dem Open Source Programm Blender. Ziel war es ein Gebäude mit Blender, auf der Grundlage von Fotos und einem Grundrissplan, zu modellieren. Weiterhin beleuchtete die Arbeit verschiedene Formate und Standards, insbesondere CityGML. Hierbei wurde besonders deutlich, dass CityGML das zukünftige Austauschformat für 3D Stadtmodelle sein wird.

Abschnitt 2.1 gibt als eine der Hauptforderung die Weitergabe von semantischen Eigenschaften in einem 3D Stadtmodell basierend auf einer Datenbank vor. Die vorliegende Arbeit gibt dagegen nur semantische Informationen aus Rasterbilder wieder. Wie auch in Abschnitt 2.1 erwähnt, wäre die Anbindung einer Datenbank in Blender über Python möglich. Aufgrund der zuvor gemachten Aussage ist die Einbindung einer Datenbank in Blender notwendig, um somit eine Voraussetzung für das Format CityGML zu schaffen. Weiterhin ist die Entwicklung eines Konverters vom typischen Blender-Format in das CityGML-Format lohnenswert. Dadurch ist eine zukünftige Nutzung von Blender als Modellierungsprogramm für 3D-Stadtmodelle gesichert.

Wie zuvor im Abschnitt 3.1 aufgeführt, überzeugte das Ergebnis der Game-Engine nicht. Gründe hierfür sind in der Leistungsfähigkeit der Game-Engine von Blender zu suchen. Alternativ muss hier auf externe Lösungen zurückgegriffen werden, ein damit verbundener Kostenaufwand ist dabei unumgänglich. Das vorliegende Ergebnis der Game-Engine ist zur Situationseinweisung als befriedigend einzustufen. Entgegen der eingeschränkten Fähigkeit der Game-Engine, erwiesen sich gerenderte Bilder mit Blender als sehr anschaulich. Der Wiedererkennungswert des Modells ist als sehr hoch einzuschätzen. Bei einem Versuch mit 17 Offiziersanwärtern aus Hammelburg, wurde, nach vorheriger Einweisung der Lehrgangsteilnehmer, das modellierte Gebäude in Bonnland eindeutig vor Ort identifiziert. Weiterhin konnte jeder Offiziersanwärter im Gefechtsdienst eine grobe Geländeorientierung vornehmen, aufgrund der zuvor vermittelten Informationen. Bei einer gänzlichen Modellierung des Ortes Bonnland

ist eine vollständige Geländeorientierung im Ort Bonnland möglich. Daher ist es notwendig Bonnland als 3D Stadtmodell weiter auszubauen. Durch den Ausbau von Bonnland, steht ein Referenzmodell zur Verfügung, welches explizit durch die Bundeswehr verwendet werden kann. Anhand eines zukünftigen Bonnland-Modells ist es möglich alle Bedürfnisse und Forderungen auszuloten, welche für zukünftige Bundeswehr-Projekte notwendig sind. Denkenswert sind hier Modelle von deutschen Auslandsvertretungen, welche für Planungen bei Evakuierungsmaßnahmen von bedeutender Rolle sind. Da Bonnland im Besitz der Bundeswehr bzw. des Bundesministeriums der Verteidigung ist und der Studiengang Geoinformatik der Hochschule Neubrandenburg sich im Bereich der 3D-Modellierung immer weiter vertieft, wäre hier eine weitergehende Cooperation zwischen der Hochschule Neubrandenburg und der Bundeswehr, in Vertretung das Amt für Geoinformationswesen, empfehlenswert.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[Gmb08a] 3D Geo GmbH. Landxplorer studio. http://www.3dgeo.de/, Juli 2008.

[Gmb08b] GTA Geoinformatik GmbH. Produkte. http://www.gta-geo.de/, Juli 2008.

[Gru05] Dr. Gerhard Gröger; Dipl.-Ing. Dirk Dörschlag; Dipl.-Geogr. Marc-Oliver Löwner; Dr. Joachim Benner; Dr. Klaus Leinemann; Dipl.Geol. Rüdiger Drees; Dipl.-Ing. Ulrich Gruber. Das interoperable 3d-stadtmodell der sig 3d. Technical report, Geographisches Institut der Universität Bonn, 2005.

[Inc08] Google Inc. Geschichte. http://de.sketchup.com/?sid=249, Juni 2008.

[Kol08a] Prof. Dr. Thomas Kolbe. 3d-stadtmodellierung mit citygml. Technical report, Institut für Geodäsie und Geoinformationstechnik - Technische Universität Berlin, 2008.

[Kol08b] Prof. Dr. Thomas Kolbe. Citygml, kml und das open geospatial consortium. Technical report, Institut für Geodäsie und Geoinformationstechnik Technische Universität Berlin, 2008.

[Kol08c] Thomas H. Kolbe. About. http://www.citygml.org, Mai 2008.

[Kra04] Gröger; Kolbe; Drees; Kohlhaas; Müller; Knospe; Gruber; Krause. Das interoperable 3dstadtmodell der sig 3d der gdi nrw. Technical report, Initiative Geodaten Infrastruktur NRW, 2004.

[Kro08] Oliver Krone. Vrml vs. x3d. http://www-lehre.informatik.uni-osnabrueck.de/ okrone/DIP/node3.html, September 2008.

[Lak08] Ron Lake. Introduction to gml. http://www.w3.org/Mobile/posdep/ GMLIntroduction.html, Juni 2008.

[Moß08] Jürgen Moßgraber. Kapitel 3: Virtual reality modeling language (vrml): Interaktive 3d-dokumente. http://i31www.ira.uka.de/docs/mm+ep/03_VRML/, Juni 2008.

[Plü04] Gerhard Gröger; Thomas H. Kolbe; Lutz Plümer. *Mitteilung des Bundesamtes für Kartographie und Geodäsie, Band 31 - Zur Konsistenz bei der Visualisierung multiskaliger 3D-Stadtmodelle.* Verlag des Bundesamtes für Kartographie und Geodäsie, September 2004. ISBN: 3-89888-882-7.

[War07] Carsten Wartmann. *Das Blender-Buch*. dpunkt.verlag, Juni 2007. ISBN: 978-3-89864-466-2.

[Wik08] Wikipedia. Soft bodies. http://de.wikibooks.org/wiki/Blender_Dokumentation:_Soft_Bodies, Mai 2008.

[Zür08] RTH Zürich. Gml. http://www.gis.ethz.ch/Teaching/lecture/inf3/exercise/df/GML_Gruppe1.pdf, Juni 2008.

Danksagung

Diese Diplomarbeit widme ich meinem Vati, welcher während des Studiums verstarb, aber auch meiner Mutti sowie meinen Großeltern, da sie nicht nur mein Studium zum größten Teil finanziert haben, sondern auch ständig ein sehr großes Interesse an meiner Arbeit zeigten und mich so gut es ging unterstützten.

Zusätzlich möchte ich mich an dieser Stelle bei all denen bedanken, die mich bei der Anfertigung meiner Diplomarbeit kräftig unterstützt haben. Dies sind vor allem:

- Prof. Dr.-Ing. Hillmann (Hochschule Neubrandenburg)
- Dipl.-Ing. Winck (AGeoBw)
- Dipl.-Inf. Deistung (ZGDV)
- Oberleutnant Glosa (AGeoBw)
- Oberfeldwebel Grundmann (AGeoBw)

Vielen Dank für die hilfreichen Anregungen und die Geduld.

Weiterhin gilt mein Dank Oberst Gieske (AGeoBw), welcher sich bereits im Grundstudium für mich einsetzte, mich forderte und förderte. Vergessen möchte ich auch nicht das Personal des Geoinformationsdienstes des Einsatzführungskommandos der Bundeswehr in Potsdam, welche mir bereits im Grundstudium die Möglichkeit gaben, als Geoinformationsberater selbstständig tätig zu sein.

Vergessen möchte ich auch nicht meine Freunde, die mich zusätzlich unterstützen, in schweren Zeiten zu mir standen und dafür gesorgt haben, neben dem Studium auch das normale Leben nicht zu vergessen.