Fachhochschule Konstanz
Hochschule für Technik, Wirtschaft und
Gestaltung

Diplomarbeit

zur Erlangung des akademischen Grades
Diplom-Informatiker (FH)

Fachbereich Informatik/Wirtschaftsinformatik

Stand der Technik der Virtual Reality-Technologie im Web –

dargestellt am Beispiel einer neuartigen Fahrradschaltung

Diplomand : Braun Christian, Bargenerstr. 28, 78234 Engen-Bargen
Betreuer : Professor Dr. Ulrich Hedtstück und Prof. Dr. Wilhelm Erben
Eingereicht : Konstanz, den 02.05.2003

VORWORT

Thema der Diplomarbeit

Bei der Suche nach einem geeigneten Thema war es mein Anliegen, mich im weitesten Sinne mit Virtual Reality Technologien und deren Anwendungen zu beschäftigen.

Durch Herrn Prof. Dr. Hedtstück wurde ich dazu ermutigt, eine anfängliche Idee über ein stufenloses Fahrradschaltgetriebe interaktiv darzustellen. Durch diese Arbeit hatte ich die Möglichkeit, aus einer ursprünglich vagen Konzeption ein praxisnahes und realistisches Modell des stufenlosen Schaltgetriebes zu entwickeln. Anfängliche Überlegungen zur Realisierung stellten sich schnell als undurchführbar heraus, weshalb im Laufe der Umsetzung eine Vielzahl an Details der anfänglichen Idee angepasst und verändert werden mussten.

Durch die Entscheidung, die interaktiven Modelle auf Web3D-Technologien anzupassen, ist es nun möglich, die Funktionsweise der Schaltung einer breiten Masse von Anwendern zugänglich zu machen.

Inhalt

Nach einer Einführung in Virtual Reality Systeme im allgemeinen, folgen die Grundlagen derzeitiger Web3D-Systeme und 3D-Schnittstellen. In den Kapiteln Web3D-Formate und Web3D-Systeme werden die Grundlagen verschiedener Konzeptionen und deren Umsetzungen näher erläutert. Die Ausarbeitung schließt mit dem bereits erwähnten praktischen Teil ab.

Danksagung

An dieser Stelle möchte ich mich für die Betreuung und individuelle Unterstützung von Herr Prof. Dr. Ulrich Hedtstück, sowie Herr Prof. Dr. Wilhelm Erben, welcher sich zur Zweitkorrektur bereit erklärt hat, bedanken.

Nicht weniger wichtig war die Unterstützung, die ich während dieser Ausarbeitung durch meine Familie und viele Freunde erfahren habe. Für Ihr Verständnis und ihre Hilfsbereitschaft möchte ich mich herzlich bedanken.

INHALTSVERZEICHNIS

1 EINFÜHRUNG

1.1 Virtuelle Welten / virtuelle Realität

Es gibt keine offizielle Definition für virtuelle Realität. Das hat zur Folge, dass VR als Modewort für alles, von Computerspielen bis hin zu 3D Filmen, verwendet wird.

Definition VR

„Virtuelle Realität ist eine computererzeugte Simulation einer dreidimensionalen Umgebung, bei der der Anwender die Inhalte seiner Umgebung betrachten und manipulieren kann“. [Mats96]

Dies kann ein CAD[1]-Modell sein, wie z.B. die realistische Darstellung eines Gebäudes, das man sich näher betrachten möchte. Weitere Beispiele wären wissenschaftliche Simulationen in der Medizin und Physik.

1.1.1 Geschichte

Die Grundkonzepte der virtuellen Realität gehen bereits auf die 50er Jahre zurück. Im Jahre 1960 entwickelte Morton Heilig ein Simulationsmodell, welches den Benutzer mittels Lüfter, verschiedenen Gerüchen und Vibratoren sowie einem eingespielten Film auf eine realistische Motorradtour durch Brooklyn einlud. Das sog. Sensorama wurde aber nie kommerziell eingesetzt, da es zu kostspielig und kompliziert war.

Ivan Sutherland entwickelte 1965 das erste HMD[2]. Dieses Display zeichnete Kopfbewegungen nach und synchronisierte die Bewegungen mit dem Display. Nun war es möglich, in die virtuelle Welt einzutauchen und die Szene somit realistischer darzustellen.

In den 70er Jahren fing das NASA Ames Research Center an, die virtuelle Realität zu erforschen. Dort baute man ein einfaches HMD aus Teilen von Radio Shack[3] und verband es über ein magnetisches Tracking-System mit einem Host Computer.

---

[1] CAD = Computer Aided Design

[2] HMD = Head Mounted Device / Display

[3] Neben dem Commodore PET und dem Apple II einer der ersten Homecomputer überhaupt

1.1.2 Einsatzgebiete

Den Einsatzgebieten von VR-Systemen sind keine Grenzen gesetzt. Die Vorteile liegen auf der Hand, da seither die Herstellung aufwendiger, und somit teurer, physikalischer Modelle weitestgehend entfällt. Durch künstlich erzeugte 3D-Welten können Modelle erschaffen, verändert, getestet und erkundet werden.
Die nachfolgend vorgestellten Beispiele geben einen Ausblick auf aktuelle und zukünftige Implementierungen. Aufgrund der Fülle der Möglichkeiten handelt es sich dabei jedoch lediglich um eine Auswahl. [Cybe03]

Einsatz in der Medizin

Ärzte werden in nicht allzu ferner Zukunft mit „Röntgenblick“ in das Innere ihrer Patienten blicken können. Ermöglicht wird dies durch die Kombination von Erweiterter Realität[4] mit modernsten Bildgebenden Verfahren, wie der Computertomographie.
Der Arzt, ausgestattet mit einer Videobrille, blickt auf den Patientenkopf. Sogleich erhält er die anatomischen Strukturen lagerichtig über dem Kopf des Patienten eingeblendet. Diese Systeme befinden sich noch in der Entwicklungsphase und können für die Ausbildung von Medizinern, die Planung von Operationen und die Unterstützung bei komplizierten Eingriffen Verwendung finden.

Einsatz in der Architektur

Mit dem Besuch eines virtuellen Bauwerks ist es möglich, eine Reise vorzubereiten oder historische Bauwerke zu besichtigen, die zerstört, bzw. für Touristen nicht mehr zugänglich sind. Mit Hilfe eines virtuellen Reiseführers wird der Besucher durch die einzelnen Räume geführt und erhält, je nach Aufenthaltsort, zusätzliche Informationen zur Architektur und Baugeschichte.

Einsatz in der technischen Dokumentation

In der technischen Dokumentation lässt sich das klassische Handbuch durch einen tragbaren Computer und eine transparente Monitorbrille mit eingebauter Videokamera ersetzen. Über die Videokamera werden die

[4] siehe Glossar unter „Augmented Reality“

Kopfposition des Technikers und die im Sichtfeld befindlichen Objekte verfolgt und analysiert. Je nachdem, mit welchem Arbeitsschritt der Techniker gerade beschäftigt ist, werden über die Brille die benötigten Informationen und Arbeitsanweisungen eingeblendet. Im Gegensatz zum Handbuch erhält der Techniker so ohne suchen stets die erforderliche Information. Zudem kann er während dessen seine Arbeit fortsetzen, ohne den Blick von der Maschine abwenden zu müssen.

1.2 VR-Systeme - Technologie

Immersion

VR-Systeme bilden eine Art Schnittstelle zwischen der virtuellen Welt und dem Benutzer. Der Immersionsgrad bezeichnet hierbei den Grad, inwieweit ein Benutzer die virtuelle Welt als seine eigene Realität wahrnimmt. Im besten Fall ist der Anwender vollständig davon überzeugt, dass die ihm dargestellte virtuelle Welt vollständig real ist. Dieser Zustand ist zum heutigen Zeitpunkt nur durch zusätzliches Equipment annäherungsweise zu erreichen:

- VR-Helme (HMD)

  HMD

  sind Systeme, welche anhand von zwei kleinen Displays unmittelbar vor den Augen des Benutzers ein virtuelles Computerbild erzeugen.

  Durch den angebrachten „Head Tracker“ am HMD-System wird die aktuelle Kopfposition und Orientierung des Benutzers berechnet und auf den Displays angezeigt.

  Somit entsteht der Eindruck, Teil der virtuellen Welt zu sein, da die Ansichten je nach Kopfbewegung angepasst werden.

- Räumliche Eingabegeräte,

  Daten-handschuhe 3D- Mäuse

  wie z.B. Datenhandschuhe und 3D-Mäuse. Bei den Handschuhen handelt es sich im Prinzip um normale Handschuhe, welche über geeignete Elektronik die Lage, bzw. die Krümmung der Hand messen und an den Computer weitergeben.

Weitere Technologien, welche den Immersionsgrad und die realitätsnahe Wiedergabe erhöhen, sind zur Zeit lediglich im kommerziellen Einsatz zu finden.

So werden Piloten auf speziellen Flugzeugsimulatoren ausgebildet, welche nicht nur die aktuelle Umgebung, sondern auch äußere Einflüsse darstellen und real vermitteln. Während des „Fluges“ vermitteln computergesteuerte Hydraulikstelzen das Gefühl echten Fliegens.
VR-Systeme dieser Art spielen für den privaten Anwender jedoch eine geringe Rolle, da solche Technologien noch zu kostspielig und kompliziert sind, weshalb auch nachfolgend nicht mehr näher darauf eingegangen werden soll.
Andere VR-Systeme erreichen nur einen geringen Immersionsgrad, da der Benutzer in der Regel nur der äußere Betrachter einer virtuellen Szene ist.

1.2.1 Desktop-VR

Bei dieser Art des graphischen Systems ist die Darstellung auf einen flachen, zweidimensionalen Bildschirm beschränkt. Durch Manipulation lässt sich jedoch eine Illusion der Tiefe erzeugen. So werden z.B. Objekte umso größer dargestellt, je näher sie sich am Betrachter befinden. Objekte im Hintergrund werden durch Objekte im Vordergrund verdeckt.

Window on World

Desktop VR

Diese Art der Darstellung wird Desktop VR oder Window on World genannt, da der Benutzer von einem Blickpunkt außerhalb - wie durch ein Fenster - in die virtuelle Welt hineinschaut. Komplexe, computergraphische Verfahren, wie Perspektive, Verdeckung, Textur und Beleuchtung sorgen dafür, dass ein bestmöglicher räumlicher Eindruck der Szene entsteht.
Sämtliche Formate, Technologien und Architekturkonzepte für 3D Web-Applikationen der heutigen Zeit beruhen auf diesem Prinzip.

1.2.2 Stereo Vision

Mit stereoskopischen Sichtsystemen wie Shutterbrillen, Polarisationsbrillen, Rot-/Grünbrillen oder Stereogrammen lässt sich auch auf einem flachen Ausgabemedium, beispielsweise auf einem Computerbildschirm oder sogar auf einem einfachen Blatt Papier, eine räumliche Illusion erzeugen.

Stereo Vision
Shutterbrillen
Stereogramme

Stereo Vision simuliert das räumliche Sehen des Menschen, indem zwei verschiedene Ansichten eines räumlichen Objekts erzeugt werden, eins für jedes Auge. Die Trennung der Bildinformation für jedes Auge lässt sich auch durch Farb- und Polarisationsfilter erreichen.

Shutterbrillen arbeiten mit Flüssigkeitskristallen, die meist über einen Infrarotsender mit der Bildschirmfrequenz des Monitors synchronisiert werden. Die Bilder, die jedes Auge einzeln empfängt, werden dann durch unser visuelles Wahrnehmungssystem zu einem räumlichen Gesamtbild zusammengefügt.
Noch einfachere stereoskopische Verfahren sind Stereogramme. Bekannt geworden sind sie durch das sogenannte „Magische Auge“.

1.2.3 Immersive Systeme

Ein ideales VR-System ist in der Lage, den persönlichen Blickpunkt des Betrachters vollständig in die virtuelle Welt zu legen. Das wird zur Zeit am besten mit einem, bereits erwähnten, HMD erreicht. Das HMD liefert für jedes Auge und für jedes Ohr spezielle Signale, die in der Gesamtheit den audiovisuellen Raum simulieren.

Cave

Ein anderes, ebenfalls sehr aufwendiges immersives Verfahren ist ein so genannter Cave. Hier ist der Benutzer vollständig von Projektionswänden umgeben, auf die von aussen die synthetische Umgebung projiziert wird.

1.3 Aufbau von 3D Welten

Bei der Darstellung virtueller Welten gibt es, je nach Implementierung durch verschiedene Standards und Hersteller unterschiedliche Konzepte und Technologien. Gemeinsam sind allen 3D-Anwendungen jedoch bestimmte Grundlagen und eine spezifische Terminologie.

1.3.1 3D-Koordinatensysteme

Die Position jedes einzelnen Punktes lässt sich mit Hilfe von 3 Koordinaten (X,Y,Z) eindeutig bestimmen. Wie auch beim zweidimensionalen Koordinatensystem gibt die X-Achse die horizontale und die Y-Achse die vertikale Entfernung eines Punktes zum Ursprung an. An der Z-Achse lässt sich die räumliche „Tiefe“ messen.

Dabei gibt es zwei Möglichkeiten. Zeigt die positive Z-Achse nach vorn, so spricht man von einem rechtshändigen Koordinatensystem. Zeigt die positive Z-Achse nach hinten, so liegt ein linkshändiges Koordinatensystem vor.
Modellierungswerkzeuge, wie z.B. 3ds-Max arbeiten mit einem rechtshändigen Koordinatensystem.

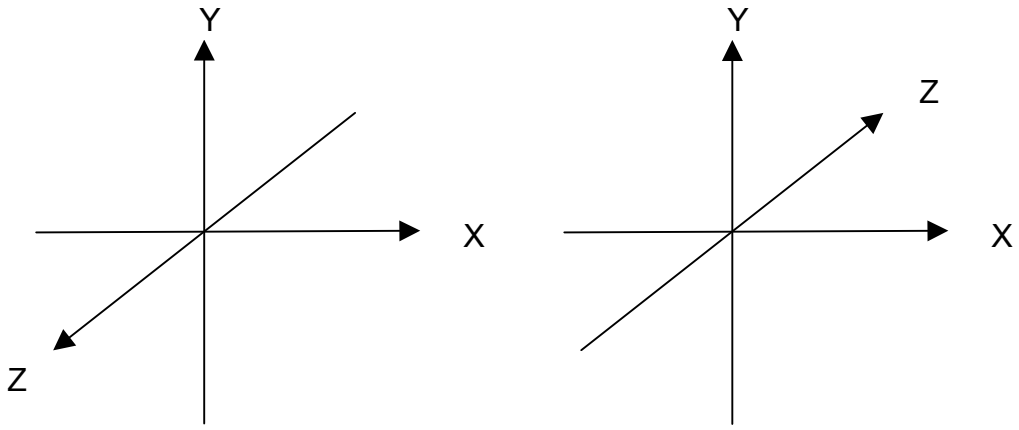


Abbildung 1-1 rechts-/linkshändiges Koordinatensystem

1.3.2 Objekte

Die Gestalt eines beliebigen Objektes in der 3D-Welt wird durch Punkte, wie in oben genannten Koordinatensystemen, beschrieben. Verbindet man die Koordinatenpunkte miteinander, so entsteht ein Drahtmodell.

Definition durch Polygon-flächen

Zur Darstellung der Objekte werden je nach Technologie entweder Polygone oder Kurven, bzw. eine Kombination aus beiden, eingesetzt.
Die einfachere Art der Darstellung erfolgt mittels Polygonen, sie ist daher am weitesten verbreitet. Polygone sind kleine plane Flächenelemente, aus denen sich beliebige, regelmäßige und unregelmäßige, Oberflächen zusammensetzen lassen. Die Orientierung jedes Polygons im Raum wird durch einen gedachten Normalvektor beschrieben, der senkrecht auf dem Flächenstück steht.
Die Anzahl der benötigten Polygone ist Abhängig von der Komplexität der Oberfläche des 3D-Objektes.
Folgende Illustration zeigt zwei identische Objekte, obwohl sie sich vom Aussehen kaum unterscheiden, weist die fein strukturierte Teekanne 10.816, die grob strukturierte jedoch lediglich 576 Polygone auf.

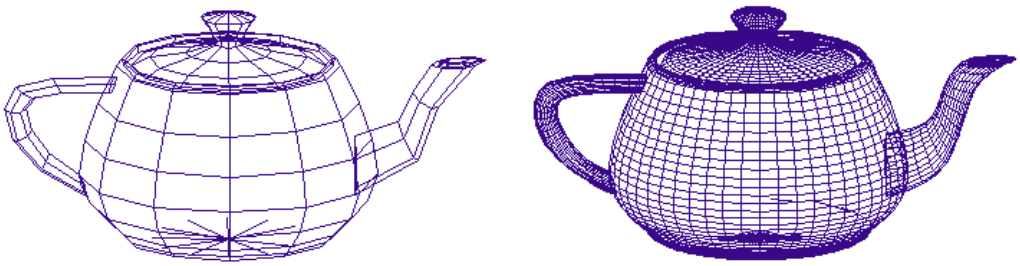


Abbildung 1-2 Zwei Objekte mit unterschiedlicher Polygonanzahl

Beim Rendering erhält dann jedes einzelne Polygon bestimmte Eigenschaften, wie z.B. Farbe und Lichtreflexe. Dadurch wird das Polygongitter verdeckt und es entsteht die Illusion eines aus einem Stück bestehenden 3D-Objekts. Je nach Darstellungsqualität/Anzahl der Polygone, können die einzelnen Polygone dennoch erkennbar sein.

1.3.3 Modellierung

Unter Modellierung versteht man die Erstellung dreidimensionaler Objekte und ihre Anordnung in einer Szene. Die einzelnen Objekte lassen sich meist in einem normalen Editor manuell erstellen.

Da die meisten Elemente über einfache Darstellungen hinaus gehen und somit einen gewissen Komplexitätsgrad erreichen, ist die manuelle Bearbeitung zu mühsam, weshalb in diesen Fällen spezielle Modellierungsprogramme verwendet werden.

Es gibt eine Vielzahl von 3D-Dateiformaten. Aus diesem Grund stellen die meisten Programme entsprechende Exportfunktionen bereit.

Bekannte Dateiformate sind z.B. DXF (Digital Exchange Format), COB (Caligari trueSpace), WRL (Virtual Modeling Language) und MAX (3D Studio Max).

Geometric Modeling
Behavioural Modeling

Bei der Erstellung und Anordnung der 3D-Objekte werden zwei Arten des Modellierens unterschieden: Geometric Modeling und Behavioural Modeling.

Geometric Modeling beschreibt den Prozess, in dem Objekte anhand ihrer Form und Gestalt definiert werden. Beim Behavioural Modeling dagegen wird festgelegt, wie die Objekt agieren und interagieren. So wird ein Objekt

zunächst über die Form und das Aussehen definiert. Verhaltensweisen (wie z.B. Bewegung oder Interaktion) werden dem Objekt später zugewiesen.
Nachfolgendes Listing zeigt eine einfache, nicht animierte, Kugel in WRL.

```
DEF Kugel Transform {
  translation -296.3 0 -390.1
  children [
    Shape {
      appearance Appearance {
      material Material {
      diffuseColor 0.7216 0.8941 0.6
        }
      }
      geometry Sphere { radius 37.36 }
    }]}
```

Listing 1-1: Einfache Kugel im WRL Format

1.3.4 Rendern

Alle 3D-Daten, gleich welchem Format, werden numerisch gespeichert. Zur Darstellung auf einem Bildschirm müssen diese Daten vom Computer in Bilder umgesetzt werden. Dieser Umsetzungsprozess wird als Rendern bezeichnet. Das Datenvolumen variiert entsprechend der Komplexität der 3D Objekte.
Typische 3D-Daten sind Informationen zu Geometrie (Form und Größe), Position und Oberflächeneigenschaften einzelner Objekte. Interaktive 3D-Anwendungen enthalten zusätzlich dazu Informationen zum Verhalten der einzelnen Objekte. Sie beschreiben, in welcher Beziehung die Elemente zueinander stehen, und wie sie mit der Anwendung außerhalb der 3D-Welt interagieren. [Wals01]
Die einzelnen 3D-Visualisierungssysteme verwenden verschiedene Methoden, um räumliche Objekte auf einem flachen Bildschirm darstellen zu können.
Bewegt man sich in einem virtuellen Raum, muss die gesamte Szene mit allen Objekten für jede einzelne Phase neu berechnet, und als einziges Bild – auch Frame – genannt auf dem Bildschirm dargestellt werden.
Aufwendige, fotorealistische Darstellungen erfordern daher enorm viel Rechenleistung, weshalb für interaktive Web 3D-Anwendungen oft Techniken

angewandt werden, die eine nicht so hohe optische Qualität liefern, dafür jedoch in Echtzeit auf die Aktionen des Anwenders reagieren können.

1.3.5 Szene

Ein virtueller, dreidimensionaler Raum, in dem ein oder mehrere Objekte angeordnet sind, wird als Szene (auch Welt oder Universum) bezeichnet. Bewegt sich der Benutzer durch die Szene, so bewirkt die Änderung des Blickpunktes gleichzeitig auch ein permanentes Rendering. Wobei es vom jeweiligen Blickwinkel des Betrachters abhängig ist, ob einzelne Objekte sichtbar (im Blickfeld des Nutzers), oder nicht mehr sichtbar (außerhalb des Blickfeldes des Nutzers) sind. Ebenfalls bewirkt eine Veränderung des Abstandes zu den Objekten eine neue Darstellung. Je nach Richtung werden die einzelnen Objekte entsprechend vergrößert oder verkleinert.

Szenegraph

Sämtliche Elemente, mit denen eine Szene beschrieben wird, sind im so genannten Szenegraph enthalten. Dies sind zunächst die in der Szene vorkommenden Objekte, jedoch auch zusätzliche Informationen über Lichtquellen, Hintergründe, Verweise auf externe Inhalte, wie z.B. Audio- und Videosequenzen.

Die Informationen der einzelnen Objekte sind innerhalb des Szenegraphen in einer hierarchischen, baumähnlichen Struktur angeordnet. Diese Struktur ist, äquivalent zur Anordnung der Objekte, in der Szene aufgebaut. Dadurch wird die dynamische Verarbeitung der enthaltenen Daten erleichtert. So ist, insbesondere bei üblichen Operationen (z.B. bei der Berechnung, welche Objekte gerade sichtbar sind), eine schnellere Anpassung der Darstellung möglich.

Auch bei der Modellierung von Verhaltensweisen ist die Struktur des Szenegraphen von Vorteil. Die Sprache, in der die eigentliche Szene erstellt wird, ermöglicht meist nur einfache Animationen bzw. Interaktionen. Für komplexere Anwendungen werden deshalb oft externe Programme eingesetzt, die über definierte Schnittstellen auf die 3D-Szene zugreifen. Durch die hierarchische Struktur des Szenegraphen wird dieser Eingriff wesentlich vereinfacht. [Wals01, Kloss97, Daes02]

2 GRUNDLAGEN WEB3D-GRAFIKEN

2.1 Client/Server- Architektur

Die Repräsentation einer Web3D-Applikation erfolgt durch einen Webserver, auf dem die Szenebeschreibung und die eventuell dazugehörigen Objektgeometrien und Medienobjekte enthalten sind. Die bidirektionale Kommunikation zwischen Client und Server erfolgt meist mittels des http-Protokolls, welches wiederum auf das TCP/IP-Protokoll aufbaut.

HTTP
TCP/IP

Auf dem Client erfolgt dann die Darstellung der Szene durch einen Player innerhalb der Webseite. Wobei der Begriff „Player“ als Interpretation der Web3D- Szene als Java Applet oder als PlugIn zu verstehen ist.

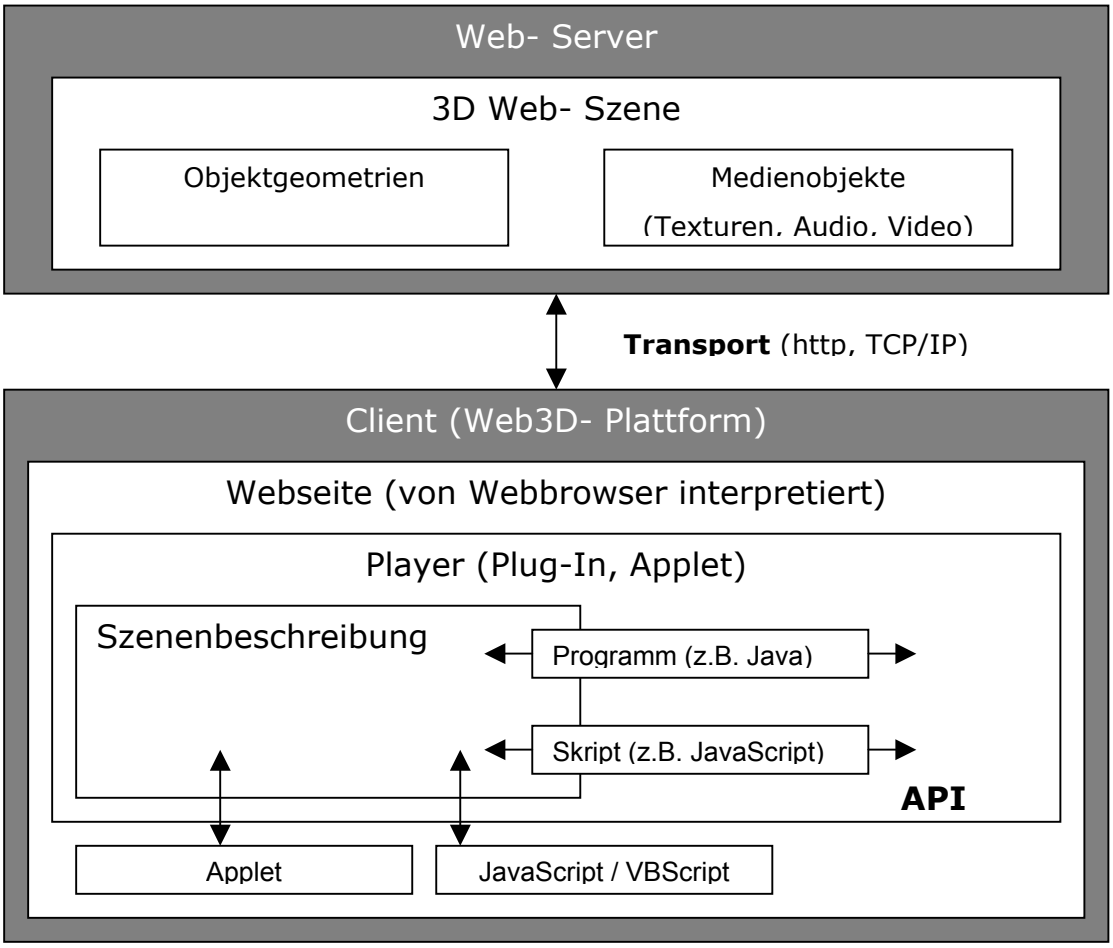


Abbildung 2-1: Client/Server- Architektur

2.2 Interaktion mit einer Web3D-Applikation

Die Interaktionsmöglichkeiten lassen sich in die Bereiche Objektmanipulation, Blickpunktmanipulation und Anwendungskontrolle unterteilen.

2.2.1 Objektmanipulation

Objekt-selection / -manipulation

Diese Art der Interaktion stellt die elementaren Möglichkeiten des Nutzers mit der 3D-Szene dar. So können einzelne Objekte z.B. ausgewählt, verschoben, skaliert, rotiert, hinzugefügt, verändert und gelöscht werden.
Hierbei können zwischen realen Verhaltensweisen zur Umwelt, wie z.B. verschieben und rotieren, sowie irrealen Verhaltensweisen zur Umwelt, wie z.B. skalieren und löschen, einzelner Objekte unterschieden werden.
In diesem Zusammenhang werden für die Objektmanipulation auch Begriffe wie Selektion und Manipulation verwendet.

2.2.2 Blickpunktmanipulation

Virtuelles Auge

Single Object 3D Animation

Sie beschreibt einerseits die Navigation durch die Szene durch Bewegung des „virtuellen Auges“ und andererseits die Veränderung der Szene gegenüber eines nichtbeweglichen Betrachters.
So werden bspw. Bewegungen innerhalb eines virtuellen Gebäudes durch die Bewegung des „virtuellen Auges“ hervorgerufen. Man kann sich das „virtuelle Auge“ auch als einzelne Kamera vorstellen, welche durch die Szene hindurch geschoben wird.
Bei Präsentationen von einzelnen Objekten wird das entsprechende Objekt meist rotiert, gezoomt oder innerhalb des Blickfeldes verschoben. Diese Vorgehensweise wird auch als Single Object 3D Animation bezeichnet.

2.2.3 Anwendungskontrolle

Sie beschreibt die Kommunikation zwischen dem Anwender und dem Teil der Anwendung, welcher nicht direkt zur Szene gehört. So gehört die indirekte Interaktion des Benutzers mit der Szene, z.B. über die Funktionalität einer Webseite, zu diesem Bereich.

Durch Anklicken eines Links kann innerhalb der Webseite eine Objekt-manipulation ausgelöst werden. [Hand97, Bowm99]

2.3 Realisierung der Animation / Interaktion

Es ist zwischen drei grundsätzlichen Ansätzen für die Realisierung von Verhaltensweisen zu unterscheiden:

2.3.1 Realisierung in der Szenenbeschreibungssprache

Unabhängig von der verwendeten Beschreibungssprache (X3D[5], VRML[6]), existieren nicht nur Konstrukte für die Definition von Objekten und deren Eigenschaften. Es besteht ferner die Möglichkeit zur Definition von einfachen Animationen, Interaktionen und Verhaltensweisen innerhalb der Beschreibungssprache.

2.3.2 Interaktion durch Skript- und Programmiersprachen

Die Interaktion der Szene kann ebenfalls durch Skriptsprachen, wie z.B. JavaScript, erfolgen. Weiterhin besteht die Möglichkeit, innerhalb der Szenenbeschreibung, auf externe Programme, bspw. realisiert in Java, zuzugreifen.

2.3.3 Externer Zugriff auf die Szene über eine API

Durch entsprechende API's[7] kann eine Kommunikation zwischen der Webseite und dem Player – z.B. durch ein Java Applet oder durch Skripte innerhalb der Webseite – erfolgen. Mittels dieser Schnittstelle kann auf die Szene zugegriffen werden.

Diese drei Ansätze für die Realisierung von Verhaltensweisen sind innerhalb der verschiedenen Web3D-Technologien unterschiedlich ausgeprägt und mächtig. [Broll98]

---

[5] X3D = Extensible 3D

[6] VRML = Virtual Reality Modeling Language

[7] API = Application Programming Interface

2.4 Problematik von Web3D Grafiken

Die zur Zeit größten Problemfelder der Web3D-Technologien sind leistungsschwache Web3D-Plattformen, eine zu geringe Bandbreite zwischen Client und Server, sowie das Fehlen spezialisierter Autorenwerkzeuge.

2.4.1 Leistungsschwache Web3D-Plattformen

Das Rendering einer Web3D-Applikation stellt hohe Anforderungen an die Web3D-Plattform, welche eine Kombination aus Soft- und Hardware ist.
Die Plattform benötigt für die Darstellung einer 3D-Szene einen Webbrowser, der entweder durch ein entsprechendes PlugIn oder durch ein automatisch geladenes Java Applet erweitert wurde.
Für einfache, kleine 3D-Szenen mit einfachen Objektgeometrien ist es meist sinnvoll, ein Java Applet zu verwenden. Die Benutzung der Klassenbibliothek Java3D setzt jedoch das Vorhandensein einer entsprechenden Laufzeitumgebung voraus, welche bei vielen Clients noch nicht installiert ist, und somit nur beschränkt eingesetzt werden kann.

API's, DirectX OpenGl

Durch die Installation und Verwendung von speziellen PlugIns, welche meist performansstarke Technologien, wie z.B. Grafik-API's, DirektX oder OpenGL verwenden, können auch komplexe 3D Szenen dargestellt werden.
Jedoch hat ein Applet gegenüber einem PlugIn den Vorteil, dass keine zusätzliche Installation notwendig ist. Benutzer sehen meist davon ab, eine Installation eines PlugIns vorzunehmen, da somit ein unerwünschter Eingriff in das System des Benutzers vermieden werden kann.

Hardware-anforderungen des Clients

Eine zu geringe Rendering-Leistung des Client Computers kann zu hohen Animations- und Interaktionsverlusten, außerdem zu einer schlechten Darstellungsqualität führen, weshalb an die Clients folgende Mindestanforderungen der Hardware gestellt werden:

- Prozessor mit mind. 500 MHz und 3Dnow! – oder ISSE[8] Erweiterung

[8] ISSE = Internet Streaming SIMD Extensions

Der Prozessor wird für Transformationen (skalieren, drehen, verschieben), für die Beleuchtung der Objekte und für die Anwendungslogik (Interaktionen, Kollisionserkennung) benutzt.

- Grafikkarte mit 3D-Beschleuniger für das Texture Mapping und Shading
- Arbeitsspeichergröße > 128 Mbyte

2.4.2 Geringe Bandbreite zwischen Client und Server

Komplexe Objektgeometrien, Animationen und Texturen, sowie umfangreiche Video- und Audiosequenzen einer 3D-Szene erfordern eine möglichst hohe Datenübertragungsrate.
Durch die steigende Verbreitung von 56k/V90-Modems, ISDN- und TDSL-Verbindungen und der vermehrten Anwendung von Modularisierungs-, Kompressions-, Streaming- und Skalierungstechniken in den Web3D-Anwendungen, verschwindet diese Problematik zunehmend.

2.4.3 Fehlen spezialisierter Autorenwerkzeuge

Cosmo Worlds
Spazz3D
Maya
3ds Max

Zur Zeit existieren viele semiprofessionelle Modellierungstools für die Erstellung von Web3D-Applikationen, wie das nicht mehr vertriebene Cosmo Worlds oder Spazz3D, deren Funktionalitäten zur Modellierung von Objekten, im Vergleich zu Maya oder 3ds max, nur ungenügend sind.
Dieses Problem versucht man dadurch zu entschärfen, dass man für professionelle Modellierungstools, welche nicht zur Verhaltensmodellierung geeignet sind, entsprechende Exporter zur Verfügung stellt.
Diese generieren dann ein bestimmtes Format, welches anschließend durch ein weiteres Modellierungsprogramm zur Verhaltensdefinition weiterverarbeitet wird. Problematisch ist hierbei aber, dass bei einem Export das Objekt oder die Szene dem Zielformat angepasst werden muss, wodurch Informationen konvertiert oder sogar weggelassen werden müssen.
Die Kombination aus einer geringen Bandbreite und einer leistungsschwachen Web3D-Plattform auf Clientseite führt zu einer hohen Startzeit, da viel Zeit für Abruf, Transport, Decodieren und Rendering der 3D-Szene in Anspruch genommen wird. [Wals01]

2.5 Lösungsansätze für Web3D Probleme

In der Vergangenheit erforderte die Übertragung von monolithischen[9] Web3D-Szenen eine hohe Bandbreite, sowie eine hohe Anforderung an die zur Verfügung stehende Rechenkapazität.

monolithische Szenen

Durch Streaming, Kompression und modulare Ansätze versucht man das Problem der geringen Bandbreite zu lösen. Auf eine leistungsschwache 3D-Plattform kann mittels Skalierung (Anpassung der Szene an die verfügbare Rechenleistung) reagiert werden.

modulare Ansätze

Durch eine schnellere und sukzessive Übertragung mithilfe von Kompression und Streaming, kann der Benutzer sofort, wenn auch mit eingeschränktem Detaillierungsgrad, mit der Szene interagieren.

2.5.1 Modulare Ansätze

Eine 3D-Szene kann in verschiedene Module aufgeteilt werden, deren Zusammengehörigkeit durch entsprechende Referenzen realisiert wird. Diese einzelnen Module, wie auch Mediendaten (Audio, Video, Texturen), werden somit erst nach Bedarf übertragen.

Stellt die 3D-Szene bspw. ein Haus dar, werden die einzelnen Module, wie Wohnzimmer, Küche usw. erst nach Betreten des Nutzers geladen. [Kloss97]

Ebenfalls zählt die clientseitige Speicherung von Daten, zur wiederholten Anwendung, zu diesen modularen Ansätzen. Diese Daten sind entweder durch eine entsprechende Bibliothek standardmäßig vorhanden oder werden nach einer Initialen Übertragung dort gespeichert.

So können bspw. unter Verwendung von bones-Animationen komplexe Charakter mit geringem Bandbreitenbedarf erstellt werden. Es wird zuerst ein initialer Charakter mit definierten Gelenkpunkten übertragen, welcher anschließend nur durch entsprechende Bewegungsinstruktionen gesteuert wird. Die Steuerung der Gliedmassen muss dabei nicht einzeln erfolgen, sonder wird in der Regel durch inverse Kinematik realisiert.

Bones Animation

Inverse Kinematik

[9] monolithisch, eine (untrennbare) Einheit bildend; aus einem Stück, zusammenhängend fugenlos hergestellt. (Brockhaus Enzyklopädie)

Abbildung 2-2: Bsp. inverse Kinematik Animation

2.5.2 Kompression der 3D-Daten

Zur Komprimierung der 3D-Daten werden meist hybride[10] Verfahren benutzt, welche verschiedene Kompressionsverfahren miteinander kombinieren. Das Hauptinteresse bei der Kompression liegt in der Komprimierung der umfangreichen Geometrie- und Texturdaten der Objekte. Die restlichen Daten werden meist durch eine verlustfreie Entropiecodierung, bspw. durch das Huffmann Verfahren, komprimiert.

2.5.2.1 Geometriekompression

Man unterscheidet zwischen dem Non-Lossy- (verlustfreie Komprimierung der Bilddaten) und dem Lossy-Verfahren. Bei letzterem geht ein Teil der Daten verloren.

Mesh Reduction

Bei der verlustbehafteten Geometriekompression „Mesh Reduction“ handelt es sich (im eigentlichen Sinne) nicht um ein Kompressionsverfahren. Die Reduktion der Daten wird hierbei durch eine Reduzierung der Anzahl an Knoten und Kanten erreicht. Dieser Vorgang stellt somit keine Komprimierung, lediglich eine Reduzierung dar. Somit kann das „Mesh Reduction“-Verfahren als eine Mögliche Vorstufe zur verlustfreien Kompression angesehen werden.

[10] [lateinisch] allgemein: aus Verschiedenem zusammengesetzt, gemischt; von zweierlei Herkunft; zwitterhaft.(Brockhaus Enzyklopädie)

Die verlustfreien Verfahren sind sich in ihrer Verfahrensweise sehr ähnlich. Der Vorgang der Komprimierung wird meist in zwei Teilbereiche gegliedert:

Connectivity Encoding

Connectivity Encoding

Hierbei werden die Kanten der Dreiecke in einer bestimmten Form kodiert. Es wird erfasst, wie die Dreieckspunkte durch die einzelnen Kanten miteinander verbunden sind.

Vertex Coordinate Compression

Vertex Coordinate Compression

Der zweite Schritt dient der Komprimierung der Koordinaten der Dreieckspunkte. Diese sind erforderlich, um mit Hilfe der Kanteninformationen das ursprüngliche Gittermodell zu rekonstruieren.

Durch die Verbindung dieser beiden Verfahren werden alle notwendigen Daten erfasst, damit beim Dekomprimieren die vollständige Struktur des Gittermodells ohne Verluste wiederhergestellt werden kann.
[Breu01, Cohe99, Velho99]

In den letzten Jahren wurden viele solcher Verfahren zur „Mesh Compression“ vorgestellt, wie z.B.:

- Geometric Compression Through Topological Surgery
  (1998, Gabriel Taubin – Jarek Rossignac) [Taub98]
- Triangle Mesh Compression
  (1998, Costa Tourma – Craig Gotsman) [Toum98]
- Real Time Compression ob triangle mesh connectivity
  (1998, Stefan Gumhold) [Gumh98]
- Edgebreaker Compressing the incidence graph of triangle meshes
  (1999, Jarek Rossignac) [Ross99]

2.5.2.2 Texturkompression

Um eine höhere Realitätsnähe zu erreichen, ohne die Komplexität der 3D-Objekte zu erhöhen, werden in Web3D-Szenen meist zusätzliche Texturen verwendet.

So kann man sich ein 3D-Landschaftsmodell vorstellen, welches durch ein Gittermodell realisiert wurde. Um dieses Modell realer wirken zu lassen, wird eine Textur, z.B. eine Abbildung von Naturlandschaften, über dieses Gitter gelegt.

JPEG, GIF MPEG, PNG Texturen

Diese eingesetzten Texturen werden meist im JPEG[11]-, GIF[12]-, MPEG[13]- oder PNG[14]- Format übertragen. Es existieren aber auch Kompressionsverfahren, die speziell für die Texturkompression entwickelt wurden.

2.5.3 Streaming

Durch Verwendung von Streaming kann eine verbesserte zeitliche Bandbreitennutzung, eine sukzessive Verbesserung der Szene und eine Erhöhung der Interaktionsgeschwindigkeit mit der Applikation erreicht werden.

In Web3D-Applikationen wird der Begriff Streaming in mehreren verschiedenen Bedeutungen verwendet:

Nachladen von Szenebestandteilen

Nachladen von Szene-bestandteilen

Das Nachladen von Szenenbestandteilen ist die am häufigsten verwendete Form des Streaming. Meist erfolgt ein initialer Ladevorgang für die Applikation und im Laufe der Zeit werden die neu benötigten Daten für die Szene nachgeladen. Solche Daten können Szenenbeschreibungen, Medienobjekte oder 3D-Objekte sein. Dieses Vorgehen wird durch Modularisierung der Szenenbestandteile ermöglicht.

---

[11] JPEG = Joint Photographic Experts Group

[12] GIF = Graphics Interchange Format

[13] MPEG = Motion Picture Experts Group

[14] PNG = Portable Network Graphics

Zeitabhängige Updates der 3D-Szene

Zeitabhängige Updates der Szene

Hierbei werden die Veränderungen in der Szene, bzw. eines Teils daraus, mittels Ersetzung durch jeweils vollständige, aktualisierte (Teil-) Szenenbeschreibungen bewirkt. Solche Updates werden meist durch den Server gesteuert. [Olbr00]

2.5.3.1 Audio- und Video Streaming

Audiovisuelle Medienströme können ebenfalls in eine Web3D-Szene eingebunden werden. Diese Mediendaten werden normalerweise durch einen virtuellen Videobildschirm, ein mit einer Videotextur versehenes Rechteck, in die 3D- Szene eingebunden. Die Wiedergabe der Audioströme erfolgt entweder mittels räumlich angeordneter Audioquellen (z.B. können Audiodaten erst nach Betretung eines virtuellen Raumes abgespielt werden) oder durch eine nicht räumliche Zuordnung der Audiodaten zur Szene.
Zur Zeit erfolgt nur die Benutzung des klassischen Audio- und Video-Streaming . [Olbr00]

2.5.4 Skalierung

Im Zusammenhang mit Web3D-Technologien ist unter Skalierung die dynamische Anpassung der Präsentationsqualität, entsprechend der verfügbaren Datenübertragungsrate und der Leistungsfähigkeit der Clientplattform, zu verstehen.
Diese Skalierung kann durch eine Anpassung der Bildrate, der Objektgeometrien und der Erscheinung der Objekte erfolgen.
Um dem menschlichen Auge jedoch eine flüssige Animation vorzuspiegeln, sollten mindestens 10 Bilder pro Sekunde wiedergegeben werden können, da die Animation sonst als ruckelnd empfunden wird.

3 3D-SCHNITTSTELLEN

OpenGL
Direct3D
QuickDraw3D

Zum rendern einer 3D-Grafik wird eine 3D-Schnittstelle, auch 3D-API genannt, benötigt. In der Vergangenheit mussten einzelne Hardware-Komponenten bei der Programmierung direkt angesteuert werden, sofern man deren Möglichkeiten voll ausschöpfen wollte. API's sind genormte Schnittstellen, die den Informationsfluss zwischen Hardware und Software ermöglichen.
Zur Zeit gibt es lediglich drei „Basis" API's, welche nicht auf die folgenden Technologien aufsetzen:

OpenGL, Direct3D und QuickDraw3D

Letztere API wurde von Apple für den Macintosh entwickelt und ist nicht weit verbreitet, weshalb sie auch im folgenden vernachlässigt werden soll.
In den nächsten Abschnitten erfolgt daher lediglich ein Einblick in die OpenGL und Direct3D Technologie, sowie deren gemeinsamer Aufsatz Java3D.

3.1 OpenGL

SGI[15], der führende Hersteller im 3D-Grafikhardwarebereich erkannte in den frühen 90er Jahren die Notwendigkeit für eine standardisierte Grafik API in der Computergrafik und Visualisierung.
Diese Schnittstelle sollte dem Anwender die Möglichkeit bieten, schnell und effektiv 3D-Grafiken zu programmieren. So muss sich der Anwender nicht mehr um die direkte Hardware-Programmierung kümmern, da durch die Standardisierung eine Portierung schnell und einfach möglich ist.
Die erste OpenGL-Version wurde im Jahre 1992 veröffentlicht. Im selben Jahr wurde die OpenGL Architecture Review Board (ARB) gegründet, welche sich unabhängig mit den OpenGL Spezifikationen beschäftigt und Erweiterungen beschließt. Das ARB besteht zur Zeit aus folgenden Mitgliedern: 3Dlabs, Apple, ATI, Dell Computer, Evans & Sutherland, Hewlett-Packard, IBM, Intel, Matrox, NVIDIA, SGI, Sun. [Arbo03, SGI02]

OpenGL Architecture Review Board (ARB)

[15] SGI = Silicon Graphics Inc.

OpenGL hat sich in den letzten Jahren zum Industriestandard und zur weitverbreitetsten 3D-API entwickelt .
Es lassen sich folgende Vorteile von OpenGL benennen:

- OpenGL weist eine hohe Stabilität durch die jahrelange Verfügbarkeit auf diversen Plattformen und die strikte Einhaltung der Spezifikationen auf
- Durch die konsistente Darstellungsqualität auf allen Plattformen ist die Zuverlässigkeit und Portierbarkeit unabhängig von Hardware und Betriebssystem gegeben.
- Durch die Erfahrungen beim Einsatz auf PCs, Workstations und High End Computern wurde eine große Skalierbarkeit erreicht.
- Logische Befehle und eine gute Strukturierung haben dafür Sorge getragen, dass die OpenGL Bibliothek einfach und intuitiv zu handhaben ist.
- OpenGL ist durch diverse Literatur und Beispielprogramme zu der am besten dokumentierten 3D-API geworden.

3.1.1 Render Modi

Immediate Mode
Retained Mode

Es wird zwischen Immediate Mode und einem Retained Mode unterschieden.
Wie die Namen bereits vermuten lassen, handelt es sich beim immediate[16] mode um einen hardwarenahen Programmiermodus, beim retained[17] mode hingegen um einen Programmiermodus, der über eine API-Schnittstelle weitestgehend vordefiniert ist. Der Immediate Mode lässt sich somit als low level modus bezeichnen. Die Schicht der Programmierschnittstelle liegt nahe an der Hardware-Ebene und erlaubt dem Programmierer einen direkten Zugriff auf spezielle Funktionen der jeweiligen Hardware-Komponente.
Der Retained-Mode (High-Level-Modus) ermöglicht es z. B. ein definiertes 3D-Objekt mit Texturen in eine Windows-Applikation zu laden. Dort kann es mit Hilfe von API-Befehlen manipuliert und bewegt werden. Die Umsetzung erfolgt

[16] Immediate = unmittelbar

[17] retained = zurückhaltend

in Echtzeit, ohne dass die programmiertechnische Struktur des Objektes bekannt sein muss. [Foru01]

3.1.2 Aufteilung der OpenGL-API

In der Standard-API sind ca. 200 Befehle enthalten, um die Grafikhardware direkt anzusprechen. Somit können Objekte, basierend auf zehn geometrischen Grundtypen, den sogenannten Primitiven (z.B. Punkte, Linien und Polygone), dargestellt werden. Diese Grundtypen werden über ihre Eckpunkte definiert. Für realitätsnahe Darstellungen wird außerdem eine Vielzahl an Rendering-Funktionen zur Verfügung gestellt.

OpenGL Utility Library (GLU)

Ein weiterer Bestandteil von OpenGL ist die Utility Library (GLU). Sie erweitert OpenGL um ca. 50 neue Funktionen und stellt einen Großteil der Modellierungs-Funktionalität zur Verfügung. So können unter anderem damit schnell und einfach quadratische Flächen und NURBS[18] Kurven generiert oder Grundkörper wie Kugel, Zylinder oder Würfel genutzt werden. [Domi97]

Die Interaktion mit dem Benutzer und die Darstellung der gerenderten Szene ist nicht mehr Bestandteil von OpenGL, da der Hauptgedanke eine Vereinfachung der Portierung auf andere Hardware ist.
Somit übernimmt ein Fenstersystem die Visualisierung. Die verschiedenen Fenstersysteme können durch entsprechende Bibliotheken wie GLX[19] (X Windows) oder WGL[20] (Microsoft Windows) angesprochen werden.

OpenGL Utility Toolkit (GLUT)

Das OpenGL Utility Toolkit (GLUT) stellt Funktionen für die Ereignisverarbeitung und für das Fenstermanagement bereit.
Es gibt eine Vielzahl von Möglichkeiten, die zur Verfügung stehenden Grundfunktionen von OpenGL zu erweitern. So stellt SGI auf verschiedene Visualisierungsbereiche ausgelegte Erweiterungen, wie OpenGL Multiple,

[18] NURBS = Non- Uniform Rational B-Spline und Surfaces

[19] GLX = Graphics Library Windows X Extension

[20] WGL = Microsoft Windows Graphics Library Extension

OpenGL Optimizer, OpenGL Performer, OpenGL Shader, OpenGL Volumizer und OpenGL Viewer, bereit. [WooM99]

Das ebenfalls von SGI entwickelte OpenInventor wird mittlerweile von TGS[21] weiterentwickelt und vertrieben und hat sich zum Standard High-Level-API entwickelt und wird vorwiegend für qualitativ hochwertige Grafik Darstellungen verwendet. [Shre00]

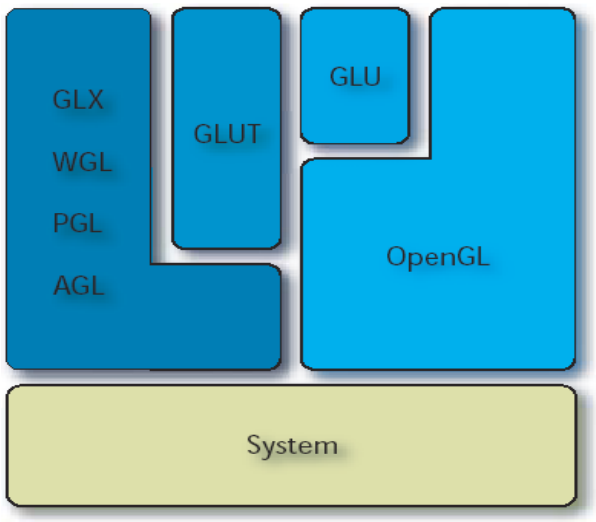


Abbildung 3-1: Schema der OpenGL-*API* [Goet02]

3.2 Direct3D / DirectX

DirectX wurde ursprünglich für Spiele auf Windows Betriebssystemen von Microsoft entwickelt. DirectX ist somit die meistgenutzte API für Windows Spielanwendungen. Eine im Umfang reduzierte Version läuft sogar auf Microsofts Windows CE Betriebssystem für Handhelds.

Viele der aktuellen professionellen Windows 3D Produkte aus den Bereichen Rendering, CAD und 3D-Visualisierung bieten die Möglichkeit auf die in DirectX enthaltene Direct3D-API aufzusetzen.

Neben Direct3D bietet DirectX noch Schnittstellen für 2D-Grafiken, Film, Sound, Eingabegeräte und Netzwerke.

[21] TGS = Template Graphics Software

Treiber für die gängigen Grafikkarten werden meist im Lieferumfang des Windows Betriebssystems zur Verfügung gestellt. [Barg98]

Mit DirectX werden die folgenden Hauptziele verfolgt:

- Mit DirectX entwickelte Multimedia Anwendungen sollen, unabhängig von der Hardware, auf allen Windows Betriebssystemen laufen
- Produkte, welche auf DirectX basieren, sollen ein Maximum an Performance und Qualität auf der jeweiligen PC-Hardware bieten

3.2.1 Render Modi

Ähnlich zu OpenGL stellt die im DirectX enthaltene Direct3D-API zwei verschiedene Modi zur Verfügung.

Immediate Mode

Retained Mode

Der Retained Mode ist eine High-Level Schnittstelle die, vergleichbar mit einer 3D-Engine, bereits eine Vielzahl an komplexen Methoden beinhaltet. Somit lassen sich schnell und mit geringem Aufwand 3D-Szenen darstellen. Erstellte Szenen können im Vollbild Modus sowie auch im Fenster Modus verwendet werden. Im Fenster Modus sind auch GUI[22] Elemente von Windows nutzbar, wobei im Vollbild Modus auf die DirectDraw-Objekte (Objekte, welche von DirectX zur Verfügung gestellt werden) zurückgegriffen wird.
Der hardwarenahe immediate Mode ist umfangreicher im Programmieraufwand als der Retained Mode, da sämtliche Funktionen, die eine 3D-Engine-Pipeline bietet, implementiert werden müssen. So müssen alle Datenstrukturen zur Verwaltung von Polygonen und Objekten selbst entwickelt werden. Der Immediate Mode ist vergleichbar mit der von OpenGL. [Enge01]

3.2.2 Aufteilung der Direct3D-API

Die Direct3D-Schnittstelle besitzt die Möglichkeit auf zwei verschiedene Schichten aufzusetzen.

Hardware Abstraction Layer (HAL)

Über den Hardware Abstraction Layer (HAL) wird die zur Verfügung stehende Hardware abstrahiert, damit Direct3D jede Grafikkarte einheitlich ansprechen kann. So muss zur Unterstützung der 3D-Funktionalität einer Grafikkarte

[22] GUI = Graphical User Interface (grafische Benutzeroberfläche)

lediglich der Hardware Abstraction Layer und nicht die gesamte Direct3D-API angepasst werden.

Hardware Emulation Layer (HEL)

Der Hardware Emulation Layer (HEL) emuliert mit Hilfe der CPU (Central Processing Unit) sämtliche Funktionen, die Direct3D zur Verfügung stellt. So ist es möglich, ohne spezielle 3D-Hardware, Direct3D-Programme zu benutzen. Die Folge ist jedoch ein enormer Performance Verlust.

Transformation Module

Herzstück der Schnittstelle ist das Transformation Module, welches zur Berechnung der Position der dreidimensionalen Objekte dient. Das Lightning Module ist für die Beleuchtung der Objekte mit verschiedenen Lichtquellen verantwortlich. Das Raster Module belegt die Objekte mit Texturen und berechnet aus allen Informationen das resultierende Bild. [Goet02]

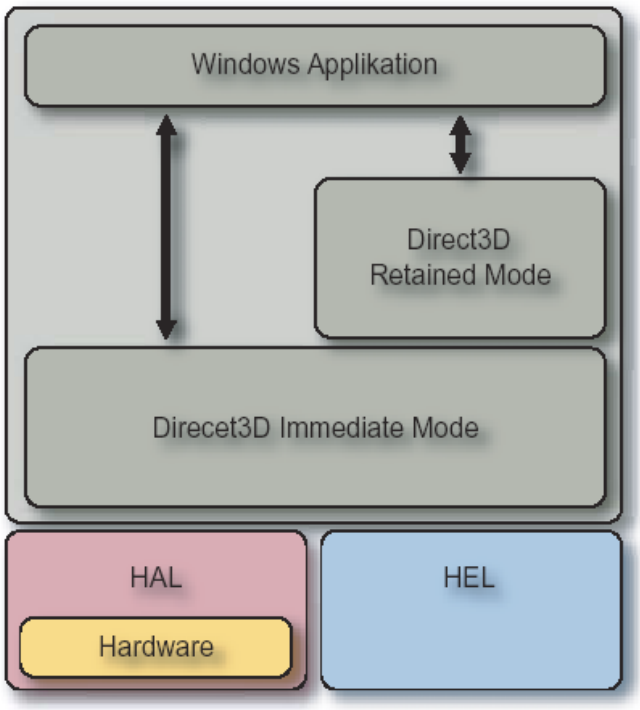


Abbildung 3-2: Schema der Direct3D-*API* [Goet02]

3.3 Java3D

Java3D bietet Entwicklern die Möglichkeit qualitativ hochwertige, skalierbare und plattformunabhängige 3D-Grafik in die Java-Technologie zu integrieren. Dazu bietet die Java3D-API eine Menge von objektorientierten Schnittstellen, die problemlos nutzbar sind. Mit diesen können 3D-Objekte und visuelle

Umgebungen erstellt, gerendert und deren Verhaltensweisen kontrolliert werden. Die Java3D-Technologie erweitert das „Write Once - Run Everywhere“ Konzept von Java um 3D-Grafik Applikationen. [Tyma98, Sowi98]
Java3D setzt in der Regel immer auf die vom Betriebssystem zur Verfügung gestellte OpenGL-API auf. Bei Windows besteht zusätzlich die Möglichkeit, Direct3D zu verwenden. Bei Mac kann auf das, eingangs erwähnte, QuickDraw3D zurückgegriffen werden. Java3D benutzt einen eigenen Software Renderer, falls kein OpenGL zur Verfügung stehen sollte.
Java3D wird hauptsächlich in Wissenschaft, Forschung und Lehre zur Simulation, Animation oder Visualisierung von Daten eingesetzt.
Es eignet sich ebenfalls gut für Internet basierende Spiel- und Lernprogramme oder 3D-Webpages.
Durch den zu VRML ähnlich aufgebauten Szenegraphen wird Java3D auch gerne für die Entwicklung von VRML-Viewern verwendet. Der zur Zeit in der Beta-Phase befindliche Open-Source Xj3D-Browser basiert ebenfalls auf Technologien von Sun. [Emme00]

3.3.1 Render Modi

Immediate Mode

Retained Mode

Compiled-Retained Mode

Ähnlich zu OpenGL und DirectX besitzt Java3D einen Immediate und Retained Mode. Zusätzlich existiert jedoch noch der Compiled-Retained Mode.
Der Immediate Mode wird in zwei weitere Render-Modi unterteilt:
Der Pure Immediate Mode unterstützt nur einen minimalisierten Szenegraphen sowie kein automatisches Rendering. Somit stehen dem Entwickler unzählige Möglichkeiten zur Verfügung, sämtliche graphischen Strukturen selbst zu entwickeln.
Der Mixed Mode ist eine Vorstufe zum Retained Mode. Im Gegensatz zum Pure Immediate Mode läuft der Renderer ununterbrochen und der Szenegraph benötigt mehr Struktur.
Mit dem Retained-Mode lässt sich der Szenegraph uneingeschränkt verändern und somit werden alle Manipulationen umgehend sichtbar. Der Retained-Mode wird am häufigsten verwendet. [Amma97, LeaR96]

Compiled Retained Mode

Um eine höhere Performance zu erlangen, kann der Compiled Retained Mode verwendet werden. In diesem Mode werden alle Teilbäume intern in einer effizienten Datenstruktur abgelegt. Hiermit wird eine höhere Ausführungsgeschwindigkeit, aber analog auch ein Verlust an Flexibilität, erreicht. [Sowi98, Java03]

3.3.2 Java3D Szenegraph

Vergleichbar zu vielen High-Level-3D-APIs besitzt Java3D einen Szenegraphen. Somit lässt sich die aufwendige Programmierung primitiver Grafik-Bibliotheken reduzieren. Wiederholende Schritte bei der Gestaltung von Grafikszenen können somit automatisiert werden.

Der Szenegraph bietet eine hierarchische, baumartige Struktur zur Verwaltung von Objekten, Gruppenbildung und Vererbung. Die Verwendung von Java3D zusammen mit dem VRML-Format ist vorteilhaft, da beide Szenegraphen ähnlich aufgebaut sind.

Virtual Universe Object

Die Wurzel des kompletten Szenegraphen bildet das Virtual Universe Object und dient zum logischen Gruppieren und Strukturieren von Teilbäumen.

Locate Object

Branch Group Nodes

Teilbäume können innerhalb einer virtuellen Szene, unter Verwendung des Locate Objekts, positioniert werden. Als Kind des Virtual Universe Objects gruppiert es die einzelnen Branch Group-Knoten. Die Positionierung der einzelnen Branch Groups kann im Bereich von Pikometern bis Lichtjahren erfolgen und ist somit sehr genau.

Es existieren zwei Arten von Node Objects. Zum einen Blätter, sie repräsentieren die tatsächlich zu rendernden Elemente, und zum anderen Gruppenknoten, die entweder auf weitere Gruppenknoten oder Blätter verweisen. Davon abgeleitet sind z.B. auch Knoten wie Transformation und Skalierung.

Scenegraph Viewing Object

Zur Kontrolle aller Anzeigeparameter und –aspekte werden die zentralen Scenegraph Viewing Objects benötigt. Es stehen vier unterschiedliche Objekte zur Definition der Parameter in einer Anzeige zur Verfügung: [Goet02, Dieh01]

- Innerhalb eines Fensters kann mittels des Canvas3D Objects gerendert werden.

- Über das Screen3D Object wird die Beschreibung des Bildschirms, in dem gerendert wird, ermittelt.
- Mit dem Physical Body Object werden alle physikalischen Aspekte, wie Kopfposition oder Position des Auges erfasst.
- An Kalibrierungsinformationen für Head tracker, Hand tracker, etc. gelangt man durch Verwendung des Physical Environment Objects.

3.3.2.1 Wiederverwendung von Teilgraphen

Es gibt zwei Möglichkeiten, Teilgraphen in Java3D wieder zu verwenden:

- Zum einen können sich verschiedene Szenegraphen einen gemeinsamen Baum teilen. Das hat zur Folge, dass das Verändern eines Knotens in einem geteilten Unterbaum sämtliche darauf verweisende Szenegraphen mit beeinflusst.
- Zum anderen kann aber auch die Knotenhierarchie eines gemeinsamen Unterbaumes dupliziert werden. Alle gemeinsamen und gleich bleibenden Daten werden geteilt. So werden Geometrien und Texturen nur einmal im Speicher gehalten.

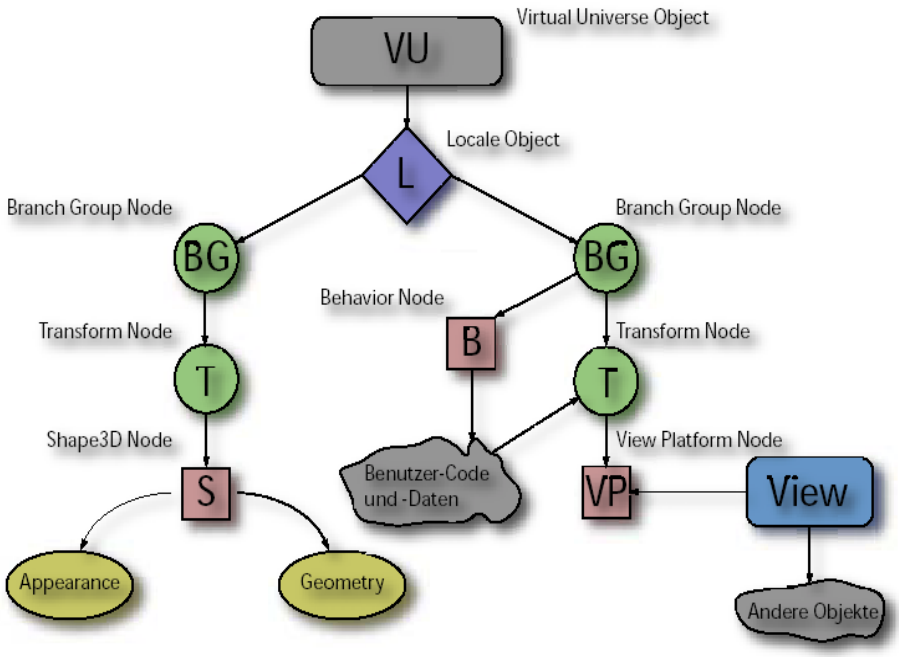


Abbildung 3-3: Beispiel Java3D Szenegraph [Goet02]

4 WEB3D FORMATE

Um die aktuellen Formate objektiv vergleichen zu können, wurden verschiedene Kriterien gewählt. So werden die gängigen Technologien unter folgenden Gesichtspunkten betrachtet: [Rukz02]

- Erstellung der 3D Welt
- Kommunikationsmöglichkeiten mit der Anwendung
- Präsentation der 3D-Objekte
- abschließende Bewertung

4.1 Vergleichskriterien der Formate

4.1.1 Erstellung

In diesem Bereich werden die Formate hinsichtlich des Formates sowie dem Autorenprozess betrachtet.

Dateiformat

Dateiformat

- Ist das Format der 3D-Dateien standardisiert oder Proprietär[23] ?
- Handelt es sich um ein deklaratives Format oder um eine imperative, bzw. objektorientierte Programmierung?
- Erfolgt die Codierung der 3D-Szene im Klartext (bspw. UTF8[24]-Format) oder in binärer Form?

Interne Struktur des Szenemodells

Interne Struktur

- Wird das Modell intern anhand eines Szenegraphen beschrieben?
- Wird die Syntax nur durch ein entsprechendes Dokument spezifiziert oder existiert zusätzlich eine direkt verfügbare XML[25]-Grammatik?

Integration von Skript- und Programmiersprachen

Integration von Programmier-sprachen

- Können Skript- und Programmiersprachen in die 3D-Szene eingebunden werden?

[23] Mit Proprietär bezeichnet man Software, Protokolle oder Systeme, die in privatem Besitz stehen und deren Verbreitung Copyrighteinschränkungen unterliegen

[24] UTF = Unicode Transformation Format

[25] XML = Extensible Markup Language

- Ist eine Manipulation der Objekte in der 3D Szene anhand von Skript- bzw. Programmiersprachen möglich?

Integration zusätzlicher Medien

- Können zusätzliche Medien, wie bspw. Video, Audio oder Bilder (Texturen) eingebunden werden?
- Sind die Medien Bestandteil der Datei, oder wird auf sie referenziert ?

Zusätzliche Medien

4.1.2 Kommunikation

Skalierung

- Erfolgt eine Anpassung der Daten an die jeweils verfügbare Bandbreite?

Skalierung
Streaming

Streaming

- Wird Streaming unterstützt?

4.1.3 Präsentation der Web3D-Szene

Player

- Erfolgt die Präsentation anhand eines PlugIns oder durch ein Applet?

PlugIns
Applet
Skalierung

Skalierung

- Erfolgt eine Anpassung der Szene/Dateien an die jeweils verfügbare Plattform des Client Rechners?

4.1.4 Bewertung

Einsatzgebiet

- Für welche Anwendungen hat sich die Technologie etabliert?

Einsatzgebiet
Nutzung
Zukunft
Fazit

Nutzungsbedingungen

- Welche Nutzungsbedingungen gibt es für die Verwendung?

Zukunftsaussichten

- Sind Nachfolgetechnologien geplant, bzw. in der Entwicklung?
- Wird das vorhandene Format ggf. weiterentwickelt?

Fazit

- Wie ist die aktuelle Akzeptanz des Formats?
- Wie ist die Dokumentation, der Support zu bewerten?

Falls ein Vergleichskriterium bei den jeweiligen Formaten nicht aufgeführt ist, kann dies bedeuten, dass dieses Kriterium oder diese Funktionalität entweder nicht existiert oder dokumentiert ist. Somit erfolgt keine sture Abarbeitung der einzelnen Vergleichskriterien, sondern sollen lediglich als Anhaltspunkt dienen.

4.2 Standardisierte Formate

ISO/IEC Standard

Um die Portabilität einer Web3D-Applikation zu erreichen, sollten sämtliche Verfahren, Formate, Schnittstellen und Protokolle einer übereinstimmenden technischen Spezifikation zugrunde liegen. Nach einem von einer Normierungsorganisation, wie der ISO/IEC[26] durchgeführten öffentlichen Verfahren wird aus einer Spezifikation eine Norm.
Bei den sogenannten Gremienstandards besteht jedoch die Gefahr, dass diese eine lange Zeit zur Fertigstellung benötigen und anschließend, in vielen Fällen, zu komplex werden. [FeyJ01]

Web3D-Konsortium

Moving Picture Experts Group (MPEG)

Die im nachfolgenden vorgestellten Kapitel werden vom Web3D-Konsortium oder von der Moving Picture Experts Group (MPEG) entwickelt.
Nach der formalen Anerkennung von VRML97 als internationaler Standard ISO/IEC 14772 konzentrierten sich die Aktivitäten der verschiedenen Arbeitsgruppen des VRML-Konsortiums zunächst auf die Weiterentwicklung von VRML97. Man erkannte jedoch schnell, dass jede Weiterentwicklung des 3D-Standards nicht losgelöst von den anderen webtechnologischen Entwicklungen erfolgen kann. Um dies zu verhindern, fasste das VRML-Konsortium 1998 den Entschluss, die Tätigkeit der Organisation nicht nur auf VRML zu beschränken, sondern auch für andere 3D-Webtechnologien zu öffnen. Um dieser Entwicklung auch namentlich Rechnung zu tragen, wurde das VRML-Konsortium in Web3D-Konsortium umbenannt.
Die Mitglieder des Konsortiums sind verschiedene Firmen, wie 3Dlabs, Microsoft Corporation, Sun Microsystems, usw., sowie universitäre Einrichtungen und Einzelpersonen.

[26] ISO/IEC = International Organisation for Standardization / International Electrotechnical Commission

Die Moving Picture Experts Group beschäftigt sich, in Zusammenarbeit mit der ISO/IEC, mit der digitalen Repräsentation für Audio- und Videoinformationen.

VRML97
X3D
MPEG-4

Zunächst erfolgt die Charakterisierung von VRML97. Diese grundlegenden Methoden zur Realisierung und Entwicklung von 3D-Grafiken im Web dienten und dienen als Vorbild für die Entwicklung anderer, aufbauender Web3D-Technologien.

Anschließend erfolgt die Vorstellung des neuen Standards, der Arbeit an X3D. Bei diesem Format handelt es sich um eine Weiterentwicklung von VRML97.

Schout3D und blaxxun3D sind die ersten Implementierungen von X3D. Da diese Formate sich aber nach 1999 von der X3D-Entwicklung abgekoppelt haben und eigene Erweiterungen entwickeln, sind sie im Grunde als proprietär anzusehen.

Zum Schluss erfolgt eine kurze Vorstellung des MPEG-4/BIFS[27] Format, welches ebenfalls auf VRML97 aufbaut und in eine komplexe Medienarchitektur integriert ist.

4.2.1 VRML97

Virtual Reality Modeling Language

Open Inventor

Analog zur Entwicklung von HTML[28] als Standard zur Präsentation von Inhalten im Web wurde 1994 mit der Entwicklung von VRML begonnen. Diese Sprache wurde zunächst, in Anlehnung an HTML, Virtual Reality Markup Language genannt. Später wurde daraus dann Virtual Reality Modeling Language.

Die erste Version war weitgehend an einen Standard der Firma Silicon Graphics Inc. (SGI) zur Beschreibung von 3D-Objekten, Open Inventor genannt, angelehnt. Nach der Freigabe des Open Inventor Formats durch Silicon Graphics präsentierten Marc Pesce und Toni Parisi bereits wenige Monate später, im Oktober 1994, auf der 2. WWW-Konferenz in Chicago den Entwurf für VRML 1.0. Die erste offizielle Version wurde im April 1995 verabschiedet.

[27] BIFS = Binary Format for Scenes

[28] HTML = Hypertext Markup Language

Diese Spezifikation stellte allerdings weder Entwickler noch Anwender zufrieden, da noch wesentliche Elemente einer VR-Umgebung, wie Interaktion und Animation, fehlten. [Kloss97, VRML97]

VRML 1.0
VRML 2.0
VRML97

Die nachfolgende Version von VRML 1.0 wurde unter VRML 2.0 vorgestellt. Nun war es erstmals möglich, Interaktion und Animation ins Web3D-Geschehen einzubauen. Unter dem Namen VRML97 wurde die Entwicklung zum Industriestandard der International Standards Organisation (ISO) veröffentlicht.

In den vergangenen Jahren hat sich VRML97 zu einem etablierten Standard entwickelt und wird von maßgeblichen Software Firmen als Export Format unterstützt. Auch sind VRML-Browser für alle gängigen Plattformen frei erhältlich. [Meis01]

4.2.1.1 Erstellung

Dateiformat

Dateiformat

Bei VRML97 handelt es sich um ein standardisiertes, deklaratives Austauschformat, dessen Zeichenkodierung im UTF-8-Format erfolgt. Es sind VRML und WRL Dateierweiterungen möglich.

Internes Modell und interne Struktur

internes Modell

interne Struktur

Das interne Modell basiert auf einem Szenegraphen, wobei die Struktur in die Bereiche Kopf, Szenegraph, Prototypen und Ereignisbehandlung unterteilt wird. Es können insgesamt 54 verschiedene Knotenarten (Nodes) integriert werden, deren Eigenschaften durch 20 verschiedenen Feldtypen (Fields) bestimmt werden können. Nachfolgend sind einige Knotentypen beispielhaft dargestellt.

Knotentypen

Geometrische Primitive	`Box, Cone, Cylinder, ElevationGrid, Extrusion, IndexedFaceSet, Text IndexedLineSet, PointSet, Sphere,`
Gruppierung	`Anchor, Billboard, Collision, Group, Inline, LOD, Switch, Transform`
Lichtquellen	`DirectionalLight, PointLight,`

	`SpotLight`
Erscheinungsbild	`Appearance, Material, MovieTexture`
Sound	`AudioClip, Sound`

Tabelle 4-1: Beispiele für Knotentypen

Ereignismodell

Ereignis-modell

Durch sogenannte Sensoren können Ereignispfade mit anderen Knoten verknüpft, und somit eine Änderung der Szene hervorgerufen werden. Jeder Knotentyp definiert Namen und Typen von Ereignissen, die er generieren oder empfangen kann. Die Verbindung zwischen Ereigniserzeuger und Ereignisempfänger erfolgt durch separate Ereignispfade (Route).

Beispiel:

```
Route NodeName.eventOutName To NodeName.eventInName
```

So genannte Interpolatoren bearbeiten Ereignisse und generieren daraus Neue. Durch die Verwendung von Ereignispfaden und Interpolatoren können einfache Animationen und Interaktionen realisiert werden.

Integration von Programmier- und Skriptsprachen

Integration von Skript- / Programmier-sprachen

Für die Integration von Skripten kann Java sowie JavaScript verwendet werden. Dadurch ist es möglich, Verhaltensweisen zu realisieren deren Komplexität nur durch die verwendete Sprache und der Schnittstelle zum Szenegraphen beschränkt wird.

Medienintegration

Medien-Integration

Sämtliche Medien, wie Bild-, Audio-, und Videodaten können in die Szene eingebunden werden. Die Mediendaten werden innerhalb des Szenegraphen referenziert. Wobei das entsprechende Medienobjekt erst nach vollständiger Übertragung der entsprechenden Dateien angezeigt werden kann.

Die Integration von Bildern (JPEG, GIF) und Videos (MPEG-1) erfolgt durch die Texturknoten `ImageTexture` und `MovieTexture`. Audioquellen werden durch den `Sound`- Knoten eingebunden.

Werkzeuge

Werkzeuge

VRML97 Szenen können in einem beliebigen Texteditor erstellt werden, es existieren jedoch auch spezielle Modellierungstools, wie das nicht mehr vertriebene Spass3D, Internet Space Builder oder Beyond 3D Extreme.
Desweiteren stellen führende Hersteller professioneller 3D-Anwendungen, wie z.B. 3ds Max oder Maya Export-Funktionen bereit, um VRML-Dateien zu generieren.

4.2.1.2 Kommunikation

Streaming

Streaming

Erst nach vollständiger Übertragung der 3D-Objekte kann die Szene angezeigt werden. Somit dauert das Laden und Öffnen der Szene relativ lange.
Es existiert nur eine eingeschränkte Methode zum dynamischen Nachladen von Szenebestandteilen. Durch den `Inline`- Knoten ist es möglich, die 3D-Szene in mehrere Bestandteile aufzuteilen, die erst bei Bedarf geladen werden.

4.2.1.3 Präsentation

Player

VRML Player PlugIns

VRML-Applets

Die Darstellung der VRML97 Szene erfolgt meist durch einen clientseitig installierten VRML Browser. Leider gibt es große Unterschiede in dem Verhalten der PluIns, was dazu führen kann, dass die Szenen unterschiedlich oder sogar gar nicht dargestellt werden.
Die verbreitetsten PlugIns sind CosmoPlayer und Cortona. Es existieren auch Applets, wie bspw. shout3D und blaxxun, für die Darstellung von VRML97-Applikationen. Sie realisieren meist aber nicht den gesamten Sprachumfang der VRML97 Spezifikation.

Skalierung

Skalierung

Durch die Verwendung des `LOD`- Knotens können Objekte in verschiedenen Detaillierungsstufen eingebunden werden. Entsprechend der aktuellen Performance können dann verschiedene `LODs`, welche vom Autor selbst erstellt und definiert wurden, vom Player ausgewählt werden. Die Verwendung von `LODs` stellt somit eine sehr eingeschränkte Form der Skalierung dar, zumal alle

LODs unabhängig voneinander gespeichert werden und somit die zu übertragende Datenmenge erheblich erhöhen.

Beispielszene VRML97

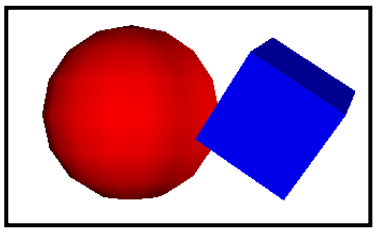


Abbildung 4-1: Beispiel VRML

```
#VRML V2.0 utf8
Transform {
   children [
            NavigationInfo { headlight FALSE
}
            DirectionalLight {
            direction 0 0 -1
}
Transform {
   translation 3 0 1
   children [
     Shape {
        geometry Sphere { radius 2.3 }
        appearance Appearance {
           material Material { diffuseColor 1 0 0 }
        }
     }
   ]
}
Transform {
   translation -2.4 .2 1
   rotation 0 1 1 .9
   children [
      Shape {
         geometry Box {}
         appearance Appearance {
            material Material { diffuseColor 0 0 1 }
         }
      }]}]}
```

Listing 4-1: Beispielszene VRML

4.2.1.4 Bewertung

Einsatzgebiet:

Einsatzgebiet

VRML wurde und wird in fast allen Gebieten eingesetzt. Es handelt sich um ein gut strukturiertes Format. Da sich die proprietären Formate meist auf die

Anwendungsbereiche Spiele, Produktpräsentationen und Charakter-animationen konzentrieren, werden diese hierfür auch bevorzugt eingesetzt.

Nutzungsbedingungen

Nutzungs-bedingungen

Die Dokumentationen des VRML97-Standards sind frei verfügbar und somit keinerlei Einschränkungen unterworfen.

Zukunft von VRML97

Zukunfts-aussichten

Es gibt viele Arbeitsgruppen, welche sich mit der Entwicklung von VRML97 beschäftigten, jedoch in Zukunft zu X3D oder MPEG-4 wechselten. X3D ist deshalb als direkte Weiterentwicklung von VRML97 anzusehen, welche letztendlich zu einem VRML200x- oder X3D-Standard führen wird.

Fazit

Fazit

Die Standardisierung, die Flexibilität, sowie die Vielzahl von Dokumentationen spricht für VRML97. Mit Hilfe dieses Formats lässt sich nahezu jede denkbare Web3D-Applikation erstellen.
VRML hat Maßstäbe für die Entwicklung der Web3D-Technologien gesetzt, aber wenig Akzeptanz im Massenmedium Internet im kommerziellen Bereich gefunden. Die Gründe hierfür sind vielschichtig und nachfolgend auszugsweise erläutert. [Maye02]

- *Hohe Ladezeiten / große Dateien*

Probleme In VRML97

Durch die Codierung im UTF-8 Zeichenformat entstehen relativ große Dateien. Die Spezifikation erlaubt lediglich eine Komprimierung mit Hilfe des LZW-Algorithmus.
Durch den Export von Modellierungsprogrammen, welche die Funktionalität von VRML97 nicht optimal nutzen, entstehen zusätzlich vergrößerte Dateien. So werden meist mehr Polygone erzeugt, als zur Darstellung notwendig sind.
Abhilfe schafft hier nur eine Nachbearbeitung der Objekte in den entsprechenden Autorensystemen.

Die folgende Illustration zeigt zwei identische Hammer, mit dem Unterschied, dass beim 2. Hammer das Polygonnetz um den nichtsichtbaren Teil verringert wurde. Die Polygonanzahl reduziert sich so von 816 auf 616 Polygone.

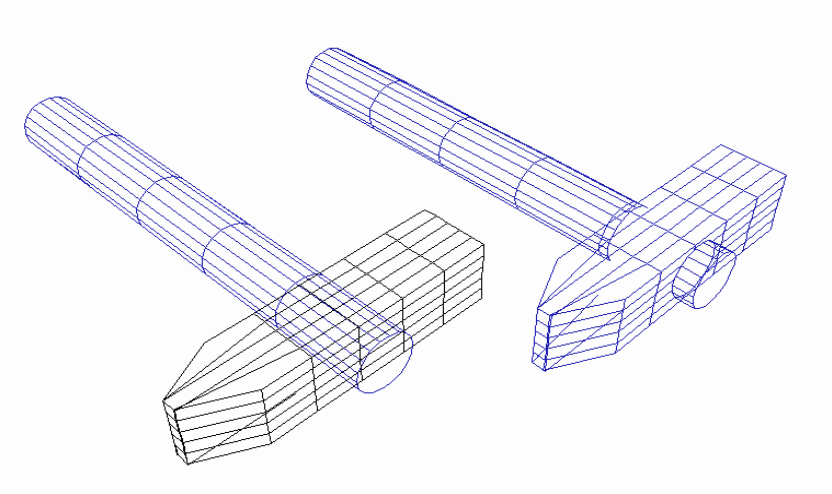


Abbildung 4-2: Reduzierung der Polygone

- *Komplexe Spezifikation*

komplexe Spezifikation

Mit VRML97 wurden die Anforderungen für verschiedenste Anwendungen in einem einzigen Standard implementiert. Dies hat den Vorteil, dass mit einer Sprache sowohl kleine 3D-Animationen, als auch dynamische Datenbank-Visualisierungen möglich sind.
Nachteile sind die daraus resultierende hohe Komplexität und mangelnde Flexibilität der Spezifikation.

- *Keine standardisierte Schnittstelle*

keine standardisierte Schnittstelle

Die Kommunikation zwischen externen Programmen und der mit VRML97 erstellten 3D-Welt erfolgt über die externe EAI[29] Schnittstelle. Diese Schnittstelle ist jedoch nicht standardisiert und entsprechend unterschiedlich implementiert. Je nach Browser/PlugIn Kombination entstehen dadurch voneinander abweichende Verhaltensweisen der 3D-Anwendungen.

[29] EAI = External Authoring Interface

Die mangelnde Konformität und die nicht standardisierte Schnittstelle widerspricht dem zugrunde gelegten Gedanken eines offenen Standards und erschwert die Erstellung von Anwendungen.

4.2.2 X3D

Aufbauend zu VRML97 arbeitet das Web3D-Konsortium seit 1998 an einer neuen Spezifikation mit dem Ziel, die Schwächen von VRML97 zu beheben und den sich ständig ändernden Anforderungen für die Darstellung von 3D-Inhalten im Internet gerecht zu werden.
Somit wurden bei der Entwicklung von X3D (Extensible 3D) folgende Schwerpunkte ins Auge gefasst: [X3D03]

- vollständige Kompatibilität zu VRML97
- modularer Aufbau
- Kompakter X3D Kern (Core X3D)
- Erweiterungsmöglichkeit zum schlanken Core Kern
- Unterstützung von XML

Der modulare Aufbau erleichtert die Einbindung künftiger Erweiterungen in die X3D-Spezifikation. Somit können anwendungsbezogene Profile hinzugefügt werden, ohne die zugrunde liegende Spezifikation als Ganzes ändern zu müssen.
Um die Erstellung integrierter Multimedia-Anwendungen zu erleichtern, nutzt X3D die Metasprache XML (Extensible Markup Language). Die von VRML97 bekannte Schreibweise wird um die XML-Syntax, sowie um ein Binärformat, erweitert. X3D ist vollständig abwärtskompatibel zu VRML97.

4.2.2.1 Erstellung

Dateiformat

Dateiformat

X3D ist ein deklaratives Austauschformat, dessen Zeichencodierung im UTF-8 Format, bzw. in binärer Form, erfolgt.

Internes Modell und interne Struktur

Internes Modell

Interne Struktur

Die Struktur einer X3D-Datei wird durch einen Szenegraphen abgebildet, der die sichtbaren und verhaltensbeschreibenden Elemente, sowie die Beziehungen untereinander beschreibt.

Elemente können ein oder mehrere Kind-Elemente enthalten. Die Kind-Elemente ihrerseits können über weitere Kind- Elemente verfügen. Jedes Kind-Element kann aber auch mehrere Eltern-Elemente haben.

Zusätzlich zu dieser hierarchischen Struktur ist die Referenzierung von einem Element auf ein anderes erlaubt. Elemente können dadurch zueinander in Beziehung gesetzt werden. Die Verbindungen zwischen ihnen werden, ähnlich zu VRML97, Routen genannt. Zusammen mit der Möglichkeit, Programme in den Szenegraphen zu integrieren, ergeben sich somit vielfältige Möglichkeiten zur Erstellung dynamischer Multimedia-Anwendungen.

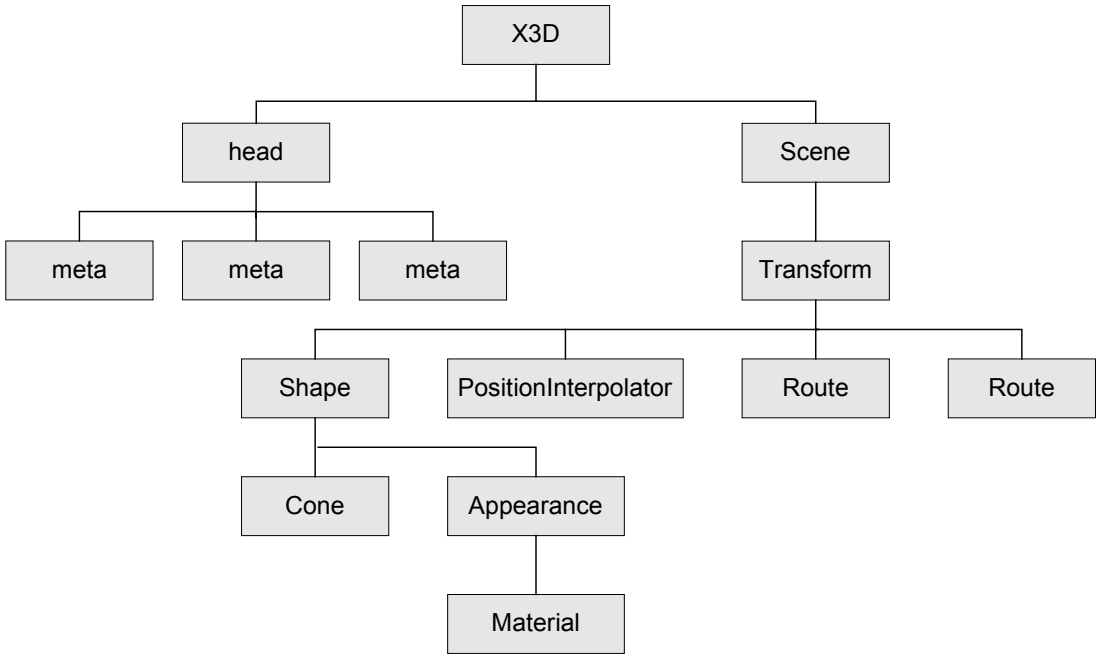


Abbildung 4-3: Beispiel eines X3D-Szenegraphen [Maye02]

Wie bereits erwähnt, stellt die Basis der X3D-Grammatik das Core-Profile dar. Es besteht aus 26 Knoten, wobei die Eigenschaften der einzelnen Knoten durch 23 verschiedene Feldtypen (Fields) bestimmt werden. Zu den 20

Feldtypen von VRML97 wurden Booleans, Vector3Double und Vector3DoubleArray hinzugefügt.

Die Basisfunktion des Core-Profils kann durch zusätzliche Profile erweitert werden. Bei der Einführung von neuen Komponenten wird ein von der OpenGL-Entwicklung bekanntes Verfahren benutzt. Es wird jeder Firma ermöglicht, eigene Komponenten zu erstellen. Je nach Akzeptanz können diese Komponenten einen Multi-Vendor-Status erhalten, der zu einer Aufnahme in eine spätere, offizielle Spezifikation führen kann.

Deshalb sind bestehende VRML97 Browser gleichzeitig auch X3D- Player, da ein entsprechendes Profil für VRML97 zur Verfügung gestellt wurde.

Es gibt momentan sechs Profiles, die in der X3D-DTD[30] aufgeführt werden:

- VRML97: Die eigentliche Umsetzung von VRML97
- GeoVRML: Geographische Visualisierung
- H-Anim: Zur Darstellung von Charakteren
- NURBS Verwendung von NURBS
- DisJavaVRML: Zur Nutzung verteilter, interaktiver Simulationen
- LatticeXvl: Zur kompakten Übertragung kompletter Geometrien

VRML97
GeoVRML
H-ANIM
NURBS
DisJavaVRML
LatticeXvl

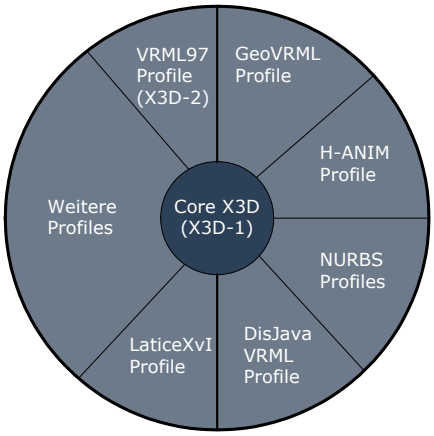


Abbildung 4-4: X3D- Architektur

Durch die X3D-Komponentenarchitektur und die niedrige Knotenanzahl des Core-Profils können kompakte, erweiterbare und leistungsfähige Player entwickelt werden.

[30] DTD = Document Type Definition

Integration von Skript- und Programmiersprachen

Skript-Programmier-sprachen Integration

Für den Zugriff auf die Szene stellt X3D umfangreiche Programmierschnittstellen zur Verfügung. Mit der Nutzung dieser Schnittstellen erhält der Entwickler die Möglichkeit Elemente zu löschen, zu erzeugen, Ereignisse auszulösen und Verbindungen zwischen Objekten zu erstellen. Elementwerte können eingelesen oder neu gesetzt werden, der Szenegraph kann durchlaufen und Browser Operationen können kontrolliert werden.
Momentan wird an dem X3D-Scene Authoring Interface (SAI) gearbeitet. Hierbei handelt es sich um eine komfortable API für den Zugriff auf den Szenegraphen. Es ist geplant, dass X3D dabei unterschiedliche Programmiersprachen und Technologien unterstützt, so z.B. Java, JavaScript, COM[31] und DOM[32].

Medienintegration

Medien-integration

Sämtliche Medien, wie Bild-, Audio-, und Videodaten können in die Szene eingebunden werden. Die Integration erfolgt zur Zeit analog zu VRML97.

Werkzeuge

Werkzeuge

Zur Erstellung eines X3D-Dokuments genügt ein einfacher Texteditor. Es existieren darüber hinaus jedoch spezielle XML/X3D-Editoren. Der zur Zeit beste Editor ist X3D-Edit, mit dem das fehlerfreie Editieren, Erstellen und Validieren von X3D möglich ist. X3D-Edit ist eine Kombination des visuellen XML-Editors Xeena von IBM und einer, auf der X3D-DTD aufbauenden Xeena-Konfigurationsdatei.
Modellierungsprogramme mit Vorschaumodus gibt es zur Zeit noch nicht. Es ist jedoch zu erwarten, dass die heute eingesetzten Modellierungsprogramme (3ds max, Maya) zusätzlich zu VRML97 auch X3D als Exportformat unterstützen. Um dieses Problem zu umgehen, existieren jedoch Konverter um eine in VRML97-Format vorliegende 3D-Welt in X3D zu konvertieren. Sowohl für die Konvertierung von VRML97 nach X3D, als auch für die Konvertierung von X3D

[31] COM = Component Object Model von Microsoft

[32] DOM = Document Object Model

nach VRML97 und in andere Dateiformate, werden die vom Web3D-Konsortium veröffentlichten XSL[33]-Stylesheets eingesetzt.
Der VRML97 to X3D Konverter des NIST (National Institute of Standards and Technology, USA) ist die derzeit einzige Anwendung zur Konvertierung von VRML97 nach X3D. Es handelt sich dabei um eine Java-Anwendung, die auf dem VRML97-Parser[34] PW des IICM (Institution for Information processing and Computer supported new Media, Österreich) basiert.
Der X3D-To-VRML97-NIST Konverter ist ebenfalls eine Java-Anwendung und basiert auf dem XML-Parser XERCES und dem XSLT Prozessor XALAN. Durch Anwendung verschiedener XSL-Stylesheets auf die X3D-Datei lassen sich mit dem NIST-Konverter nicht nur VRML97-Dateien, sondern auch beliebige Ausgabeformate erzeugen.

4.2.2.2 Kommunikation

Bei der Übertragung von X3D-Dateien werden Standard-Übertragungsprotokolle, die MIME[35]-Typ Identifikation und eine Adressierung durch URL (Uniform Resource Locator) verwendet. Eine einzige X3D-Welt kann aus mehreren einzelnen Dateien bestehen, die dann vom Browser des Anwenders zusammengesetzt werden. So lassen sich insbesondere für komplexe Szenen modulare und mehrfach verwendbare Inhalte erstellen. Dies optimiert die Ladezeit der Anfangsszene und spart weitere Übertragungszeit dadurch, dass jeweils nur die gerade benötigten Inhalte übertragen werden.
Darüber hinaus unterstützt X3D die Einbindung von Hyperlinks. Klickt der Anwender auf das mit dem Hyperlink verbundene Objekt, wird die entsprechende Adresse vom Browser geladen.
Über die klassischen Streaming-Funktionalitäten und Skalierung auf den Client-Computer konnten zu diesem Zeitpunkt noch keinerlei konkreten Hinweise gefunden werden.

---

[33] XSL = Extensible Stylesheet Language

[34] Programm, das Text oder Programmcode syntaktisch zerlegt

[35] MIME = Multipurpose Internet Mail Extensions

4.2.2.3 Präsentation

XJ3D ist bisher die einzige Laufzeitimplementierung, mit der im X3D-Format vorliegende 3D-Welten dargestellt werden können. XJ3D ist ein Open-Source-Projekt des Web3D-Konsortiums. Ziel bei der Entwicklung der Java-Anwendung ist es, die Spezifikationen von X3D, als auch die von VRML97 zu implementieren.

Die im August 2001 gegründete Browser Working Group (BWG), welche aus den Firmen blaxxun, Nexternet, OpenWorlds und ParallelGraphic besteht, ist mit der Weiterentwicklung von X3D beschäftigt. Die primären Ziele dieser Arbeitsgruppe sind die beschleunigte Entwicklung des Core-Profils, sowie die Entwicklung des Erweiterungsmechanismus.

Beispielszene X3D, analog zu VRML97 Beispiel

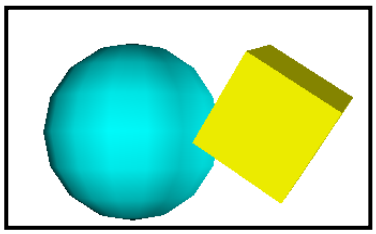


Abbildung 4-5: Beispiel X3D

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE X3D PUBLIC
[<!ENTITY % Vrml97Profile „INCLUDE">
<!ENTITY % CoreProfile „IGNORE">
<!ENTITY % X3dExtensions „IGNORE">
<X3D><Scene><Transform>
     <children>
     <NavigationInfo headlight="false"
          avatarSize=" 0.25 1.6 0.75"
          type="&#34;EXAMINE&#34;"/>
     <DirectionalLight/>
     <Transform translation="3.0 0.0 1.0"><children>
       <Shape>
          <geometry> <Sphere radius="2.3"/> </geometry>
          <appearance>
            <Appearance>
               <material>
               <Material diffuseColor="1.0 0.0 0.0"/>
               </material>
            </Appearance>
          </appearance>
       </Shape></children>
     </Transform>
     <Transform translation="-2.4 0.2 1.0"
       rotation="0.0 0.70710677 0.70710677 0.9">
      <children>
        <Shape>
          <geometry> <Box/>
          </geometry>
          <appearance>
             <Appearance>
               <material>
                 <Material diffuseColor="0.0 0.0 1.0"/>
               </material>
             </Appearance>
          </appearance>
        </Shape></children>
     </Transform></children>
</Transform>
</Scene></X3D>
```

Listing 4-2: Beispielszene X3D

4.2.2.4 Bewertung

Mit X3D soll sich jede Art von Web3D-Applikationen erstellen lassen. Dies wird insbesondere durch die zahlreichen Erweiterungsmechanismen für die Integration neuer Funktionalitäten erreicht.

Wie bereits erwähnt, befindet sich X3D noch in der Entwicklung und hat deshalb noch nicht allzu große Verbreitung gefunden.

Die einzelnen Bestandteile der X3D-Spezifikation werden von den unterschiedlichen Arbeitsgruppen des Web3D-Konsortiums erstellt. Die X3D-Arbeitsgruppe ist z.B. für die Entwicklung des X3D-Core Profils verantwortlich. Sie wird dabei unter anderem von der Browser-Arbeitsgruppe und der MPEG-Arbeitsgruppe unterstützt.

Die Browser-Arbeitsgruppe konzentriert sich auf die Entwicklung von X3D-Browsern. Um bei der Umsetzung der X3D-Spezifikation nicht auf einzelne Herstellerimplementierungen angewiesen zu sein, stellt sie den Open-Source-Browser XJ3D zur Verfügung. Darüber hinaus sorgt sie dafür, dass die Unterstützung der X3D-Spezifikation in kommerziellen Browsern möglichst konform erfolgt. Ebenfalls für die Konformität verschiedener X3D-Implementierungen zeichnet sich die Conformance-Arbeitsgruppe verantwortlich, welche entsprechende Konformitätstest durchführt.

Aufgabe der EAI-Arbeitsgruppe ist die Entwicklung einer standardisierten Schnittstelle zwischen einer X3D-Welt und einer externen Anwendung. Dabei ist ein wichtiges Ziel, die Kompatibilität zu VRML97 sicherzustellen. Darüber hinaus ist die interne Schnittstelle in das Schnittstellenkonzept einzubinden. Derzeit als verbindlich anzusehen ist die Unterstützung von Java/JavaScript. Weitere Programmiersprachen sind geplant.

Die MPEG-Arbeitsgruppe arbeitet zusammen mit der MPEG an der Integration der beiden Spezifikationen X3D und MPEG. So werden z.B. spezielle Audio- und Videoelemente in die X3D-Spezifikation aufgenommen, die konform zu denen der MPEG-Spezifikation sind. Ein weiteres Beispiel ist das künftige Binärformat für die Darstellung eines X3D- bzw. MPEG-Szenegraphen. Auch hier wird eine gemeinsame Entwicklung angestrebt.

Zusätzlich zu den aufgeführten Arbeitsgruppen gibt es weitere, die z.B. X3D-Profile für bestimmte Branchen und Anwendungsbereiche entwickeln.

Ein wesentliches Kriterium für den Erfolg von X3D wird sein, ob sich alle Hersteller auf eine gemeinsame DTD einigen können und diese dann in ihren jeweiligen Implementierungen unterstützen. Auch muss sich X3D gegen bereits am Markt etablierte 3D-Anwendungen, wie z.B. Shockwave von Macromedia, und Atmosphere von Adope Systems, durchsetzen. Von Vorteil hierbei ist, dass es derzeit einen Trend in Richtung Open-Source Produkte gibt, von dem X3D, ähnlich zu VRML97, profitieren wird.

Ebenso entscheidend für den Erfolg von X3D wird sein, inwieweit die Nachteile von VRML97 damit behoben werden.

Der wichtigste Grundstein für den Erfolg von X3D ist zweifelsohne mit der Entscheidung für die Datenbeschreibungssprache XML gelegt worden, die sich in den vergangenen Jahren zunehmend zum De-facto- Standard für Business-to-Business und Content-Management Anwendungen entwickelt hat. Durch die weite Verbreitung von XML ist die X3D-Syntax nicht, wie VRML97, nur einem speziell interessierten Autorenkreis vorbehalten. Auch können X3D-Werkzeuge schneller entwickelt werden, da sich XML-Werkzeuge ohne größere Anpassungen für verschiede X3D-Ausprägungen einsetzen lassen. Weitere Synergieeffekte entstehen dadurch, dass verschiede XML-Anwendungen über dieselben standardisierten Schnittstellen auf unterschiedlichste XML-Daten zugreifen können.

Mit XML steht erstmals nicht mehr ein Softwareprodukt mit seinen individuellen Funktionalitäten und einem Proprietären Dateiformat im Vordergrund, sondern die plattfom- und anwendungsunabhängige Ablage multimedialer Daten. Ob CAD-Daten, Musik- oder Datenbankinformationen, die Daten werden stets getrennt vom Ausgabeformat abgelegt. Aus einer einzigen Datenquelle lässt sich dann verlustfrei jedes beliebige Dateiformat erzeugen.

Abschließend lässt sich sagen, dass die beschleunigte Open-Source Entwicklung von X3D und die Integration in MPEG-4 in Zukunft eine große Bedeutung erlangen wird.

[X3D03, Web3Dkonsortium, Wals01, Behm00, Birb01, Gold99]

4.2.3 MPEG

MPEG1
MPEG2
MPEG4
MPEG7
MPEG21

Ende der 80er Jahre begann die Motion Picture Experting Group (MPEG) mit der Entwicklung eines Standards für die digitale Speicherung von Bewegtbildern. Dieser sollte die Vorläufer M-JPEG der Joint Photograpic Expert Group ersetzen. Im Laufe der Zeit wurden folgende Standards veröffentlicht:

- MPEG1 (ISO/IEC 11172)
  Standard für Video CD und MP3
- MPEG2 (ISO/IEC 13818)
  DVD und digitales Fernsehen nutzen diesen Standard
- MPEG3 wurde in MPEG2 aufgenommen
- MPEG4 (ISO/IEC 14496)
  Multimediastandard für Internet und mobile Kommunikation
- MPEG7 (ISO/IEC 15938)
  Standard für die Beschreibung und Suche von Audio- und Videoinhalten
- MPEG21 (ISO/IEC 21000-3)
  Allumfassendes Multimedia-Framework

Durch MPEG4 war es im Gegensatz zu MPEG1 und MPEG2 erstmals möglich, Szenenabläufe nicht nur zu betrachten, sondern auch unmittelbar zu beeinflussen. Aus diesem Grund soll dieser Standard im nachfolgenden näher betrachtet werden.

4.2.3.1 MPEG4 / BIFS

MPEG4 / BIFS

Im Vergleich zu seinen Vorgängern (MPEG1 und MPEG3) bietet MPEG4 nicht nur effizientere Algorithmen zur Kompression, sondern greift auch ein VRML-ähnliches Szenenmodell auf. Durch Betrachtung einzelner Objekte, anstatt des gesamten Bildes, erreicht MPEG4 viele Möglichkeiten der Skalierbarkeit.
MPEG4 ist in sechs Teile untergliedert:
Systems, Visual, Audio, Conformance Testing, Reference Software sowie Delivery Multimedia Integration Framework (DMIF).

Desweiteren wird zwischen Visual-, Audio-, Graphics-, Scene Graph-, MPEG-J- und Objekt Descriptor- Profilen unterschieden. Diese einzelnen Profile sind wiederum in verschiedene Levels unterteilt.
Alle nachfolgenden Informationen beziehen sich auf die Möglichkeiten der Web3D-Funktionalitäten in MPEG4. Informationen und Funktionalitäten, die sich nicht auf den 3D-Bereich von MPEG4 beziehen, werden hierbei vernachlässigt.

4.2.3.2 Erstellung

Dateiformat

Binary Format for Scene description

Das Binärformat zur Szenenbeschreibung BIFS (Binary Format for Scene description) basiert auf einer Erweiterung von VRML und erlaubt die hierarchische Gruppierung von audiovisuellen Objekten und deren Positionierung in einem zwei- oder dreidimensionalen Raum.
Das zeitliche Verhalten der Animation lässt sich mit BIFS manipulieren und umfasst z.B. das Ändern von Positionen, Farben und Lautstärken einzelner, oder ganzer Gruppen von Objekten. BIFS erlaubt die Darstellung von Linien bis hin zu komplexen Polygonzügen. Somit unterstützt BIFS die nahtlose Integration von natürlichen und audiovisuellen Effekten.
Die Codierung erfolgt im UTF-8 Format, welches aber serverseitig vor der Auslieferung in ein Binärformat umgewandelt wird.

Internes Modell und interne Struktur

Interne Struktur

BIFS Szenegraph

Der BIFS Szenegraph besteht aus Knoten, Feldern und Ereignissen. MPEG4 Szenen werden aus einer Sammlung von Knoten zu einem azyklischen Graph zusammengesetzt. Ähnlich zu VRML enthält MPEG4 Knoten zur Interpolation, Sensoren und Routen. Im Unterschied zu VRML definiert MPEG4 auch neue Knoten für zusätzliche Funktionalitäten. So umfasst MPEG4 grob 100 Knoten, im Vergleich zu 54 Knoten in VRML, welche in 20 Kategorien fallen. Die zusätzlichen Knoten unterstützen neue Funktionalitäten für 2D, Audio und Streaming.
Die Struktur eines Szenegraphen in MPEG4 BIFS wird in nachfolgender Abbildung dargestellt. Er enthält ein natürliches audiovisuelles Objekt, eine

animierte, synthetische Darstellerin und weitere synthetische graphische Elemente:

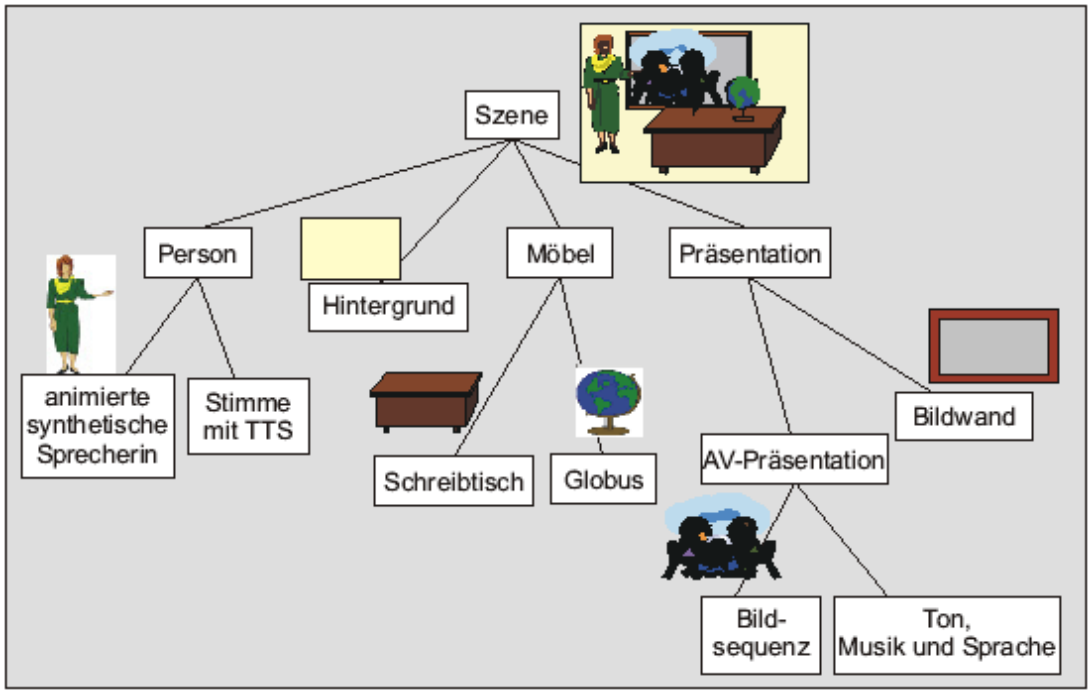


Abbildung 4-6: Bsp. für einen *BIFS*-Szenengraphen in MPEG4 [Kaup01]

Szenenstruktur

Szenen-struktur

Es existieren folgende Arten von Knoten:

- Grouping Nodes: Bauen die Struktur des Szenegraphen auf
- Children Nodes: Ermöglichen die Gruppierung von Multimedia-Objekten
- Bindable Children Nodes: Sind Knoten, die jeweils nur einen Unterknoten aktivieren können
- Interpolator Nodes: Sind Knoten, mit denen man Sequenzen generieren kann
- Sensor Nodes: Nehmen Benutzer und Umgebungsbedingungen wahr

Knoten und Felder

Knoten und Felder

Die BIFS-Knoten bestehen, ähnlich zu VRML, aus Feldern und Grundfeldtypen: Bolean, Boolean, ganze Zahl, floating Point, 2 und 3-dimensionale Vektoren, normale Vektoren, Winkel, Farben, URLs, Strings und Images. Die Knotenfelder können Events empfangen, senden oder auch beides zusammen.
Die Animation einer Szene kann durch Routen hergestellt werden. Dabei gibt es einen Knoten der Daten sendet, einen anderen Knoten der diese Daten empfängt. Es können aber nur Knoten Daten empfangen, welche ein EventIn-Feld besitzen und Daten senden können nur Knoten mit einem InputIn-Feld:

```
ROUTE <OutNodeID> outFieldName TO <InNodeID>inFieldName
```

Face Definition Parameter

Body Definition Parameter

Charakterdarstellungen, wie z.B. Gesichts- und Köpermodelle können durch die Face Definition Parameter (FDP) und Body Definition Parameter (BDP) definiert werden. Somit stehen für die Modellierung und Beschreibung des Gesichts 66 zusätzliche Parameter, für die Modellierung des Körpers 260 Parameter zur Verfügung.
Ähnlich zu den Charakter Definitions Parameter, existieren sogenannte Animationsparameter, um den Modellen realistische Bewegungsabläufe zu garantieren. Für die Animation des Gesichts stehen Facial Animation Parameter (FAP) und für die Animation des Körpers Body Animation Paramter (BAP) zur Verfügung.

Integration von Skript- und Programmiersprachen

Skript-Programmier-sprachen Integration

Gegenwärtig wird lediglich JavaScript unterstützt. Wobei mithilfe von JavaScript in die Szene direkt eingegriffen werden kann. Durch Route Knoten können so einzelne Animationen abgespielt und manipuliert werden. Außerdem können einzelne Werte vom Benutzer direkt abgefragt werden, um z.B. Werte einzelner Knoten zu setzen. So kann die Position, die Skalierung, usw. einzelner Objekte beeinflusst werden.

Medienintegration

Medien-integration

In MPEG4 können 2- und 3 dimensionale Objekte untereinander „gemischt“ werden. Im Gegensatz zu VRML kann ein 2 dimensionales Objekt eine lokale Ebene der 3D-Szene bilden. So können durch die Knoten Layer2D und Layer3D Ebenen definiert werden, in denen 2D- und 3D-Unterszenen eingebunden werden. Dadurch kann bspw. ein 3 dimensionales Logo in eine 2 dimensionale Video-Sequenz eingebunden werden. Mit Hilfe der Knoten `CompositeTexture2D` und `CompositeTexture3D` ist es ebenfalls möglich 2D- und 3D-Szenen als Texturen zu realisieren.

Werkzeuge

MPEG- 4 DevStudio MDS

MPEG4 Dateien können mit einem normalen Texteditor erzeugt werden, da die Umwandlung ins Binärformat erst später erfolgt.

MPEG-4 Toolbox

Ein Beispiel für ein einfaches Autorensystem wurde von der französischen Hochschule École Nationale Supérieur Telecommunications (ENST-Télécom Paris, France) erstellt. Das MPEG4 DevStudio MDS ist eine betriebssystemunabhängige Implementation in Java und erfordert fundierte Kenntnisse der Elemente mittels BIFS.

Ein weiterer visueller MPEG4 Editor ist die MPEG4 Toolbox, welche aber im praktischen Einsatz kaum Verwendung findet.

4.2.3.3 Kommunikation

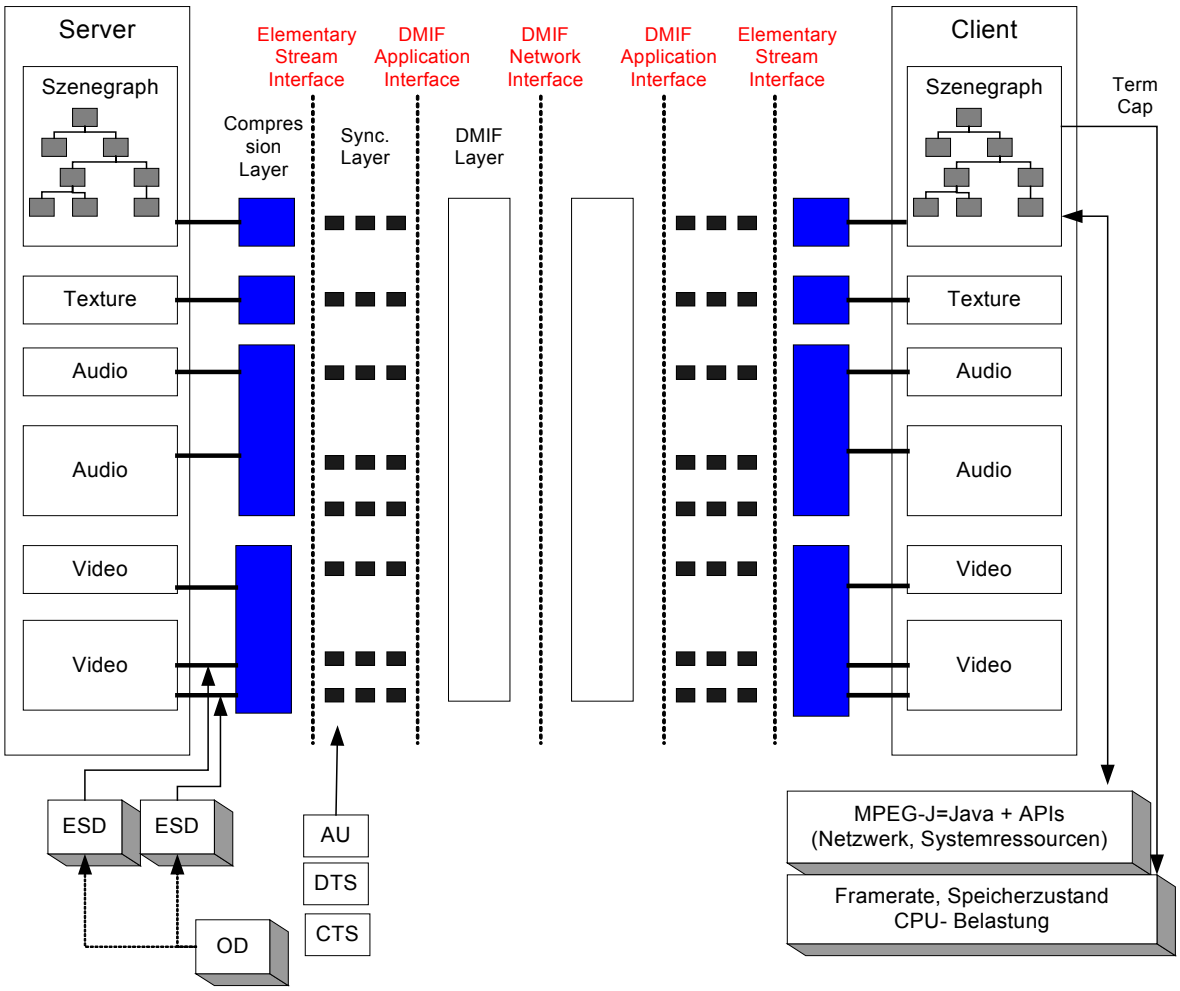


Abbildung 4-7: MPEG-4 Architektur [MPTU01]

Compression-Layer

Elementary Stream Descriptor

Object Descriptor

Im Compression-Layer werden die audiovisuellen Objekte komprimiert und in ein Binärformat umgewandelt. Dieser Stream wird über einen Elementary Stream Descriptor (ESD) referenziert. Ein einzelnes, audiovisuelles Objekt kann in mehrere Streams zerlegt werden. Für jeden einzelnen Stream wird nun ein Objekt-Descriptor (OD) angelegt, in dem vermerkt wird, wie die Decodierung zu erfolgen hat und in welchem Format das Objekt vorliegt.

Die im Compression-Layer komprimierte Szenenbeschreibung erreicht einen Komprimierungsfaktor von 10 bis 15.

Synchron-isations Layer

Decoding Time Stamp

Im Synchronisation-Layer werden die verschiedenen Streams in einzelne Einheiten aufgeteilt, die als Access Unit (AU) bezeichnet werden. Durch einen Decoding Timestamp (DTS) wird der jeweilige Decodierungszeitpunkt und durch einen Composition Timestamp (CTS) wird der Präsentationszeitpunkt der Access Unit bestimmt.

Nun werden im Multiplexverfahren die verschiedenen Ströme zu einem Einzelnen zusammengefasst und an den Client-Rechner übertragen. MPEG4 verfügt nicht über eine eigene Transportschicht, weshalb auf RTP[36], TCP, MPEG2-Transport Layer, H-323 oder ATM[37] zurückgegriffen werden muss. [TuIs01]

Auf dem Client-Rechner erfolgt nun wiederum das Demultiplexing in die einzelnen Streams, anschließend die Dekompression und die Synchronisation der Objekte. Anhand dieser Informationen kann nun mit dem Rendern der Daten begonnen werden.

Bei der Dekompression teilt der Demultiplexer den Code auf die verschiedenen Decoder auf. Der Code wird decodiert und dann vom Compositor synchron wiedergegeben. Die Szene wird im Binary Format for Scenes (BIFS) beschrieben. Dadurch erkennt der Multiplexer die Art der Objekte der Szene und teilt die Szene richtig auf.

Es gibt unter anderem extra Encoder für folgende Objekte: natürliche Bilder und Ton, Sprache, künstlicher Ton und Film, Texte, physische Objekte, etc. [Koen01, MPTU01]

Veranschaulichung der Szenerie

Beispiel zu MPEG-4

Ein Moderator sitzt in einem Büro mit Möbeln und spricht in die Kamera. Bei MPEG1 würde die ganze Szene, getrennt nach Film und Ton, übertragen. MPEG4 allerdings macht eine noch feinere Unterteilung. Dort werden die einzelnen Hintergrundobjekte (Möbel) als ein Objekt gesehen und mit einer niedrigen Bildwiederholfrequenz (z.B. 10pps[38]) übertragen, was keinen

[36] RTP = Realtime Protocol

[37] ATM = Asynchronous Transfer Mode

[38] PPS = Pictures per Second

Qualitätsverlust bedeutet, weil der Hintergrund konstant bleibt. Der Ton wird auch getrennt kodiert und gesendet. Der Moderator wird mit z.B. 25pps übertragen, weil er in Bewegung ist. Für den Betrachter dieser Szene läuft also alles flüssig ab, obwohl alle Objekte einzeln kodiert wurden und andere PPS-Raten (pictures per second) haben, was natürlich zu einer Reduzierung der Bitrate führt.

Skalierung

Skalierung

Die Algorithmen zur Komprimierung der Objekte sind parametrisierbar und lassen sich somit an die jeweilig verfügbare Bandbreite anpassen.

Streaming

Streaming

Es bestehen zwei Möglichkeiten der Übertragung einer Szene zum Client. Einerseits kann die gesamte Szene detailarm übertragen und Additiv verbessert werden, andererseits besteht durch das BIFS-Befehlsprotokoll und das BIFS-Anim-Protokoll die Möglichkeit, Szenendaten sukzessiv zu übertragen.

4.2.3.4 Präsentation

Player

Player

MPEG4 kann beispielsweise mit dem Windows Media Player dargestellt werden. Es gibt aber auch unzählige Shareware-Player, welche das MPEG4 Format unterstützen. Voraussetzung für die korrekte Darstellung ist ein MPEG4-Video-Codec. Der Codec wurde kostenlos zur Verfügung gestellt.

Skalierung

Skalierung

Durch den `TermCap`-Knoten ist es möglich, auf Daten, wie die aktuelle Framerate, den Speicherzustand und die CPU-Auslastung, zuzugreifen und entsprechend darauf zu reagieren.

Beispielszene MPEG-4 BIFS

```
Group {
DEF position Transform {translation 1 0 0children [Shape
               {geometry DEF MeineÜberschrift Text {
                    string „Test Laufschrift“
                    }}]}
}
DEF PosInterp PositionInterpolator {
     key [0 1]
     keyValye [1 0 0, -1 0 0]
     # Bewegung von x=1 nach x=-1
}

DEF MeineZeit TimeSensor {
     loop TRUE
     StoppTime -1}

ROUTE MeineZeit.fraction_changed TO PosInterp.keyROUTE
PosInterp.keyValue to position.translation
```

Listing 4-3: Beispielszene MPEG-4 *BIFS*

4.2.3.5 Bewertung

Einsatzgebiet

Einsatzgebiet

MPEG4 wurde für den Einsatz im digitalen Fernsehen, für interaktive Grafikapplikationen und für interaktive multimediale Anwendungen im Internet entwickelt.

Verbreitung

Verbreitung

DivX

Das DivX[39]-Format hat sich mittlerweile zum Standard bei der Verbreitung von Videodateien im Internet entwickelt. Mit MPEG4 codierte Videos werden sehr stark komprimiert. Während bspw. im MPEG2 Verfahren komprimierte Filme mit 2 Std. Spieldauer in einer akzeptablen Qualität noch einen Speicherplatz von 8.000 MB benötigen, kann der gleiche Film, in einer VHS übersteigenden Qualität mit Stereo-Vertonung auf ca. 650 MB gespeichert werden.

[39] DivX = Digital Video Express

Die Tatsache, dass MPEG4 durch reine Software-Decoder auf einem normalen Heim-PC in Echtzeit dargestellt werden kann, verhilft diesem Dateiformat zu einer weltweit rasanten Verbreitung.

Nutzungsbedingungen

Nutzungs-bedingungen

MPEG4 (ISO/IEC 14496 zertifiziert) kann bei der ISO käuflich erworben werden. Die Referenzsoftware MPEG4 ist durch internationale Patente geschützt. Die Dokumentation des kompletten Standards kostet zur Zeit ca. $ 1.100.

Zukunft des Formats

Zukunft

MPEG-7
MPEG-21

Mit MPEG7 und MPEG21 wurde bereits an einer Erweiterung des MPEG4 Formates begonnen. Diese Formate stellen aber keine Neuerungen im Bezug auf interaktive 3D-Animationen dar.

Ebenfalls eine neue Systemkomponente von MPEG4 ist das sogenannte XMT[40]-Format, welches eine textuelle Syntax für MPEG4 zur Szenen-Beschreibung ist. Somit wird Autoren von MPEG4 Inhalten der Austausch von Inhalten ermöglicht und die Zusammenarbeit mit X3D und SMIL[41] vereinfacht.

Zusammen mit dem Web3D-Konsortium wird weiter an der Entwicklung von 2D- und 3D- Animationen gearbeitet. [MPTU01, Koen01, Rukz02, Mild95]

Fazit

Fazit

Die bestehenden Implementierungen basieren zur Zeit hauptsächlich auf Audio- und Videobereiche. Es bleibt somit abzuwarten, inwieweit die anderen, umfangreichen Konzepte in Zukunft umgesetzt werden.

MPEG4 bietet umfassende Lösungen für die effiziente und skalierbare Übertragung der Objektdaten über das Internet und die Skalierung der Szene beim Client. Der Standard ist jedoch kompliziert und umfangreich, wobei die Dokumentation leider nicht frei verfügbar ist. Interaktive 3D-Szenen im MPEG4 BIFS Format sind noch nicht umgesetzt. Diese werden wahrscheinlich auch nie

[40] XMT = Extensible MPEG-4 Textual

[41] SMIL = Synchronized Multimedia Integration Language

im großen Umfang eingesetzt werden, da die Funktionalitäten für interaktive 3D-Objekte in MPEG4 zukünftig auf X3D aufsetzen werden.
Nachfolgende Illustration soll den Zusammenhang der bereits vorgestellten Technologien, wie MPEG4/BIFS, VRML und X3D nochmals verdeutlichen:

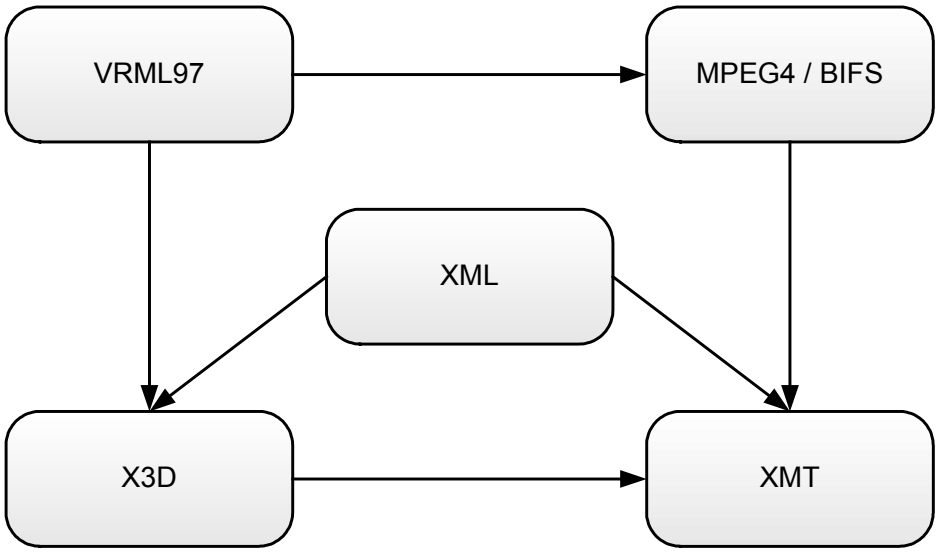


Abbildung 4-8: Abhängigkeiten der Formate

4.3 Nichtstandardisierte Formate

4.3.1 Shockwave3D

Shockwave3D wurde von Macromedia in Zusammenarbeit der Intel Architecture Labs (IAL) für die Macromedia Director Version 8.5 entwickelt. Das Havok-Team hat die 3D-Physik-Engine erstellt. Nx View hat die Software-Renderer-Engine verfasst, die eingreift, wenn das System des Benutzers kein Hardware-Rendern unterstützt.
Wenn das Client-System eine Grafikkarte mit 3D-Beschleuniger hat, so wird das Rendering der 3D-Szene auf Grundlage von OpenGL, DirectX durchgeführt, je nachdem was installiert ist. Ansonsten wird ein Software-Renderer benutzt.
Um 3D-Szenen abspielen zu können, wird ein PlugIn benötigt. Dieser SW Player erlaubt es, Mac und Windows Benutzern, 3D-Inhalte darzustellen. Der Shockwave Player bietet neben 3D-Darstellungen ebenfalls die Möglichkeit

Multimedia Komponenten, wie z.B. Text, Bitmaps, Sound und Flash-Filme darzustellen.

Um Szenen in Shockwave3D anzeigen zu können, muss die 3D-Szene im Shockwave3D-Format (W3D) vorliegen. Um Objekte, welche von professionellen Autorenwerkzeugen, wie z.B. 3ds Max von Discreet, im Macromedia Director 8.5 verwenden zu können, muss der Shockwave3D-Exporter installiert werden. Dieser kann von Macromedia oder Discreet heruntergeladen werden.

Abb. 4-9: SW- Player

In den nachfolgenden Kapiteln werden lediglich die Elemente zur Erstellung von Web3D-Grafiken mithilfe von Shockwave besprochen. Das erwähnte Autorenwerkzeug beschränkt sich auf Macromedia Director 8.5, da das, ebenfalls von Macromedia angebotene, Shockwave Studio den selben Funktionsumfang besitzt um 3D-Szenen darstellen zu können. [Gros02]

4.3.1.1 Erstellung

Werkzeuge

Werkzeuge

Macromedia Director

Shockwave3D-Dateien werden mit dem Macromedia Director entwickelt. Die Versionen ab Release 8.5 erlauben, neben dem üblichen Funktionsumfang, auch zusätzlich die Option der 3D-Webdarstellung.

Die Arbeitsoberfläche des MM-Director wird in eine Werkzeugpalette, eine Bühne, ein Drehbuch, die eigentliche Besetzung sowie den Eigenschaften-inspektor unterteilt.

Bühne
Besetzung
Darsteller
Drehbuch

So lässt sich MM Director am besten am Beispiel einer Theaterproduktion beschreiben. Die Show findet auf der Bühne statt. In der Besetzung sind alle Darsteller untergebracht (3D-Objekte, Sound, Bilder, usw.). Im Drehbuch wird das ganze Stück geregelt. Die lineare Zeitleiste im Drehbuchfenster bestimmt, was zu jedem beliebigen Zeitpunkt sichtbar ist, sich bewegt oder gerade zu hören ist.

Eine, beispielsweise in 3ds Max erstellte, 3D-Szene kann in den MM-Director importiert werden. Es besteht aber auch die Möglichkeit, einfache 3D-Szenen, grafische Primitive, Dreiecksnetze und Partikelsysteme direkt im MM-Director zu definieren.

Diese einzelnen Objekte (Sprites) können anhand einer Fülle von Funktionalitäten verändert werden. So ermöglicht die Bibliothekspalette im MM-

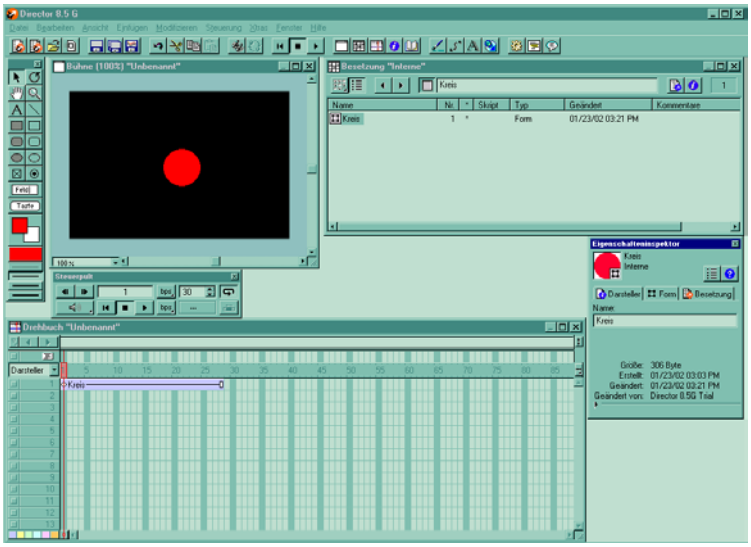


Abbildung 4-10: Macromedia Director 8.5

Director 3D-Modelle zu drehen, zu skalieren und erscheinen zu lassen. Über den Eigenschafteninspektor können einige Lichter und weitere 3D-Eigenschaften verändert werden.

Skriptsprache Lingo

Um benutzerdefinierte Eigenschaften zu erstellen, muss jedoch meist auf die MM-Director eigene Skriptsprache Lingo zurückgegriffen werden. Die Version 8.5 stellt insgesamt 300 neue Funktionen und Eigenschaften für 3D-Darsteller bereit. Lingo ist somit vom Funktionsumfang mit VRML97 und X3D zu vergleichen.

Lingo, und das schließt die mitgelieferten Verhalten in der Bibliothek, sowie anderen Quellen ein, kann sehr komplex und vielseitig sein. Es bietet dem Director-Anwender eine große Anzahl an Abfragen und Aktionen an. War Lingo anfänglich eine prozedurale Sprache, entwickelt sich Lingo immer mehr zu einer objektorientierten Skriptsprache.

Dateiformat

Dateiformat Shockwave 3D- Format (W3D)

Der Import von 3D-Objekten in den MM-Director erfolgt im proprietären, szenegraphbasierenden Shockwave 3D-Format (W3D). Die erstellten Objekte

werden im MM-Director im *.DIR Format gespeichert. Die eigentliche Publikation der Shockwave3D-Applikation erfolgt schließlich im *.DCR Format.
Die angegebenen Formate lassen sich nicht in einem Texteditor überarbeiten, da sie nicht im Klartext vorliegen.

Intenes Modell und interne Struktur

Internes Modell

Ein Shockwave 3D-Modell enthält alle benötigten Informationen, um eine 3D-Welt anzeigen zu können. Dazu gehören im Einzelnen die Geometriedaten der Modelle, die Modelle an sich, die (eventuellen) Texturen, die Shader, die Animationen , die Lichtquellen sowie die Kameras.
Für jedes dieser Objekte gibt es jeweils eine Palette in der Shockwave 3D-Welt. So werden die Geometriedaten intern in einer sogenannten Modellresource-Palette abgelegt. Die Modelle wiederum liegen alle in ihrer Modellpalette, genauso wie alle Shader in der Shader-Palette liegen, die Texturen in der Texturpalette usw.
Der Vorteil dieses zunächst etwas kompliziert anmutenden Aufbaus, liegt im Vorteil der Widerverwendbarkeit der Elemente. Hat man etwa die Geometriedaten einer Kugel, so können beliebig viele Kugelmodelle erzeugt werden, die diesen einen Satz als Geometriedaten benutzen.

parent- child linking

Shockwave Szenegraph

Die einzelnen Modelle, Lichter und Kameras werden in einem Szenengraphen organisiert, um die einzelnen Elemente besser bewegen und animieren zu können. Das Verbinden von Objekten nennt man „parent-child-linking“, also eine Eltern-Kind-Beziehung. Dabei wird ein Knoten als Elternteil definiert, welches dann dem Kindknoten als Bezugspunkt für die Bewegung, Rotation und Skalierung dient. Bewegt sich der Elternknoten, so bewegt sich der Kindknoten immer relativ zum Elternteil mit. Jeder Knoten (Node) kann jeweils nur einen Parent-Knoten haben. Ist kein anderer Parent-Knoten angegeben, so ist die Gruppe „Welt“ das Standard Parent jedes Knotens.

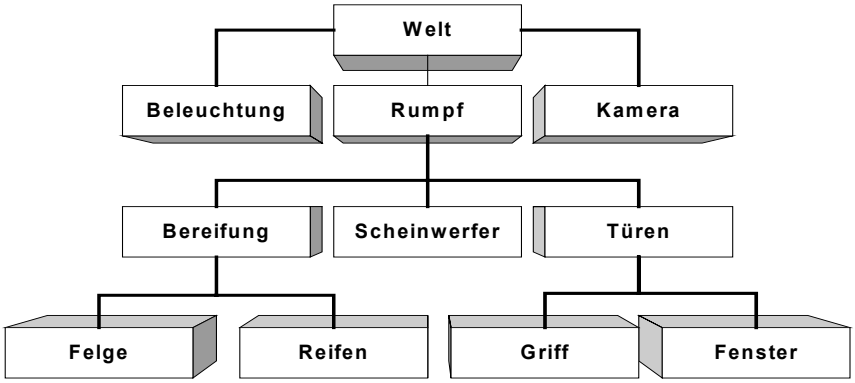


Abbildung 4-11: einfacher Shockwave Szenegraph

Somit ist jedes 3D-Modell eine Ansammlung von Zeigern auf einzelne Elemente innerhalb der bereits erwähnten Paletten. Ein Modell benutzt eine Modellressource in der Modellressourcen-Palette, für seine Geometrie einen oder mehrere Shader aus der Shader-Palette usw.

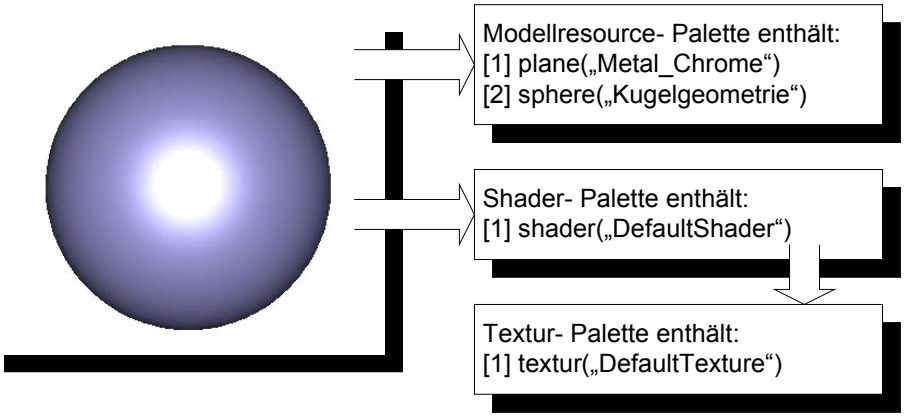


Abbildung 4-12: Referenzmodell einer Kugel SWF3D

Integration von Skript- und Programmiersprachen

Keyframe / Bones-animation

Netz-verformungen

Director 8.5 bietet einen immensen Funktionsumfang. Es gibt kaum ein Medienformat, das nicht integriert, kaum eine Programm-Anforderung die mit der eigenen Skripting-Sprache Lingo nicht umgesetzt werden kann.

Neben den grundlegenden 3D-Funktionen bietet Lingo insbesondere umfangreiche Animationen (Keyframe-, Bones-Animationen und Netz-verformungen) sowie Kollisionserkennung an. Über 300 neue Lingo-Befehle integrieren und kontrollieren interaktive 3D-Animationen, sowie Importe in den Formaten Real Audio, Real Video und Flash 5. Damit kann man etwa einzelne

Videoframes manipulieren oder als Textur für ein 3D-Modell verwenden. Auch Real-Audio-Dateien lassen sich mit Sound-Parametern beeinflussen.
Durch das Klassenkonzept enthält Lingo objektorientierte Ansätze, welches bspw. durch die Parent-Scripts (Klassen) und Child-Objects (Instanzen) repräsentiert wird. Macromedia Director kann durch spezielle Xtras im Funktionsumfang erweitert werden. [Schm01, GoalaJ]

3DPI- Xtra

Was die 3D-Programmierung anbelangt, ist das 3DPI-Xtra besonderes hervorzuheben. Mit 3DPI ist es unter anderem möglich das Hinzufügen von Kameras, Lichtern und Texturen, sowie das Einstellen von Havok-Eigenschaften zu tätigen. Der 3DPI Zusatz ist Shareware und kann für ca. $50 erworben werden.
Es existiert auch eine prototypische Implementierung eines Xtras, das es ermöglicht, bestehende X3D-Szenen als Darsteller zu importieren. [ScorMe]

Medienintegration

Medien-integration

Die nachstehende Tabelle zeigt, welche Medien in den Macromedia Director 8.5 importiert werden können:

Dateityp	Unterstützte Formate
Animation und Multimedia	Flash-Filme, animierte *GIF*s, Power Point Präsentationen, Director-Filme, externe Director Besetzungsdateien
Grafik	BMP, *GIF*, *JPEG*, MacPaint, PNG, TIFF, PICT, TARGA
Sound	AIFF, WAV, MP3-Audio, Shockwave-Audio, Sun AU, unkomprimiert und IMA komprimiert
Video	QuickTime 2,3,4,5 und AVI
Text	RTF, *HTML*, *ASCII*, Lingo Scripts

Tabelle 4-2: Medienintegration im MM-Director

Es ist zur Zeit lediglich nicht möglich, integrierte Sound-Dateien im dreidimensionalen Raum zu positionieren.

4.3.1.2 Kommunikation

Streaming

Streaming

Die erstellten Shockwave-Dateien können beim Abspielvorgang gestreamt werden.

Das große technische Problem jedoch ist, dass eine externe Shockwave 3D-Datei keine Informationen über den Zustand und Inhalt der 3D-Welt geben kann, solange sie nicht geladen ist. Somit kann bspw. kein 3D-Lingo Script ausgeführt werden, solange Director nicht „weiß“, wie die betreffende 3D-Welt aussieht und was für Elemente darin vorhanden sind.

State Eigenschaften

Dieses Problem wird mithilfe der Eigenschaftsvariablen `state` des jeweiligen Darstellers gelöst, welche den Ladezustand anzeigt.

Initial Loader Segment

Eine W3D-Datei besteht aus einem Kopfteil, dem sogenannten Initial Loader Segment (ILS) und einem Rumpfteil mit den eigentlichen Geometrie- und Texturdaten.

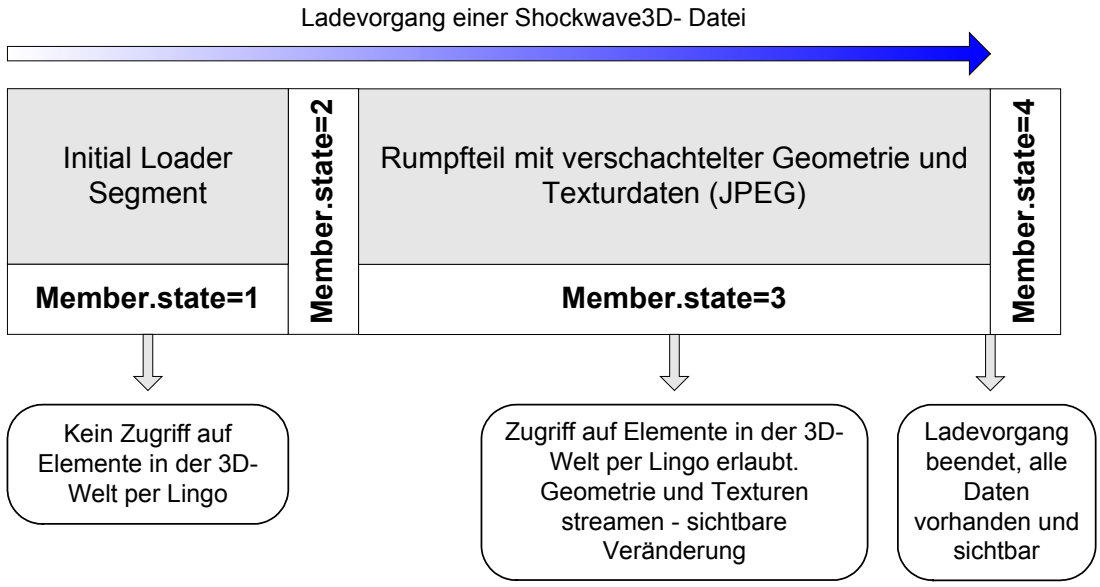


Abbildung 4-13: Ladevorgang einer Shockwave3D-Datei

Wird nun ein Shockwave 3D-Darsteller von Director in den Speicher geladen, so verändert sich der Wert der Darstellereigenschaft `state` während dieses Vorgangs von 1 bis 4. Ein `member.state` von –1 bedeutet, dass ein

Ladefehler aufgetreten ist. Es kann somit eine Anpassung der Szene entsprechend des Ladezustandes der einzelnen Objekte erfolgen. Hierfür gibt es Optionen im MM-Director, sowie spezielle Lingo-Anweisungen.
Das Streaming von 3D-Objekten kann durch folgende Lingo-Befehle gesteuert, bzw. der Status der einzelnen Objekte abgefragt werden: [Bied02]

- `member.preload` Soll der Darsteller in den Speicher vorgeladen werden?
- `member.percentStreamed` Zeigt an, wie viel Prozent der Datei schon geladen wurden
- `member.bytesstreamed` Gibt die absolute Anzahl der bereits geladenen Bites zurück
- `member.streamsize` Gibt die Gesamtgröße der Datei an
- `member.state` Gibt den Ladezustand zurück (0 – 4, bzw. –1)

Multi Resolution Mesh (MRM)

Diese Befehle sind eine Erweiterung für 3D-Objekte, zusätzlich zu den grundlegenden netzwerk- und streamingrelevanten Lingo Befehlen.
Geometrien, welche durch Exporter erstellt wurden, werden durch das Multi-Resolution Mesh (MRM) – Verfahren aufgeteilt. Bei dieser Technologie werden die Geometriedaten sukzessiv übertragen. Das Streaming erfolgt automatisch, entsprechend der zur Verfügung stehenden Bandbreite.
Das Beispiel zeigt die Multi-Resolution-Mesh Technologie anhand einer Comic-Figur.:

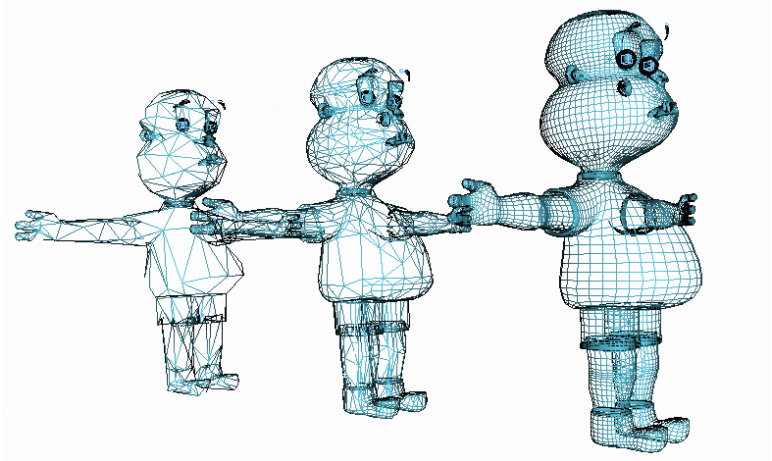


Abbildung 4-14: Beispiel der *MRM*-Technologie

4.3.1.3 Präsentation

Player

Player

Macromedia stellt den Shockwave-Player als PlugIn kostenlos zum Download bereit. Laut Macromedia wurde der Player bereits mehr als 350 Millionen mal installiert. Er steht wahlweise für Netscape und Internet Explorer zur Verfügung. Die aktuelle Shockwave-Player Version 8.5.1 hat eine Größe von ca. 3,4 MB.

Softwaretechnologischer Zugriff

LiveConnect

ActiveX Scripting

Für die Interaktion einer Shockwave-Animation in eine HTML-Seite stehen spezielle Lingo-Befehle zur Verfügung. Netscape verwendet LiveConnect und Internet Explorer arbeitet mit ActiveX Scripting. Die Verbindung zwischen Shockwave und Browser wird dabei über das PlugIn sichergestellt und spielt für den Entwickler somit keine große Rolle. [TechNo]

Folgende Lingo-Befehle stehen zur Zeit zur Verfügung:

- `netStatus` Stellt einen Text in der Statuszeile des Browsers dar
- `externalEvent` Durch diese Anweisung wird eine Nachricht von Shockwave an den Browser geschickt, z.B. ein JavaScript-Funktionsaufruf in der HTML-Seite
- `evalScript` Hierbei wird der evalScript Handler in Shockwave aufgerufen. Der Aufruf kann wiederum durch JavaScript erfolgen.

Skalierung

Skalierung

Subdivision surfaces (SDS)

Es besteht die Möglichkeit, Modelle in einer niedrigen Auflösung abzuspeichern, um Speicherplatz und damit Ladezeiten zu sparen. Diese Technologie wird auch mit dem Begriff subdivision surfaces (SDS) beschrieben. Bei der Darstellung werden diesem Modell dann interpolierte Gitternetzpunkte hinzugefügt und somit die Anzahl der Polygone vergrößert und die Darstellung verfeinert. Diese Technik bietet sich beispielsweise bei gekrümmten Oberflächen oder auch Landschaften (Terrains) an.

Somit hängt die Qualität des Modells nicht von der zur Verfügung stehenden Bandbreite, sondern von der Rechenleistung des Zielsystems ab.
Durch die Multi-Resolution-Mesh- (MRM) und die Subdivision-Surfaces- (SDS) Technologie ist somit eine Anpassung an die Bandbreite und die Konfiguration des Client-Rechners möglich.
Falls nicht anders definiert, sind die MRM- und SDS-Optionen standardmäßig aktiviert.

Beispielszene Shockwave3D Lingo

Abb. 4-15: Beispiel SW3D

```
on prepareMovie
member("Sonne").resetWorld()
mr1 = member("Sonne").newModelResource
("modRes", #sphere, #front)
radius = 50

mr2 = member("Sonne").newModel("KugelModel")
mr2.resource = member("Sonne").modelResource("modRes")

member("Sonne").newLight("ambientLicht", #ambient)
member("Sonne").light("ambientLicht").color =
rgb(210,210,210)

shd1 = obj.newShader("sunShader", #standard)
txtr1 = obj.newTexture("sunTexture",#fromCastmember,
member("sonnenTextur"))
obj.shader("sunShader").emissive = rgb(255,255,255 )
obj.shader[2].specular = rgb( 0, 0, 0 )
end prepareMovie
on exitFrame me
go to the frame
member("Kugel").model("KugelModel").transform.rotate(0,45,
0)

end
```

Listing 4-4: Beispiel eines Shockwave3D-Lingo Objektes [BalsAl]

4.3.1.4 Bewertung

Einsatzgebiet

Einsatzgebiet

Aufgrund der Multi-User-Funktionalität ist Shockwave3D für E-Commerce Anwendungen und Spiele geeignet. Durch die Anpassung auf Clientseite und die Streaming-Funktionalitäten, sowie die große Verbreitung des Shockwave-Players, ist Shockwave3D für eine breite Masse verfügbar. Die Erstellung der Modelle erlaubt, von einfachen Objekten bis zu komplizierten Charakter-darstellungen, nahezu alle Möglichkeiten der 3D-Präsentation und Interaktion.

Nutzungsbedingungen

Nutzungs-bedingungen

Der Shockwave3D-Player ist frei verfügbar. Lediglich das Autorensystem Macromedia Director, zur Erstellung der 3D-Objekte, ist mit ca. € 1.600 empfindlich teuer und somit lediglich professionellen Entwicklern vorbehalten.

Zukunft des Formats

Zukunft

Macromedia hat sich im Bereich der multimedialen Anwendungen im Internet etabliert und ist bei der Entwicklung des Macromedia Directors 8.5 mit mehreren Kooperationsfirmen auf Partnerschaften eingegangen, welche bspw. eigenständige Add-Ons für den Director 8.5 erstellen.
Damit ist Shockwave3D als eine zukunftsträchtige Technologie anzusehen, welche sich in nächster Zeit wohl noch mehr verbreiten wird.

Fazit

Fazit

Shockwave hat schon in der Vergangenheit eine breite Akzeptanz erfahren und kann somit auf eine große Anzahl Entwickler zurückgreifen, welche mit Macromedia Director und Lingo vertraut sind.
Durch die 3D Funktionalität und modernste Technologien, wie das Multi-Resolution-Mesh und Subdivision Surface Verfahren, sowie das exzellent ausgestattete Autorensystem, wird Macromedia in diesem Bereich auch in Zukunft sicher eine tragende Rolle spielen. [Inte02, HwoStu, Gros02, Bied02]

5 AKTUELLE SYSTEME

Im diesem Kapitel erfolgt eine kurze Übersicht der derzeit aktuellen, am weitesten verbreiteten und eventuell auch zukünftigen Technologien zur Darstellung von 3D-Grafiken im Internet. Auf eine Bewertung der einzelnen Systeme soll verzichtet werden. Da es zu den bereits vorgestellten aktuellen Web3D Formaten eine Menge verfügbarer und ähnlicher Lösungen gibt, ist es aus Quantitätsgründen nicht möglich, im Rahmen dieser Arbeit, auf jedes einzelne 3D-System ausführlich einzugehen. Lediglich das im praktischen Teil verwendete PlugIn von o2c und das Shout3D Applet wurden genauer betrachtet.

Das Bemühen, 3D Standards im Internet zu entwickeln, führte zu den unterschiedlichsten Formaten und Lösungen. Was die Umsetzung angeht, so wurden die verschiedenen Technologien aber meist eng an die bereits erwähnten Web3D-Formate angelehnt. Sie sind somit nicht als eigener Standard anzusehen und wurden deshalb im vorigen Kapitel vernachlässigt. Die Ausnahme bildet hierbei Macromedias Shockwave3D, da diese Technologie, obwohl es sich um ein nicht standardisiertes Format handelt, aufgrund der hohen Publikation detailliert betrachtet wurde.

Apple Quick-time VR

Adobe SVG Viewer

MM Flash Player

Technologien, die 3D-Szenen zwar darstellen und abspielen können, dies aber in sehr beschränktem Ausmaß, wurden gänzlich vernachlässigt. In diesem Zusammenhang wäre der Apple Quicktime VR, Adobe SVG Viewer und der Macromedia Flash Player zu nennen.

5.1 Auszug der Web3D-Systeme

Der folgenden Liste kann eine Auswahl der momentan für das Internet relevanten 3D-Systeme entnommen werden. Aufgezählt sind die jeweils verwendeten API's, 3D-API's, die unterstützten Formate und verfügbare Plattformen. [Goet02]

3dAnywhere

API:	Java	***3D-API***:	Keine
Format:	systemspezifisches 3DA Format, VRML		
Plattform:	Alle		
URL:	http://www.3danywhere.com/		

Adobe Atmosphere

API:	ActiveX	**3D-API:**	Direct3D
Format:	systemszpezifisches AER, MTX Format		
Plattform:	Windows, Macintosh		
Besonderheit:	speziell für virtuelle Welten, nutzt die Viewpoint Technologie		
URL:	http://www.adobe.com		

Blaxxun3d

API:	Java	**3D-API:**	Software Renderer
Format:	systemszpezifisches BX3D Format, VRML, X3D		
Plattform:	Windows, Macintosh, Linux		
Besonderheit:	sehr kleines Java Applet		
URL:	http://www.blaxxun.de		

---

blaxxunContact

API:	Java	**3D-API:**	OpenGL, Direct3D
Format:	systemszpezifisches BX3D Format, VRML, X3D		
Plattform:	Windows		
Besonderheit:	Multi-User fähig, weit verbreiteter VRML Viewer		
URL:	http://www.blaxxun.de		

Cortona

API:	ActiveX	**3D-API:**	OpenGL, Direct3D
Format:	VRML, X3D		
Plattform:	Windows, WindowsCE, Macintosh		
Besonderheit:	weit verbreiteter VRML Viewer		
URL:	http://www.parallelgraphics.com		

Cosmo Player

API:	ActiveX	**3D-API:**	OpenGL, Direct3D
Format:	VRML		
Plattform:	Windows, Macintosh		
Besonderheit:	weit verbreiteter VRML Viewer, wird jedoch nicht mehr weiterentwickelt		
URL:	http://www.cai.com/cosmo		

Cult3D

API:	ActiveX, Java	**3D-API:**	OpenGL
Format:	systemszpezifisches C3D, C3P Format		
Plattform:	Windows		
Besonderheit:	Einbindung der Szenen in MS Office Produkte möglich		
URL:	http://www.cult3d.com/		

Horizon

API:	ActiveX	**3D-API:**	OpenGL
Format:	VRML, X3D		
Plattform:	Windows		
Besonderheit:	wurde ursprünglich für NASA E-Training entwickelt		
URL:	http://www.openworlds.com		

Pivoron

API:	ActiveX	**3D-API:**	OpenGL, Direct3D
Format:	VRML, X3D		
Plattform:	Windows		
Besonderheit:	basiert auf Cosmo-Player, Web3D Mitglied		
URL:	http://www.nexternet.com		

Shockwave Player

API:	ActiveX	**3D-API:**	OpenGL, Direct3D
Format:	systemszpezifisches W3D Format, XML		
Plattform:	Windows, Macintosh		
Besonderheit:	weit verbreitetes PlugIn, basiert auf Intel-3D-Technologie		
URL:	http://www.macromedia.com		

Pulse3D

API:	ActiveX	**3D-API:**	OpenGL, Direct3D
Format:	systemszpezifisches Pulse3D Format, VRML, *DXF*		
Plattform:	Windows, Macintosh		
Besonderheit:	funktioniert auch auf PocketPC		
URL:	http://www.pulse3d.com		

Viewpoint Media Player

API:	ActiveX	**3D-API:**	Direct3D
Format:	systemszpezifisches MTS, MTX Format, *SWF*, XML		
Plattform:	Windows, Macintosh		
Besonderheit:	weitverbreitet		
URL:	http://www.viewpoint.com		

5.2 Zwei Web3D-Systeme im Detail

Wie bereits erwähnt, gibt es zur Zeit eine Vielzahl von Lösungen von Web3D Systemen. Die einzelnen Technologien werden teilweise weiterentwickelt, bzw. wurden bereits eingestellt. Die verfügbaren Viewer stehen entweder als PlugIn oder als Applet zur Verfügung.

In diesem Kapitel werden zwei Web3D Systeme detailliert betrachtet. Die Auswahl richtete sich nach der im praktischen Teil verwendeten Technologien. So war es mir wichtig, meine erstellte 3D Welt anhand eines PlugIns und eines Applets zur Verfügung zu stellen. Bei dem PlugIn von o2c handelt es sich um eine vergleichsweise junge Umsetzung eines 3D-Viewers. Shout3D wurde als Applet verwendet, da Shout3D als Mitglied des Web3D Konsortiums fungiert und durch das Shout3D-Applet großen Anklang in der Web3D-Welt gefunden hat, obwohl dieses 3D-System nicht mehr weiterentwickelt wird. [o2c03]

o2c- PlugIn

Shout3D-Applet

5.2.1 o2c

5.2.1.1 Der o2c-Player

Mit dem o2c- (objects to see) Player werden 3D Objekte fotorealistisch und kinematisch dargestellt. Die Darstellung der Objekte ist aber nicht nur auf das Internet beschränkt, sondern kann innerhalb der Standard-Office Anwendungen in Microsoft Betriebssystemen oder individuellen Softwareumgebungen eingebettet werden.

Die 3D-Objekte können mithilfe von JavaScript, Java oder VisualBasic und anderen Programmiersprachen, welche ActiveX unterstützen, interaktiv gesteuert werden. Der o2c-Player unterstützt den Microsoft Internet Explorer ab Version 4.0 und den Netscape Communicator Version 4.x unter der Betriebssystemplattform Microsoft Windows 95/98/2000/XP/NT ab 4.0. Es gibt keine weiteren Mindestanforderungen. Beide Player Varianten sind mit einem Authenticode-Zertifikat versehen.

Die Größe des o2c-PlugIns, der aktuellen Version 1.9.9.135. , liegt bei ca. 550 KB. [o2c03]

5.2.1.2 3D Modelle erstellen

o2c-Composer

3DS Import

Um 3D-Objekte darstellen zu können, müssen die Dateien im o2c-Format vorliegen. Hierbei wurde der o2c-Composer zur Verfügung gestellt. Durch den im Composer integrierten Plattendesigner ist es möglich, einfache Objekte zu erstellen und anhand einer Bibliothek einzubinden.

Der Modellierung sind jedoch Grenzen gesetzt, da z.B. komplexe Animationen im o2c-Composer nicht möglich sind. Deshalb besteht die Möglichkeit, dreidimensionale Szenen im 3ds Format einzubinden. Das 3DS-Format ist eines der gängigsten dreidimensionalen Objektformate und wird von so gut wie allen professionellen Modellierungssystemen als Exportformat unterstützt.

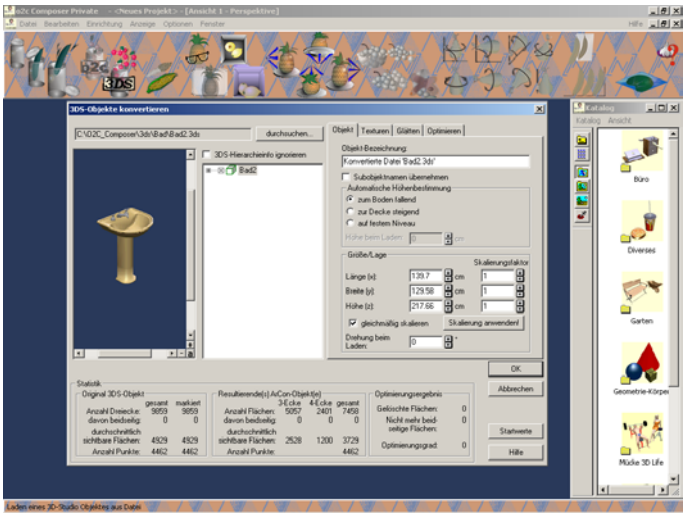


Abbildung 5-1: Der o2c-Composer

Einheiten / Objektgrößen

Einheiten / Objekte

Der 3Ds-Importfilter des o2c-Composers verwendet alle Geometrie-informationen von Objekten, die in den 3DS-Dateien gespeichert sind. Dazu gehören einerseits die Lage von Eckpunkten, andererseits aber auch Rundungsinformationen (Smothing-Groups) und Texturkoordinaten.

Materialien

Materialien

Materialien, welche bspw. im 3ds Max erstellt wurden, können nicht unbegrenzt verwendet werden. Der Grund hierfür ist, dass im o2c-Player die Objekte in

Echtzeit dargestellt werden müssen. Obwohl der Composer-Renderer extrem schnell ist, sind keine beliebig aufwendigen Materialeigenschaften möglich.

5.2.1.3 Lizenzbedingungen

Der o2c-Player wird dem Anwender kostenlos zur privaten oder gewerblichen Nutzung zur Verfügung gestellt. Lediglich für die Konvertierung und Erstellung von o2c-Business Objekten fallen Kosten an.
Es gibt zwei verschiedene Arten von Objekttypen:

Economy

Economy Objekte

Für Economy Objekte ist keine Lizenzgebühr fällig. Sie können z.B. mit dem Business Composer in beliebiger Zahl erstellt werden. Im Fuß der Grafik des o2c-Players steht lediglich der Schriftzug "powered by o2c" und im Kontextmenü erscheint ein kurzer Hinweis zum o2c-Lizenzmodell „Economy“ und ein Link auf http://www.o2c.de .

Business

Business Objekte

Für Business-Objekte ist eine Lizenzgebühr zu entrichten. Diese Objekte haben keinerlei Einschränkungen, können beliebig oft kopiert und unbegrenzt genutzt werden.
Ein o2c-Business Objekt kostet zur Zeit €15. [o2c03, o2cC03]

5.2.2 Shout3D

S3D und VRML in shout3D

Das nicht mehr vertriebene Shout3D-Applet hat in der Vergangenheit eine breite Akzeptanz zur Darstellung von Web3D-Szenen erfahren. Es handelt sich hierbei um eine Java-Anwendung, welche in der Version 2.5 ca. 110 KB groß ist. Das Java-Applet kann Szenen, welche im Shout3D (*.S3D) Format vorliegen, anzeigen. Der Shout3D Wizard importiert S3D und VRML97-Dateien und generiert eine Webseitenvorlage mit entsprechendem Applet für die Publikation der Szene. Die eingelesenen Daten werden komprimiert und in das *.S3Z- Format umgewandelt. [Shout3D, Pole99]

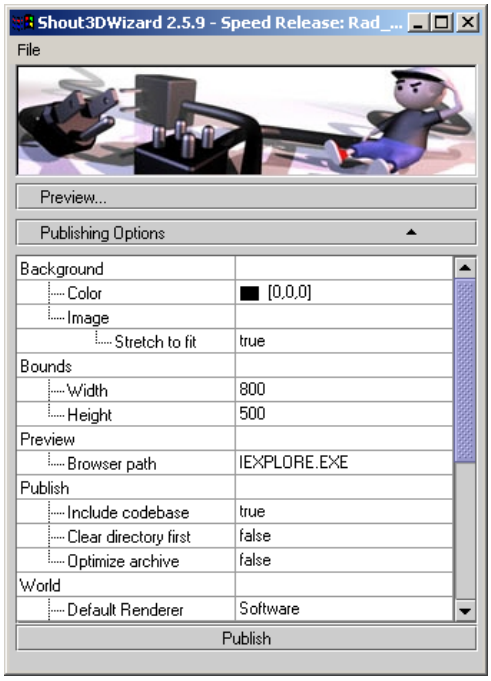


Abbildung 5-2: Shout3D Wizard

Blaxxun interactive

Ähnlich zu blaxxun3D von blaxxun interactive stellt Shout3D eine erste X3D-Implementierung dar. Shout Interactive war, bzw. ist an der Entwicklung von X3D im Web3D Konsortium beteiligt. Wobei Shout3D nicht dem aktuellen Stand der X3D Entwicklung entspricht. Eine Beschreibung der Szene durch XML ist nicht möglich. Die vollständige Abwärtskompatibilität zu VRML97 wurde ebenfalls nicht gewahrt.

Die grundlegende Technologie besteht darin, dass beim Client ein Java Applet innerhalb der Webseite angezeigt wird, welches eine eigene Rendering-Engine enthält. Dieses Applet kann durch den Autor beliebig verändert werden. Die Shout3D 2.5 Spezifikation ist abgeleitet von Shout Interactive's Core S3D von 1999.

Shout3Dapplet Shout3Dpanel Klassen

Um die 3D-Szene im Applet anzeigen zu können, muss ein Objekt der `Shout3Dapplet`-Klasse existieren, das als Container für ein Objekt der `Shout3D-panel`-Klasse fungiert. Die direkte Anzeige der Szene wird durch das `Shout3D-panel`-Objekt realisiert. Beide Klassen können beliebig erweitert werden, da alle Methoden `public` sind. Es steht ebenfalls eine Java-API zur Verfügung, die insbesondere einen komfortablen Zugriff auf den Szenegraphen mit seinen Knoten und Feldern ermöglicht.

5.2.2.1 Das Shout3D-Applet

Das Shout3D-Applet wird in eine Webseite anhand des Applet-Tags eingebunden. Der Webbrowser muss die Java Version ab 1.1 unterstützen. Durch die Attribute des Applet-Tags können dem Applet verschiedene vordefinierte Parameter übergeben werden. Hierdurch wird bspw. die Größe oder die Hintergrundfarbe bestimmt.

5.2.2.2 3D-Modelle erstellen

Spazz3D 3ds Max

Shout3D-Applikationen werden meist in 3ds Max erstellt, dem Charakter Studio von Discreet oder Spazz3D, da jeweils entsprechende Exporter in das *.S3D-Format existieren.
Die einzelnen Objekte können aber in jedem beliebigen Texteditor erstellt, bzw. überarbeitet werden.

5.2.2.3 Lizenzbedingungen

Die komplette Dokumentation und Software von Shout3D steht kostenlos zur Verfügung. Bei Publikation einer Szene wird aber immer ein verlinktes Shout3D-Logo angezeigt. Dieser Link ist nach Erwerb einer Lizenz nicht mehr vorhanden.
Eine Lizenz bezieht sich immer auf die URL der entsprechenden Webseite, in dem das Applet enthalten ist. Die Preise werden entsprechend der Anzahl, in dem ein Shout3D-Applet vorhanden ist, gestaffelt. [Shout3D, Pole99]

6 PRAKTISCHER TEIL DER DIPLOMARBEIT

Neben der Vorstellung der einzelnen Technologien der Web3D und Virtual Reality Technologie im Internet war es meine Aufgabe, eine entsprechende dreidimensionale Umgebung, anhand der in den vorigen Kapiteln vorgestellten Technologien, zu erstellen.

Was die Präsentation der einzelnen Objekte anging, so war es mein Bestreben, die einzelnen Szenen jeweils als PlugIn und als Applet zur Verfügung zu stellen. Ich entschied mich für das relativ junge o2c-Modell als PlugIn. Für das Applet war es meine Überlegung eine etablierte, wenn auch nicht mehr weiterentwickelte Technologie zu verwenden und entschied mich deshalb für Shout3D.

Das eigentliche Modell, welches dargestellt werden sollte, war ebenfalls schnell gefunden. So sollte es dem Benutzer möglich sein, ein neuartiges Produkt intuitiv und interaktiv näher kennen zu lernen.

Beispielmodell als PlugIn und Applet

Da ich zur Zeit an einer Umsetzung eines neuartigen, stufenlosen Fahrradschaltgetriebes arbeitete, wollte ich dieses Modell im Web3D-Format darstellen. So war es mir möglich, mich näher mit dieser anfänglichen Idee zu beschäftigen und nach und nach virtuell umzusetzen. Ob es sich bei diesem Entwurf um eine patent- oder schutzwürdige Entwicklung handelt, sei dahin gestellt. Die tatsächliche Realisierung dieser Idee wird wahrscheinlich nie erfolgen, da die Umsetzung zu komplex, umfangreich und folglich für den alltäglichen Betrieb zu massiv ausfallen würde.

Im Folgenden möchte ich nun auf die eigentliche Funktionsweise der Schaltung eingehen. Die Umsetzung und Realisierung wird in den Folgekapiteln behandelt.

6.1 Das Fahrradschaltgetriebe im Detail

Stand der Technik / Probleme

Das Problem der, dem heutigen Stand der Technik entsprechenden, Schaltgetriebe ist, dass ein Gangwechsel unter Last nur schwer und unter erheblichem Kraftaufwand zu realisieren ist. Desweiteren ist eine direkte

Kraftübertragung während eines Gangwechsels an das Antriebsrad nicht möglich. Ein weiteres Manko sind die vielen einzelnen Übersetzungszahnräder, welche eine exakte Schaltung im Dauerbetrieb nur schwer zulassen. Durch ein stufenloses Schaltgetriebe wird es ermöglicht, dass zu jeder Situation die entsprechend passende Übersetzung zur Verfügung steht.
Stehen bei einer herkömmlichen Radschaltung nur eine begrenzte Anzahl an Gängen zur Verfügung, kann durch diese Technologie eine unbegrenzte Anzahl an Übersetzungen gewählt werden.

6.1.1 Der Schaltzylinder

Das Modell im Detail

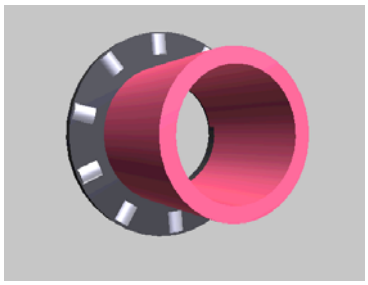


Abb. 6-1:Schaltelement

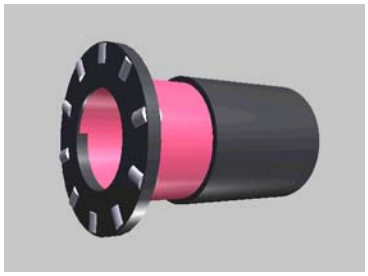


Abb. 6-2:Schaltzylinder

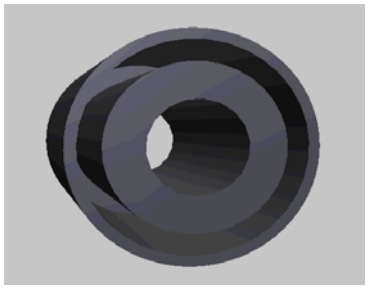


Abb. 6-3:Führungshülse

Die nebenstehenden Abbildungen zeigen den Schaltzylinder, welcher sich jeweils an der Hinterachse und der Tretkurbel befindet.
Der Schaltzylinder ist mit dem Rahmen fest verbunden, wobei durch die Führungshülse (Gehäuse Abb. 6-3) die Achse des Rades, bzw. die Tretwelle, geführt wird.
Der Zylinder besteht aus einem Mantel (Abb. 6-3), welcher eine Bohrung für die Fahrradachse, bzw. Tretwelle enthält. Zwischen dieser Bohrung und dem Außenmantel enthält dieser Zylinder einen Hohlraum, welcher nur nach oben offen ist, und das eigentliche Schaltelement (Rohr) in sich aufnehmen kann (Abb. 6-1). Der Boden des Zylinders ist zum Rad/Rahmen hin geschlossen. Es gibt nur eine Öffnung für die Zuführung des Öls. Man kann sich diesen Zylinder als normalen Hydraulikzylinder vorstellen, nur mit dem Unterschied, dass er eine Bohrung für die Fahrradachse/Tretwelle aufweist. Abbildung 6-2 soll den Zusammenhang der einzelnen beschriebenen Teile nochmals verdeutlichen. Wird nun der Öldruck, welcher durch eine Ölleitung

dem Zylinder zugeführt wird, erhöht, so wird das Schaltelement (Abb. 6-1) nach „außen“ gedrückt.

Am Kopf des beweglichen Schiebezylinders befindet sich eine Scheibe, welche fest mit dem Zylinder verbunden ist und mehrere kleine Rollenlager enthält. Diese Scheibe mit Rollen-, bzw. Kugellagern, bildet die Verbindung zwischen dem festsitzenden Schaltzylinder und der sich drehenden Rad-, Tretachse. Die eingezeichneten Rollenlager sollen dabei den Kontakt und die Reibung möglichst gering halten.

6.1.2 Teilansicht der hinteren Schaltkomponente

Funktionsweise am Beispiel der hinteren Radnabe

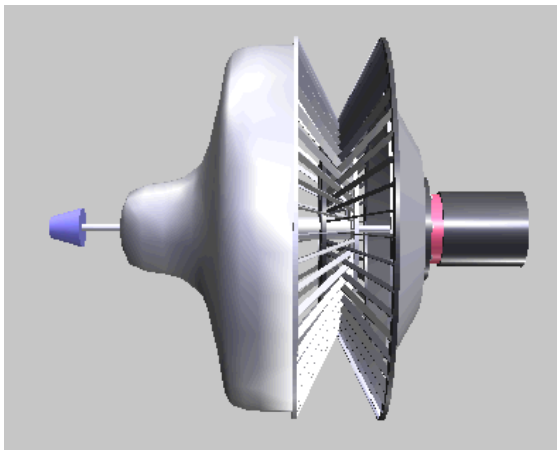


Abb. 6-4: Schaltkomponente 1

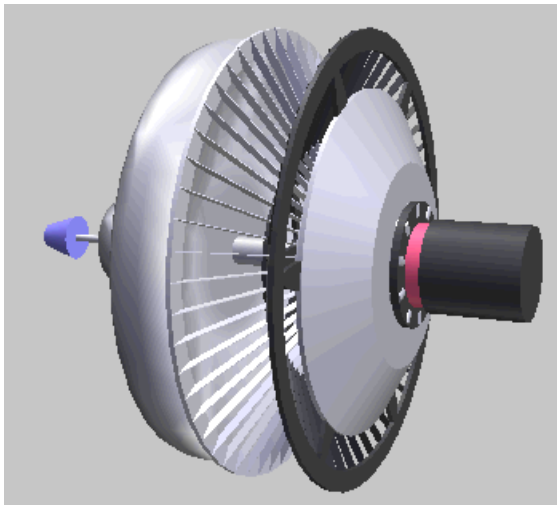


Abb. 6-5: Schaltkomponente 2

Die nebenstehenden Darstellungen enthalten Auszüge der Komponenten an den jeweiligen Achsen. In beiden Illustrationen befindet sich auf der rechten Seite wieder der Schaltzylinder, welcher hier nicht mehr näher erläutert werden soll. Die wichtigsten Bestandteile in dieser Abbildung sind die beiden kegelförmig eingezeichneten Elemente. Zu beachten ist hierbei, dass jeweils nur eine Komponente in seiner Stellung variabel, die andere fest gestaltet ist.

Der Kegel an der Tretkurbel ist zum Rahmen hin, der Kegel an der Hinterachse nach außen hin beweglich gestaltet.

In den Abbildungen 6-4 und 6-5 wurde jeweils die Radnabe des Hinterrades zeichnerisch dargestellt.

Abbildung 6-6 zeigt eine Einzelansicht dieser Komponenten. Pro Achse werden zwei solcher Elemente, welche ineinander „verzahnt“ sind, benötigt. Dieses

Abb. 6-6: Kegel

Einzelansicht der wichtigsten Komponenten

Bauteil lässt sich am einfachsten als kegelförmigen „Lampenschirm“ umschreiben.

Der Boden, bzw. Kopf des Kegels bilden jeweils zwei Ringe, welche durch Verstrebungen miteinander verbunden sind. Der innere Ring befindet sich unmittelbar um die eigentliche Achse und enthält „Führungsnasen“ (nicht dargestellt), welche die Stabilität der Positionierung erhöhen sollen. Hierbei ist zu beachten, dass die Achse einzelne, in Längsrichtung eingelassene Führungsschienen enthalten muss (nicht dargestellt). Der Boden unterscheidet sich vom Kopf nur derart, dass der Umfang des äußeren Ringes erhöht wurde, um die Anordnung eines Kegels zu gewährleisten. Diese beiden Elemente (Boden und Kopf) werden nun durch einzelne Lamellen verbunden.

Abbildung 6-4ff zeigt die „Verzahnung“ der beiden Komponenten. Die Schnittstellen der einzelnen Lamellen bilden den Umfang der Kettenführung. Dadurch, dass die beiden Komponenten leicht versetzt angebracht sind, „schneiden“ sich die einzelnen Lamellen. Jeder dieser Kreuzungspunkte bildet nun die einzelnen Angriffspunkte der im nachfolgenden Kapitel erläuterten Kette.

6.1.3 Die Antriebskette

Kettenglieder / Kraft-übertragung

Die Antriebskette lässt sich mit den herkömmlichen Fahrradketten vergleichen. Der Unterschied ist, dass sich an der Unterseite von jedem einzelnen Glied, ein oder mehrere Zähne befinden, welche sich in die Kreuzungspunkte der Lamellen „einhaken“.

Die einzelnen Kettenglieder können hierbei später etwas differieren, und sollen daher lediglich die Funktionsweise erklären. Es befinden sich in jedem Kettenglied zwei Gelenke, womit eine Bewegung der einzelnen Glieder horizontal und vertikal möglich ist (Abb. 6-7). Dadurch wird ein eventueller Reibungsverlust vermieden, da sich die Kette nicht nur um die Schaltelemente, sondern auch horizontal bewegen kann.

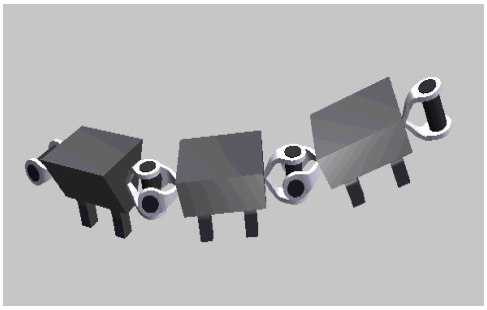


Abb. 6-7: Antriebskette

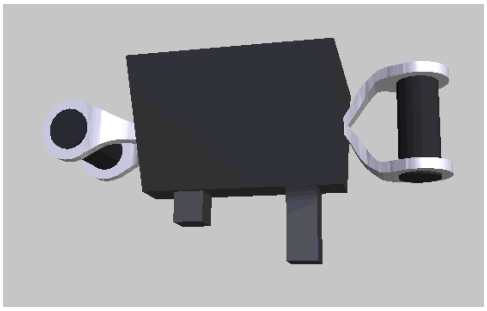


Abb. 6-8: Kettenglied

Einzelansicht Ketten-glieder

Wird nun geschaltet, „haken“ sich die in Abb. 6-7ff dargestellten Glieder in die Kreuzungspunkte der Lamellen der einzelnen Komponenten (Abb. 6-4, 6-5.) ein. Hierbei ist zu beachten, dass die Abstände der Lamellen sich nach außen hin erweitern. Im „Zentrum“ des inneren Rings sind die Lamellen gleichmäßig verteilt, ebenfalls auf dem Äußeren. Wobei sich der Abstand zwischen den Lamellen nach außen hin logischerweise erhöht. (Der Umfang des äußeren Ringes ist größer als der des inneren Ringes). Somit kann es vorkommen, dass die immer gleich lange Kette mit einem oder mehreren Zähnen direkt auf einen Kreuzungspunkt trifft. Sollte dies der Fall sein, greifen andere Kettenglieder innerhalb der Schaltkomponente. Damit die Kette weiterhin „rund“ läuft, sollten die Zähne der Kettenglieder vertikal beweglich sein, und einem auftretenden Widerstand nachgeben (Abb. 6-8). Man muss sich vorstellen, dass die Zähne durch eine Feder oder ähnliches nach außen gedrückt werden.

Diese Kette wurde nur als Beispiel dargestellt und sollte in der Realität viel mehr „Zähne“ und Glieder aufweisen. Es ist aber ebenfalls ein Riemenantrieb mit ähnlicher Funktionsweise denkbar.

6.1.4 Der Schaltvorgang

Zusammen-spiel der einzelnen Komponenten

Jede Achse besteht aus den gleichen, bzw. ähnlichen Komponenten. Die beiden Schaltzylinder sind in den Zeichnungen rot, die einzelnen Ölstände in den Schaltzylindern sind durch türkise (Hinterachse) und orange (Tretkurbel) Farben dargestellt.

Erhöht sich nun bspw. der Öldruck in dem Zylinder der Tretkurbel, so wird der bewegliche innere Schaltzylinder nach außen gedrückt. Dies hat zur Folge, dass sich die innere kegelförmige Komponente an der Tretkurbel nach „außen“ schiebt. Dadurch wird der Abstand zur anderen kegelförmigen, festen Komponente verringert. Der Reibungsverlust des fest angeordneten Schaltzylinders und der sich drehenden kegelförmigen Komponenten wird durch die in Abbildung 6-1/6-3 dargestellten Scheibe mit Rollenlager verhindert. Hierbei steht die Scheibe fest, ermöglicht aber eine Positionierung, sowie eine Drehung des kegelförmigen Schaltelements durch die eingezeichneten Rollenlager.

Abb 6-9: Die Schaltung im Detail

Abb 6-10: Fahrrad Gesamtansicht

Durch diese Verschiebung verändern sich die Schnittpunkte der Lamellen und bilden somit einen größeren Umfang der Kettenführung. Durch die schräg angeordneten Lamellen „rutscht“ nun die in Abbildung 6-7 eingezeichnete Kette an den Kreuzungspunkten nach „oben“ und verändert somit die Übersetzung des Antriebs.

Stellt man sich nun weiterhin vor, dass beide Schaltzylinder (Tretkurbel und Hinterachse) über einen gemeinsamen Öldruckbehälter (Abb. 6-10 türkise und orange Zylinder im Rahmen) verfügen, welcher bei Verschiebung (Schaltung) der vorderen Komponente den Öldruck im Tretkurbel- Schaltzylinder erhöht, und im selben Maße dadurch den Druck im hinteren Schaltzylinder verringert,

so erhöht sich der Abstand der beiden Komponenten an der Hinterachse. Die Kreuzungspunkte der Lamellen bilden nun einen kleineren Radius zur Achse gesehen, und verringern somit den Umfang der Kettenführung an der Hinterachse. Die Kette „rutscht“ an den schräg angeordneten Lamellen nach „unten“.

Simulation einer Talfahrt

Die in den Abbildungen 6-9ff dargestellte Stellung der Schaltung zeigt die „höchste“ Übersetzung, bspw. bei einer Talfahrt des Rades. Die Komponenten an der Tretkurbel bilden den höchst möglichen Umfang, wobei die Elemente der Hinterachse durch den Abstand der zwei kegelförmigen Komponenten den kleinsten Umfang bilden. Dies lässt sich ebenfalls an den Füllständen des symbolisch dargestellten Ölstandes innerhalb des Rahmens der Abb. 6-10. erkennen. Der türkise Zylinder ist mehr gefüllt, da der Schaltzylinder an der Hinterachse leer ist. Um den Abstand der Komponenten an der Tretkurbel und somit den Umfang zu erhöhen, muss Öl (orange dargestellt) in den vorderen Schaltzylinder „gepumpt“ werden. Somit ist der Füllstand des orangen Zylinders im Rahmen geringer. Abbildung 6-10 zeigt, durch die transparent dargestellten Elemente, dass der vordere Zylinder voll ausgefahren ist, durch den Druck des Öls (orange). An der Hinterachse ist der Schaltzylinder in der Ausgangsposition, da kein Druck des Öls ansteht (türkis).

Durch diese Anordnung wird nicht nur eine stufenlose Schaltung ermöglicht, es ist auch kein Kettenspanner, wie bei herkömmlichen Schaltungen, erforderlich. Der Umfang der Kette verändert sich nicht, da der Umfang einer Komponente (Hinterachse oder Tretkurbel) im selben Maße erhöht wird, wie sich der Umfang der anderen Komponente verringert.

Die kegelförmigen Elemente müssen jedoch nicht zwingend den selben Umfang an den entsprechenden Achsen aufweisen. Die gleichmäßige Veränderung des Umfangs kann dann durch unterschiedliche Ausdehnungen der Schiebezylinder erfolgen.

6.2 Realisierung des 3D-Modells

6.2.1 3ds Max

Was die eigentliche Umsetzung des Modells anging, so viel die Wahl auf das professionelle Autorensystem 3ds Max von Discreet. Die enorme Funktionsvielfalt von 3ds Max war anfänglich verwirrend, führte jedoch nach einer gewissen Einarbeitungszeit schnell zu Ergebnissen.

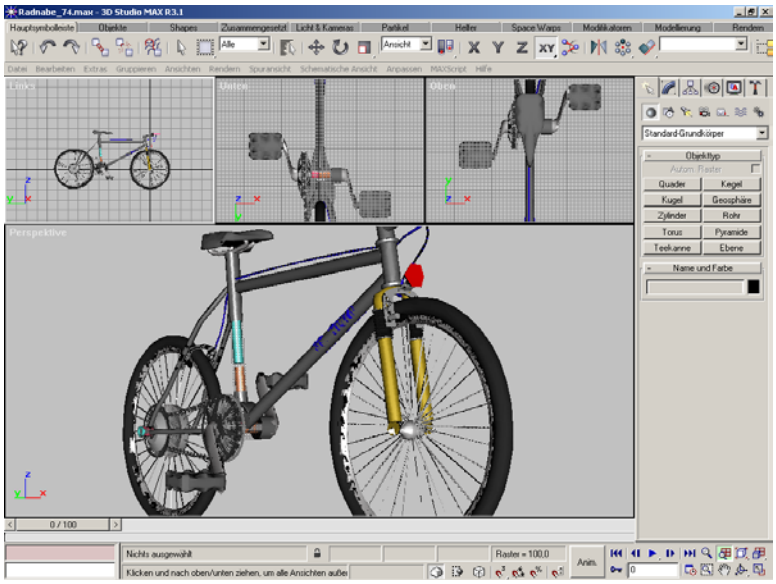


Abbildung 6-11: 3ds Max von Discreet

Da die Funktionalitäten dieses Autorensystems bei weitem den Rahmen dieser Arbeit sprengen würden (Die Handbücher zu 3ds Max umfassen ca. 2000 Seiten), sollen hier lediglich die Grundlagen, wie die einzelnen Objekte des Fahrradmodells erstellt und animiert wurden, beschrieben werden.

Space Warps
Modifikatoren
Animations-Controller

3ds Max basiert auf der objektorientierten Programmierung. Somit ist alles, was innerhalb dieses Autorensystems erstellt wird, ein Objekt. Dazu zählen unter anderem Geometrien, Kameras, Lichter, Partikelsysteme, Space Warps, Modifikatoren, Animations-Controller, Bilder und Materialien.

6.2.1.1 Das Master Objekt

Typen / Parameter

Ursprung und Ausrichtung

Das Master-Objekt ist die Objektbeschreibung, bis die Merkmale des Körpers durch die Angabe von Parametern gültig werden. Erst diese Zuordnung von Parametern und die damit verbundene Erstellung des Objekt-Datenflusses machen ein Objekt zu einem Körper.

Das Master-Objekt beinhaltet drei Objektinformationen:

- Typen (Wie z.B. Kugel, Kamera, Loft, Netz oder Patch)
- Parameter (Wie zum Beispiel Länge, Breite und Höhe eines Objekts)
- Ursprung und Ausrichtung eines Körpers (Das lokale Koordinatensystem definiert den Ursprung des Objekts, auf den sich alle Unterobjekte beziehen)

6.2.1.2 Der Objekt-Datenfluss

Alle Modifikatoren, Transformatoren, Space Warps und Eigenschaften zusammen definieren das Objekt. Die Verbindung nennt man Objekt-Datenfluss. Dabei wird ein Schritt nach dem anderen ausgeführt:

1.) Das Master-Objekt definiert den Typ und verwaltet die Grundparameter
2.) Modifikatoren verändern die Geometrie des Körpers und werden in der Reihenfolge ihrer Anwendung berechnet
3.) Transformatoren bestimmen Position, Rotation und Skalierung
4.) SpaceWarps greifen auf Geometrien zu und verändern diese
5.) Die zusätzlichen Eigenschaften identifizieren das Objekt und geben weitere Optionen zur Anzeige und zum Rendern an
6.) Das Objekt erscheint in der Szene

6.2.1.3 Objektarten / Modifikation / Animation

Objektarten in 3ds Max lassen sich in folgende Arten unterteilen:

<u>Shapes</u>

Shapes

Sind zweidimensionale Linien, wie Kreise, Rechtecke, usw. Sie besitzen ebenfalls bestimmte Eigenschaften und sind teilweise sogar parametrisch definiert.

Parametrische Objekte

Parametrische Objekte

Die einzelnen Werte der Objekte bleiben erhalten. So kann z.B. der Durchmesser, die Größe und die Segmentanzahl beliebig und zu jedem Zeitpunkt verändert werden

Nicht parametrische Objekte

Nicht parametrische Objekte

Im Gegensatz zu den parametrischen Objekten werden zunächst die Konstanten angegeben und daraus wird der Körper erstellt. Ist das Objekt einmal vorhanden, besteht es nur noch aus einer Flächenansammlung.

Compound Objects

Compound Objekte

Es können ebenfalls zusammengesetzte Körper und Objekte erstellt werden. Dies sind in der Regel parametrische Objekte. Sie speichern die Erstellungsparameter der einzelnen Körper und die Art und Weise Ihrer Vereinigung. So können bspw. Boolsche Operationen an verschiedenen Objekten vorgenommen werden.

Modifikatoren

Modifikatoren

Modifikatoren, Space Warps, Lichter, Kameras, Helfer, Animations-Controller und vieles mehr, besitzen keine sichtbare oder gar keine geometrische Form. Sie sind da, um Körper zu verändern und dienen der Visualisierung oder der Handhabung.

Einzelne Objekte können somit anhand der Modifikatoren gebogen, verjüngt, verdreht, rotiert, skaliert und positioniert werden. Auch lassen sich die einzelnen Flächen, Polygone und Scheitelpunkte als Unterobjekte bearbeiten und verformen.

Objekte Klonen

Klonen der Objekte

Einzelne Objekte können dupliziert werden. In 3ds Max wird dieser Vorgang als Klonen bezeichnet. Es wird dabei nach drei unterschiedlichen Arten unterschieden:

Kopien Instanzen Referenzen

- *Kopien*: Das neue Objekt ist ein genaues Abbild des Originals und Eigenständig.
- *Instanzen*: Eine Instanz ist ebenfalls ein völlig identischer Klon des Ursprungobjekts. Jedoch sind sie voneinander abhängig. Eine Veränderung eines der Objekte führt zur identischen Veränderung aller anderen Objekte. Es wird dabei nur das „Original“ abgespeichert und somit die Dateigröße reduziert.
- *Referenzen*: Sie sind gewissermaßen Instanzen, die nur in eine Richtung funktionieren. Alle Änderungen am Ursprungsobjekt werden auf seine Referenzen übertragen, jedoch nicht umgekehrt.

Konzept der Hierarchie

Hirarchien

An oberster Stelle steht die Welt, die eine gesamte Szene repräsentiert. In dieser Szene sind die gesamten diversen Einstellungen, wie die Umgebung, zugeordnete Klänge, Materialen und Objekte untergebracht.
Jedes einzelne Objekt wird nun wiederum in Material und Map-Hierarchien unterteilt.

Materialien und Texturen

Materialien und Texturen

Um einem geometrischen Objekt den Anschein von Realität zu geben, können ihm Materialeigenschaften zugeordnet werden. Diese Materialeigenschaften definieren, wie das Licht reflektiert wird oder durch ein Objekt hindurch scheint. Auf die Grundmaterialien lassen sich unter Zuhilfenahme von Mappings z.B. Schmutz, Staub, Etiketten, Transparenzen und Reflektionseigenschaften anbringen und verändern.

Animationskonzept

Animation

3ds Max besitzt eine Echtzeit-Funktionalität, die Bewegung über einen Zeitraum definiert. Am Anfang jeder Animation steht die Festlegung der Zeitparameter, die Art der Zeitanzeige, die Abspielgeschwindigkeit und die Dauer der Animation.

Die aufgeführten Objektarten, Transformations- und Animationskonzepte bilden nur einen kleinen Auszug der Funktionalität des Autorensystems. Mithilfe von Erweiterungen, anhand von PlugIns, kann der Umfang von 3ds Max an die Bedürfnisse der einzelnen Anwender angepasst werden.
Die Erstellung der Objekte des Fahrradmodells wurden in obiger Reihenfolge durchgeführt. So wurden die Objekte definiert, modifiziert, angeordnet und zum Schluss animiert.
Die Szene des Radmodells besteht aus einzelnen, einfachen Primitiven, zusammenhängenden Objekten, extrudierten[42], bzw. gedrehten Sprites, sowie aus, durch Patch-Verformung generierten, komplexen Objekten.

6.2.1.4 Beispiele zur Erstellung der Elemente

Einfache Grundformen

Grundformen

Rahmen-modell des Fahrrades

Der Rahmen des Fahrrades und weitere Elemente wurden anhand einfacher geometrischer Primitiven, wie z.B. Zylinder, Quader und Kugeln erstellt. So wurden bspw. die Streben der Hinterachse einzeln modelliert und später per Instanz geklont. Somit besteht der Rahmen aus einzeln modifizierten Zylindern, welche überarbeitet, positioniert und im Gesamtgebilde hierarchisch angeordnet wurden. Durch die Gruppierung werden die einzelnen Elemente jeweils immer proportional zu den gruppierten Objekten mitbewegt.

Scheitelpunkt-verformung

Das Gestänge wurde zunächst in zylindrischer Form erstellt, rotiert, gedreht und durch Umwandlung in ein bearbeitbares Netz, anhand einzelner Segmente, gebogen. Nebenstehende Abbildung veranschaulicht dies anhand der rechten Strebe der Hinterachse des Rahmens. Die rot markierten Scheitelpunkte symbolisieren das

Abbildung 6-12: einfache Primitive

[42] Mit diesem Begriff werden zweidimensionale Elemente bezeichnet, die durch „ziehen“ in dreidimensionale Objekte umgewandelt werden.

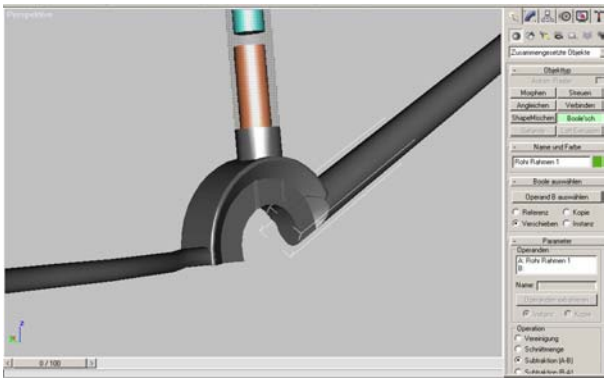
bearbeitbare Netz. Die einzelnen Scheitelpunkte wurden nun als Unterobjekte des Zylinders in ihre endgültige Position gebracht.

Zusammenhängende Objekte

Zusammen-hängende Objekte

Ein Beispiel für ein zusammenhängendes Objekt ist die Halterung der Querverstrebungen an der Hinterachse und die beiden Rahmenelemente. Durch die Schalteinheit an der Tretkurbel, musste ein Freiraum geschaffen werden, um die Elemente unterbringen zu können. Dies erfolgte durch Boolsche Operationen.

Boolsche Operationen



Durch Subtraktion der Schnittmenge der einzelnen Objekte wurde nebenstehendes Gebilde erstellt.

Der Hohlraum stellt den Raum für die angebrachte Schalteinheit zur Verfügung.

Abbildung 6-13: Boolsche Subtraktion

Shapes extrudieren und drehen

Shape-Bearbeitung

Elemente, wie Speichen, Bremsbrücke, Radnabe usw., bestehen aus einzelnen zweidimensionalen Shapes, welche entweder extrudiert, anhand eine Pfades extrudiert oder gedreht wurden.

Die Speichen bestehen bspw. aus einem Rechteck, welche gedreht und anhand von Instanzen um die Radnabe angeordnet wurden.

Die Bremsbrücke wurde anhand einer Linie gezeichnet, danach in die Höhe extrudiert. Man zeichnet somit den Grundriss eines Objektes und zieht diesen in die Höhe, um die räumliche Komponente einzufügen.

Ein Shape kann ebenfalls anhand eines Pfades extrudiert werden. So entsteht z.B. aus einem zweidimensionalen Rechteck, anhand einer Linie extrudiert, ein Quader. Diese Vorgehensweise wird Loft-Extrusion genannt.

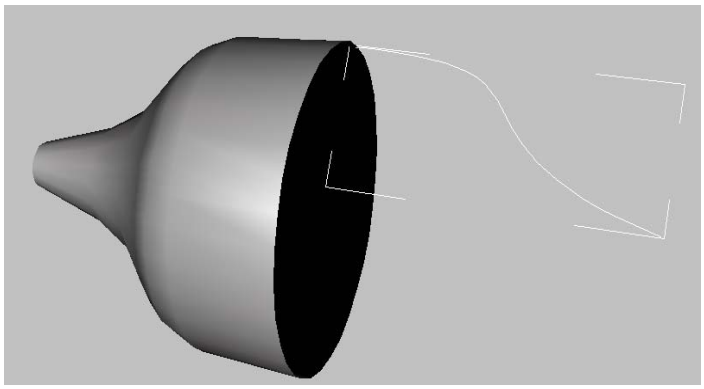


Abbildung 6-14: Nurbskurve gedreht

Nebenstehende Abbildung zeigt die Radnabe des Vorderrades. Sie wurde anhand einer NURBS-Kurve (Shape) erstellt (weiße Linie) und um 360° gedreht, danach geklont und gespiegelt.

Patch und Oberflächenverformung

Oberflächen-verformung

Jedes Objekt besteht aus einem Gitternetz, bzw. aus einzelnen Polygonen (Flicken/Patches), welche manuell nachbearbeitet werden können. So sind der Modellierung komplexer Gebilde keine Grenzen gesetzt.

Der Sattel des Fahrrades und die Tretpedale wurden anhand dieser Technik generiert.

Mesh Poly Patch und NURBS

Bei der Unterteilung des Objektnetzes wird in der Darstellung unter Mesh (Polygone mit jeweils drei Eckpunkten), Poly (Flächen mit mehr als drei Eckpunkten), Patch (besteht aus einem Gitter, das wie ein Netz in Einzelflächen aufgeteilt ist. Polygone lassen sich nachträglich erhöhen, bzw. verringern) und Nurbs (Non Uniform Rational B-Splines) unterschieden.

Anpassung des Polygon-Netzes

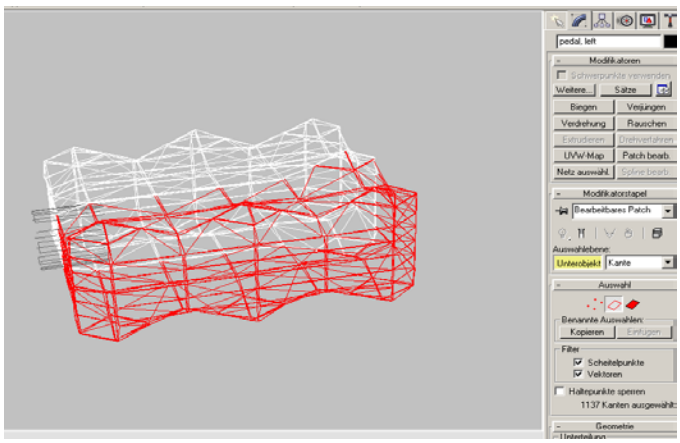


Abbildung 6-15: Patchverformung des Pedals

Nebenstehende Abbildung zeigt das Pedal, welches ursprünglich aus einem Quader in die endgültige Form gebracht wurde.

Hierzu wurde die Oberfläche in ein bearbeitbares Patch umgewandelt und die Scheitelpunkte in die endgültige Position gebracht.

Die rot markierte Hälfte wurde so generiert und durch Spiegelung und Verbindung des Polygonnetzes das endgültige Pedal erstellt.

Zuweisung des Materials

3ds Max Material-editor

Hier waren teilweise erhebliche Unterschiede im Bezug der Darstellung der einzelnen Materialien zu erkennen. 3ds Max bietet eine unbegrenzte Möglichkeit, Oberflächen zu generieren, die jedoch nur in sehr begrenztem Maße in die Web3D-Formate übernommen werden können.

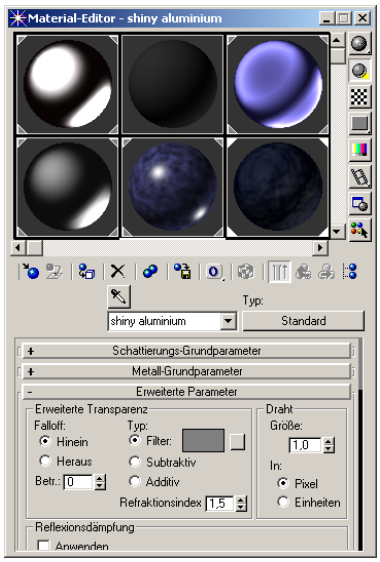


Abb. 6-16: Materialeditor

Material-eigenschaften

Der Materialeditor ist ziemlich komplex und erfordert, neben dem allgemeinen, Funktionsumfang eine längere Einarbeitungszeit. Anhand des Editors lassen sich alle Materialien abbilden und den generierten Objekten zuweisen. Um aber bspw. natürliche Materialien, wie z.B. Wasser, realistisch und mit der entsprechenden Lichtbrechung oder Reflexion darzustellen, ist einiges an Erfahrung im Umgang mit 3ds Max erforderlich.

Die Abbildung zeigt einen Auszug des Materialeditors zum Fahrradmodell.

Gruppierung der Objekte und Animation

Parent- / Child Gruppierung

Um das Fahrrad mit dem geringstmöglichem Aufwand zu animieren, wurden die einzelnen Teile miteinander verknüpft. Eine Verknüpfung besteht immer aus einem übergeordneten (Parent) und einem untergeordneten (Child) Objekt. Wird nun das Parent transformiert, bewegt, rotiert oder skaliert, folgt das Child dem Parent.

Die nachfolgende Abbildung zeigt einen Teil dieser hierarchischen Verknüpfung. So wurden die einzelnen Kettenglieder per Instanz vom Ursprungsobjekt erstellt, miteinander verknüpft, verbunden, und per inverse Kinematik animiert. Da die einzelnen Kettenglieder jedoch zu detailliert sind, war eine animierte Darstellung in diesem Umfang im Web3D-Format leider nicht möglich, da die Renderleistung der beiden Systeme schnell an seine Grenzen stieß.

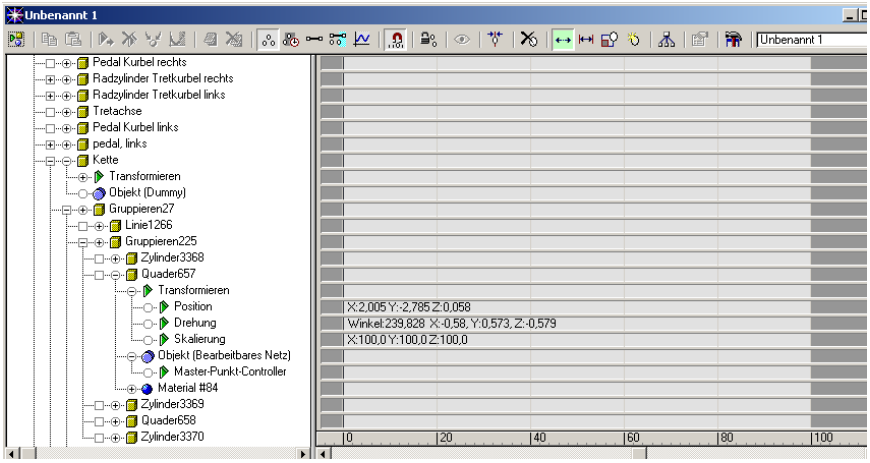


Abbildung 6-17: Hierarchische Anordnung der Elemente

Keyframe-Bild für Bild-Animation

Die Animation des eigentlichen Schaltvorgangs wurde ebenfalls in 3ds Max generiert. So läuft die Schaltung von der kleinsten bis zur größten Übersetzung durch, verändert dabei die Position der Schaltelemente und der Kette, sowie die Drehung der Räder. Der interaktive Eingriff (manuelle Veränderung der Schaltung) erfolgt zur Laufzeit der Darstellung im entsprechenden Web3D Format.

Das verwendete Animationskonzept basiert auf zwei Techniken. Die Bild für Bild Animation verschiebt die Objekte Stück für Stück anhand der Zeitachse (ähnlich wie ein Daumenkino). Bei der Keyframe-Animation wird ein Anfangs- und Endpunkt generiert. Die nötigen Zwischenschritte werden vom Computer selbständig festgelegt.

Durch die Keyframe-Animation ist es bspw. möglich einen Quader in eine Kugel umzuwandeln. Der Übergang und die Berechnung der Zwischensequenzen der Zeitachse erfolgt automatisch.

Bezier-, Linear-, TCB-, Listen-Controller

Während der Animation können die verschiedenen Objekte skaliert, rotiert, gedreht und verändert werden. Eine Ausrichtung der Elemente anhand eines Pfades ist ebenfalls möglich. 3ds Max stellt eine Vielzahl von Animationscontrollern, wie z.B. Transformations-, Parameter-, Bezier[43]-, Bezier-Positions-, Linear-, TCB-, Listen-Controller usw. zur Verfügung. Auf eine umfassende Veranschaulichung der verschiedenen Animationsmöglichkeiten

[43] beschreibt eine Freiformkurve, die in der konvexen Hülle gegebener Punkte liegt

wurde an dieser Stelle verzichtet, zumal ein Großteil der exportierten Animationen von den Web3D Formaten nicht, oder nur in unzulänglicher Form unterstützt wurden. Somit bestand die Hauptaufgabe, die einzelnen Elemente so zu animieren, dass sie im späteren Applet auch flüssig dargestellt werden konnten. Eine Nachbearbeitung der 3D-Dateien war aber dennoch unumgänglich. [Vels02, Max01, Max02, Max03]

6.2.2 o2c-Darstellung und Navigation

Um das Modell in einer hohen Detailgenauigkeit darstellen zu können, ist ein zusätzliches PlugIn notwendig, da ein Applet nicht die selben Darstellungs- und Renderfunktionen bietet. Die Entscheidung fiel hier auf das o2c-PlugIn.

3DS- Export von 3ds Max

Der o2c-Composer kann, wie bereits erwähnt, 3DS Dateien importieren und darstellen. 3DS Dateien als Exportformat werden von 3ds Max standardmäßig unterstützt. Es erfolgte jedoch eine umfangreiche Anpassung des *.max Files an das 3DS Format, da nur folgende Informationen in dieses Format aufgenommen werden können:

- Positions-, Drehungs-, und Skalierungsanimationen
- Grundmaterialfarbe und Parameter des Standardmaterials
- Einzelne Maps mit ihrem Betrag, Abstand, Skalierung
- Zielkameras, Zielspotlichter und Punktlichter

Einschränkung Des o2c-Players

Im 3DS-Export-Format werden die folgenden Informationen jedoch nicht aufgenommen:

- Zusammengesetzte Maps und Prozedur-Maps
- Transformationen an gruppierten Objekten (Der 3D-Editor kennt keine gruppierten Hierarchien)
- Globale Schattenparameter

Versuche, das detailgenaue Modell zu animieren, scheiterten an den umfangreichen Objekten und die daraus resultierende Polygonanzahl des Fahrrades. Dieses Modell ist somit als eine rein technische Darstellung zu verstehen. So wurde beispielsweise darauf geachtet, eine funktionsfähige

Darstellung der Antriebskette zu ermöglichen. Die einzelnen Kettenglieder wurden per Instanzen am Basisobjekt erstellt. Versuche per inverser Kinematik diese Kettenglieder in eine Rotationsbewegung zu setzen, scheiterten an der mangelnden Exportfunktionalität ins 3DS-Format und die zu geringe Renderleistung des o2c-Players.

Die interaktive, vereinfachte Schaltfunktionalität wurde somit im Shout3D-Format und in stark vereinfachter Form realisiert.

6.2.2.1 Die Konvertierung ins o2c-Format

3DS- Import in den o2c-Composer

Eine vorhandene 3DS-Datei wird im o2c-Composer umgewandelt. Hierbei stehen viele komfortable Funktionen zur Verfügung, um die Datei zu komprimieren und die Oberflächen zu optimieren. So können die nicht sichtbaren Flächen auf Wunsch automatisch gelöscht werden. Eine Auflösung des Modells kann ebenfalls gewählt werden und beeinflusst daher die Dateigröße und die Rendergeschwindigkeit. Texturen, sowie die Lage des Objektes, können ebenfalls überarbeitet werden.

Durch diese Funktionalitäten stehen dem Benutzer eine Vielzahl an Möglichkeiten zur Verfügung, um ein komplexes Objekt zur Übertragung im Internet zu optimieren, ohne an Darstellungsqualität des Modells einzubüssen.

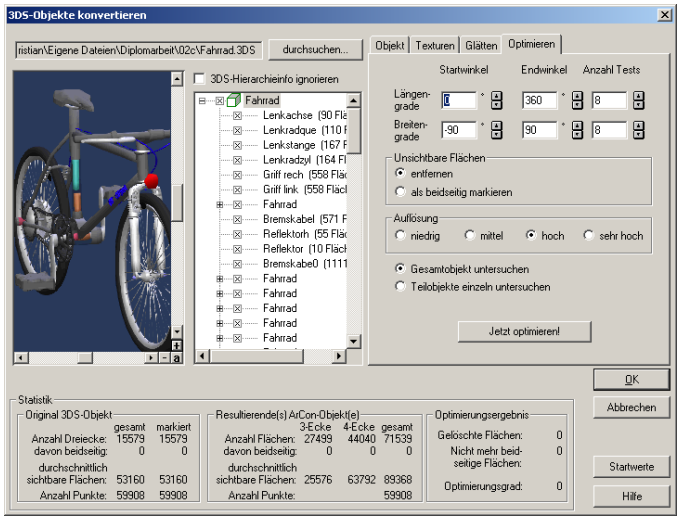


Abbildung 6-18: 3DS- Modell in o2c- konvertieren

Optimierung Umwandlung

Die nebenstehende Abbildung zeigt das Fahrradmodell in der Konvertierungs- Dialogbox.

Das umfangreiche Fahrrad-modell wurde um insgesamt 34.436 Polygone reduziert, ohne die Darstellungsqualität des Modells zu beeinflussen.

Das endgültige o2c-Fahrrad-Modell weist, im Vergleich zum 3DS-Format eine Komp-rimierung in Höhe von ca. 80% auf, und das bei einer hohen Darstellungsqualität.

6.2.2.2 *Die Einbettung in HTML*

OLE-ID

ActiveX-Control

Die Einbindung des o2c-Objekts erfolgt im Internet Explorer durch das `<OBJECT>`-, in Netscape durch das `<EMBED>`-Tag.

Das Objekt wird jeweils durch verschiedene Eigenschaften, wie die eindeutige OLE[44]-ID (Class-ID) des o2c-Players, die URL unter der das Installationsarchiv des ActiveX-Controls abgerufen werden kann, die Breite und Höhe, sowie der Name des Objektes, angegeben.

Anhand verschiedener Laufzeitparameter kann die konkrete Darstellung des Objektes nun beeinflusst werden. Die allgemeine Form dieser Parameter ist `<PARAMNAME=XXXVALUE=YYY>`. Dabei ist `XXX` der Name einer Eigenschaft und `YYY` der Wert, der dieser Eigenschaft zugewiesen werden soll. So kann anhand dieser Parameter bspw. eine Hintergrundfarbe, der Darstellungsmodus des Modells, die Rahmengröße usw. definiert werden. Eine umfassende Parameterreferenz ist unter http://www.o2c.de zu finden.

6.2.2.3 *Navigation und Interaktion*

Navigations-menü im o2c-Player

Per default sind im o2c-Player alle Elemente der Benutzeroberfläche aktiviert. Diese Option kann jedoch mit verschiedenen `<param>`-Tags eingeschränkt oder verhindert werden.

Das Popup-Menü kann direkt per Rechtsklick auf das jeweilige 3D-Objekt angezeigt werden:

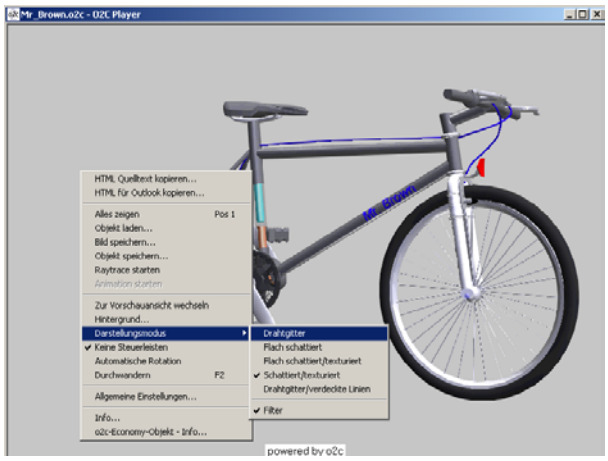


Abbildung 6-19: Dialogbox o2c

Anhand des Popup-Menüs kann bspw. ein Hintergrundbild geladen, ein Screenshot der Szene erstellt, die Darstellung des Modells (facettiert, Drahtgitter) verändert, eine Animation abgespielt, gezoomt, sowie das Objekt bewegt werden.

Die erwähnten Funktionalitäten

[44] OLE = Object Linking and Embedding

können ebenfalls anhand von JavaScript, Java oder VB-Script manipuliert werden. Dem Entwickler stehen somit umfangreiche Möglichkeiten zur Verfügung, durch Integration von Script- und Programmiersprachen, das dargestellte Objekt zu manipulieren.
Die verschiedenen Manipulationsmöglichkeiten lassen sich wie folgt einteilen:

- Auslesen der aktuellen Zustände bzw. Werte
- Änderung des Blickpunktes (Drehung, Verschiebung, Zoom)
- Veränderung des eigentlichen Modells

Eine Übersicht der zur Verfügung stehenden Methoden ist in der online-Dokumentation von o2c zu finden und soll deshalb hier lediglich anhand eines Beispiels erläutert werden.

6.2.2.3.1 Interaktionsbeispiel

Interaktion durch JavaScript

Die Umsetzung der Interaktion mit einem o2c-Objekt soll beispielhaft an der Zoom-Funktion dargestellt werden. Sämtliche Interaktionsmöglichkeiten dieser HTML-Seite wurden mithilfe von JavaScript gelöst, um die entsprechenden Reaktionen im o2c-Player hervorzurufen.

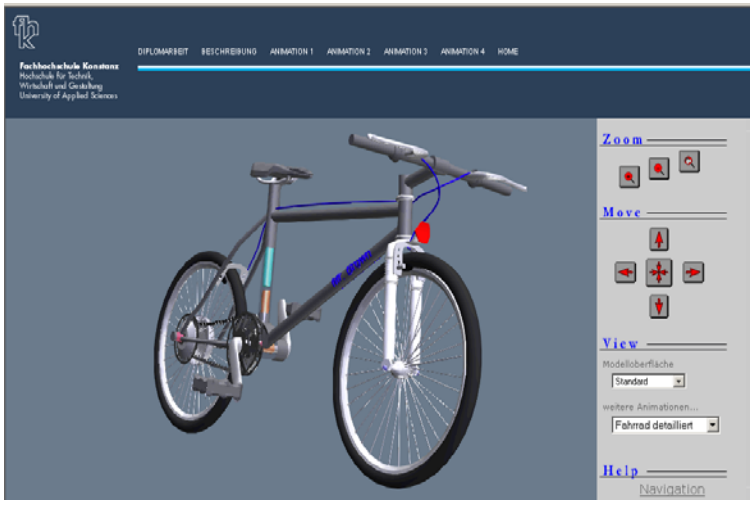


Abbildung 6-20: Fahrrad detailliert in o2c

Die Abbildung zeigt die Darstellung des detaillierten Fahrrades im o2c-Player. Die rechte Navigationsleiste erlaubt das Zoomen, das Bewegen und die Änderung der Oberfläche des Modells.
Die Einbettung des o2c-Objekts in die HTML-Seite erfolgte, wie bereits erwähnt, über das `<object>`, bzw. dem `<embed>`-Tag. Die Darstellung, der Hintergrund, die Berechtigungen der Nutzer usw. wurden über die `<param>`-Tags gelöst.

Wichtig hierbei ist folgender Auszug, welcher das Objekt mit der eindeutigen id (o2c_player) referenziert:

```
<object
classid="clsid:BF3CD111-6278-11D2-9EA3-00A0C9251384"
CODEBASE=http://www.o2c.de/download/O2CPlayer.CAB
width="800" height="500" border="0" id="o2c_player">
```

Wird nun beispielsweise der zoom-in Button vom Benutzer betätigt, so wird das o2c-Player Objekt an die o2c-API übergeben und ausgewertet:

```
<td valign="right" align="right"><br>
    <img border="0" id="zoomin" language="javascript"
     onmousedown="return zoomin_onmousedown()"
     onmouseout="return zoomin_onmouseout()"
     onmouseup="return zoomin_onmouseup()"
     onClick="objekt.style.display='block';
     hilfe_navi.style.display='none';"
     src="../images/o2c/zin.gif" width="28" height="27">
</td>
```

In obigem Listing erfolgt, je nach Benutzer Event (`onmousedown, onmouseout, onemouseup`), der Aufruf der jeweiligen `zoomin_` Funktion.

Setzen der Konstanten

Initialisierung der Variablen

Zuerst werden die erforderlichen globalen Variablen gesetzt. Sie beinhalten die Geschwindigkeit, mit der das Objekt gezoomt werden soll, den Zeitintervall in ms der vergehen soll wenn die linke Maustaste gedrückt bleibt, oder wann die eigentliche Funktion ausgeführt wird:

```
ZoomSpeed = 0.1;
ZoomDelay = 300;
ZoomIntervalDelay = 30;
```

Der Aufruf der `zoomin_onmousedown()`-Funktion:

Nach Betätigung des zoom-in Buttons wird die `onmousedown()` Funktion aufgerufen. Eine eventuell laufende Animation wird auf `false` gesetzt. In der nachfolgenden Zeile wird das veränderte gif-Bild zugewiesen, um den Tastendruck visuell darzustellen. Nun wird dem `o2c_player`-Objekt die Zoom-Geschwindigkeit zugewiesen und ausgeführt. Über die Standard Funktion `setTimeout()` erfolgt der Aufruf der eigentlichen Zoom-Funktion `zoomInterval()` durch Übergabe des strings („in“), nachdem das `ZoomDelay` Timeout abgelaufen ist:

```
function zoomin_onmousedown() {
  o2c_player.AnimPlaying = false;
  zoomin.src = bp_zoomin;
  MouseDown = true;
  o2c_player.Zoom = o2c_player.Zoom + ZoomSpeed;
  mTimeout = window.setTimeout("zoomInterval('in')",
ZoomDelay);
}
```

Der Aufruf der `zoomInterval()` – Funktion

Hierbei wird nun die `IntervalStarted` – Variable auf `True` gesetzt und das Objekt mithilfe von

```
o2c_player.Zoom=o2c_player.Zoom + ZoomSpeed
```

in der angegebenen Geschwindigkeit hineingezoomt. Diese Anweisung wird anhand der `setInterval()`-Funktion mit Unterbrechung des `ZoomIntervalDelay` wiederholt ausgeführt:

```
function zoomInterval(Button) {
  IntervalStarted = true;
  switch (Button) {
     case "in":
          mInterval =
```

```
            window.setInterval("o2c_player.Zoom =
            o2c_player.Zoom + ZoomSpeed",
            ZoomIntervalDelay); break;
        case "out": mInterval =
         … }}
```

Aufruf der zoomin_onmouseup() oder zoomin_onmouseout() Funktion

Diese Funktionsaufrufe bewirken den Abbruch der zoom-Funktion. Sie werden unmittelbar nach loslassen der linken Maustaste aufgerufen. So wird der zoom-Button wieder in die Ausgangsanzeige gebracht und zoomAbortInterval() ausgeführt:

```
function zoomin_onmouseout() {
     zoomin.src = b_zoomin;
     zoomAbortInterval(); }
```

Funktion zoomAbortInterval()

Hier erfolgt die „Löschung“ des Intervals und die angehaltene Animation wird wieder gestartet:

```
function zoomAbortInterval() {
  if (MouseDown == true) {
    MouseDown = false;
    window.clearTimeout(mTimeout);
    if (IntervalStarted == true) {
      window.clearInterval(mInterval);
    }
    IntervalStarted = false;
    if (AnimRunning == true) {
      o2c_player.AnimPlaying = true;
    }
  }
}
```

An dieser Stelle auf sämtliche Navigationsinstrumente einzugehen macht hier wenig Sinn, weshalb das oben genannte Beispiel genügen soll.

6.2.2.4 Fazit

Der o2c-Player und der dazugehörige o2c-Composer bieten dem Entwickler eine komfortable Lösung zur Erstellung komplexer 3D-Web-Applikationen in Verbindung mit dem 3d Studio Max von Discreet oder anderen professionellen Entwicklungstools.

Probleme ergaben sich lediglich dadurch, dass eine Vielzahl von generierten Oberflächen und Animationen im o2c-Composer entweder ignoriert oder falsch dargestellt wurden. Hierbei war ein enormer Nachbearbeitungsaufwand zu betreiben. Eine manuelle Nachbearbeitung einer *.3ds Datei in einem normalen Editor ist ebenfalls nicht möglich.

Ich hatte mich dazu entschieden, das Fahrradmodell in o2c mit einer hohen Detailtreue und der daraus resultierenden hohen Polygonanzahl darzustellen, da der Player eine hochwertige und sehr schnelle Rendering-Engine enthält und die Komprimierung der Dateien sehr effizient umgesetzt wurde. Dieses Modell soll somit den Aufbau der einzelnen Elemente in einer hohen Auflösung präsentieren. Dadurch konnte das Modell nicht interaktiv animiert werden, da die Rendering-Engine durch den hohen Grad der Detaillierung schnell an ihre Grenzen stieß.

hohe Render-geschw.

eingeschränkte Manipulations-möglichkeiten

Interaktionsmöglichkeiten zu realisieren ist durch die gut strukturierte online-Dokumentation relativ einfach umzusetzen. So lässt sich das Gesamtobjekt verschieben, Animationen stoppen, die Darstellungsqualität verändern, usw.

Ist es jedoch erforderlich einzelne Objekte in der 3D-Szene zu ändern, so ist dies nur in beschränktem Maße möglich. Interaktionsmöglichkeiten im Vergleich zur Shout3D Lösung zu integrieren, indem einzelne Elemente in Ihrer Position verändert, gedreht, skaliert oder erstellt werden, ist nicht möglich, da anhand der einzelnen Methoden nur in sehr geringem Umfang auf `Nodes` des systemspezifischen Formats Einfluss genommen werden kann.

Somit eignet sich o2c beispielsweise für detaillierte Produktpräsentationen mit beschränkten Interaktions- und Animationsanforderungen. [o2c03, o2cC03]

6.2.3 Shout3D- Darstellung und Navigation

Um 3D-Szenen in Shout3D darstellen zu können ist kein zusätzliches PlugIn notwendig. Auf dem Clientrechner wird ein Java Applet initialisiert, welches die 3D-Objekte darstellt.

Szenen in Shout3D werden im systemspezifischen Format *.S3D abgebildet. Durch den Shout3D-Wizard können aber auch VRML97-Dateien importiert werden. Die VRML97 Dateien werden analysiert und entsprechend der Konventionen in das Shou3D Format konvertiert.

S3D / VRML Einschränkung

Die Applets zur Szenenanzeige können hierbei durch den Autor beliebig verändert werden. Im Gegensatz zu VRML97 können keine Skripte und `PROTO`-Statements verwendet werden. Ebenfalls werden folgende VRML97- Knoten durch das Shout3D-Applet nicht interpretiert: `AudiClip`, `Collision`, `Cylinder`, `Fog`, `FontStyle`, `Inline`, `LOD`, `MovieTexture`, `Normal`, `NormalInterpolator`, `PixelTexture`, `PlaneSensor`, `PointLight`, `ProximitySensor`, `Script`, `Sound`, `SphereSensor`, `SpotLight`, `Text`, `TextureTransform`.

Es existieren insgesamt 86 Knoten und 12 Feldtypen im Shout3D Format.

JavaAPI Core- Package

Das Shout3D-Applet wurde zur Darstellung der interaktiven Fahrradschaltung gewählt. Durch die zur Verfügung stehende Java-API wird ein komfortabler Zugriff auf den Szenegraphen mit seinen Knoten und Feldern (Core-Package) ermöglicht.

Das Modell wurde im Vergleich zu o2c stark vereinfacht, um das Rendering des Modells während des interaktiven, stufenlosen Schaltvorgangs zu beschleunigen.

6.2.3.1 Die Konvertierung ins S3D bzw. S3Z-Format

S3D / VRML Export aus 3ds MAx

Die verwendeten S3D und VRML97 Dateien wurden anhand verschiedener Exporter im 3ds Max erstellt und im Shout3D-Wizard eingelesen, das Applet erstellt und die vorhandenen Dateien in das komprimierte S3Z-Format konvertiert.

Shout3Dapplet
Shout3Dpanel
ExamineApplet

Es muss ein Objekt der `Shout3Dapplet`-Klasse existieren, das als Container für ein Objekt der `Shout3Dpanel`-Klasse fungiert. Die direkte Anzeige der Szene wird durch das `Shout3Dpanel`-Objekt realisiert. Das `ExamineApplet` erlaubt dem Benutzer die Szene von jedem Blickpunkt aus zu sehen, in die Szene zu zoomen und die Maus bzw. die Tastatur zu verwenden. Das folgende Listing zeigt den `ExamineApplet.java` Source-Code:

```
package applets;

import shout3d.*;

public class ExamineApplet extends Shout3DApplet{

     public void initShout3DPanel() {
          panel = new ExaminePanel(this);
     }
     public void resetCamera(){
          ((ExaminePanel)panel).resetCamera();
     }
}
```

Somit beinhaltet jedes Shout3D-Projekt folgende Elemente:

- Eine Applet Klasse (`Shout3Dapplet`)
- Eine Panel Klasse (im Beispiel `ExaminePanel`)
- Der eigentliche Szene-File im *.WRL oder *.S3D-Format
- Eine HTML-Seite, welche das Shout3D-Applet enthält
- Nach Bedarf können noch Image- und Audio-Files integriert werden

6.2.3.2 Die Einbettung in HTML

Codebase, Archive des Applets

Die Einbindung des Shout3D-Applets erfolgt durch das `<APPLET>`-Tag, unter Angabe der Codebase (Verzeichnis der notwendigen Shout3D Java Klassen), dem Code (Angabe des Applets) und dem Archive (weist den Namen des Shout3D-zip Files auf, welcher die notwendigen Java Klassen zur Unterstützung des Applets enthält), den Namen des Applets, sowie die Angabe der Breite und Höhe zur Illustration des Objekts innerhalb der Webseite.

Ebenfalls kann anhand verschiedener Laufzeitparameter die konkrete Darstellung des Objektes beeinflusst werden. Die allgemeine Form dieser Parameter wurde im o2c-Beispiel bereits erwähnt. Hier kann nun ebenfalls anhand dieser Parameter bspw. eine Hintergrundfarbe (RGB Werte), der Darstellungsmodus des Modells, die Rahmengröße usw. definiert werden.
Auf die detaillierte Angabe der Parameterreferenz wird im Rahmen dieser Arbeit verzichtet.

6.2.3.3 Navigation und Interaktion

Manipulation des gesamten Objekts

Durch Verwendung des `ExamineApplets` ist es möglich, vordefinierte Interaktionen innerhalb des Java-Applets umzusetzen.
Die Drehung des Modells erfolgt durch die Maus (linke Maustaste), der Zoom und die Verschiebung wird mithilfe der Maus und Tastatur realisiert:

Zoom: linke Maustaste + <STRG>

Verschiebung: linke Maustaste + <SHIFT>

Abbildung 6-21: Shout3D- Java Applet

Nebenstehende Ansicht zeigt den Screenshot des Shout3D-Applets. Die Schaltungseinheit wurde entsprechend vereinfacht, um die Rendergeschwindigkeit zu erhöhen. Die Pedale wurden in eine Drehbewegung versetzt, der Schaltvorgang erfolgt in Realtime.
Die oben genannten, grundlegenden Objektmanipulationen wurden durch JavaScript-Integration ergänzt und sollen es dem Benutzer ermöglichen, den Schaltvorgang stufenlos zu betätigen und die entsprechenden Veränderungen im Java-Applet zu betrachten.

6.2.3.3.1 Interaktionsbeispiel

Interaktion durch JavaScript

Shout3D wurde in Java geschrieben und bietet somit dem Entwickler unbegrenzte Möglichkeiten, die dargestellten Objekte zu beeinflussen. Die verwendeten Dateien im Shout3D-Wizard (*.WRL, *.S3D) können anhand eines normalen Editors überarbeitet werden.

Anhand von Java oder JavaScript kann somit jeder Knoten des Szenegraphen überarbeitet und aktualisiert werden. Die Implementierung der Navigationsleiste des Beispiels erfolgte unter Verwendung von JavaScript bzw. eines Java Applets (Schieberegler).

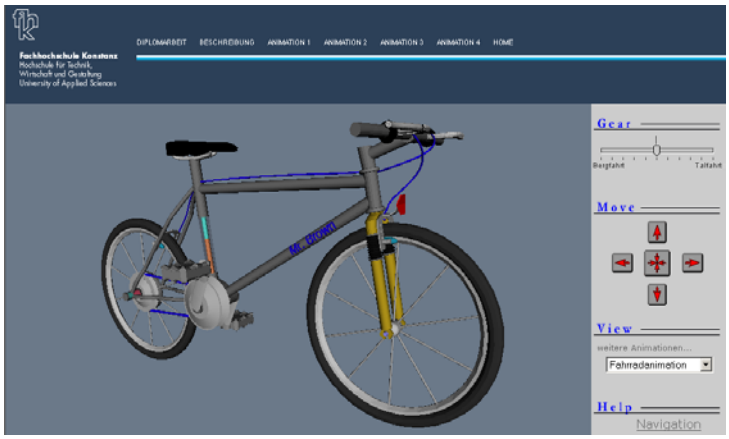


Abbildung 6-22: Shout3D Applet mit Navigation

Der Schaltvorgang erfolgt durch ein weiteres Java-Applet. Die Werte des Schiebereglers werden in Echtzeit umgesetzt und im Shout3D-Applet gezeigt.

Nachfolgendes Listing zeigt die Einbettung des Applets in die HTML-Seite. Auf Angabe der <PARAM>-Tags wurde verzichtet. Der Name des Modells ist `shout3D`. Über dieses Objekt erfolgen sämtliche Interaktionen mit dem Benutzer und dem Shout3D-Applet. Das verwendete Panel-Applet ist das bereits erwähnte `ExamineApplet`:

```
<APPLET CODEBASE="."
        CODE="applets/ExamineApplet.class"
        ARCHIVE="shout3dClasses.zip"
        NAME="Shout3D" WIDTH=800 HEIGHT=500>
        <param name="…" value="…">
            . . .
</APPLET>
```

Schieberegler als Schalteinheit

Nachfolgendes Beispiel soll den Vorgang nach Betätigung des Schiebereglers eines Benutzers aufzeigen. Die Implementierung dieses stufenlosen Reglers erfolgte ebenfalls in Java und wurde innerhalb der Webseite eingebunden.
Zwei Parameter innerhalb dieses Applets sind gesondert zu erwähnen:

```
<param name="notify"
value="segment1(document.regler1.value, 2);">
<param name="notifypromptly"
value="segment1(document.regler1.value, 2); ">
```

Beispiel zur Umsetzung des Schaltvorgangs

Durch die Definition dieser Parameter wird die `segment1` JavaScript Methode aufgerufen und die jeweiligen Werte übergeben. Der Parameter `notify` beinhaltet den Funktionsaufruf, nachdem eine Betätigung des Schiebereglers erfolgte, `notifypromptly` ruft die Funktion während der Schieberegler-betätigung auf. Somit werden die aktuellen Werte ständig aktualisiert und es entsteht die Illusion einer stufenlosen Schaltung.
Neben den üblichen Methoden (`init()`, `paint()` `repaint()`) wird die aktuelle Position des Schiebereglers im `sr.class` Applet ständig aktualisiert und an die JavaScript-Methode weitergegeben. Die Skalierung des Reglers variiert hierbei von 1 (Talfahrt) bis 100 (Bergfahrt).

Rückgabe der Werte aus `sr.class`

Nachfolgendes Listing zeigt die beiden Methoden `setElement()` und `setValue()` der `sr.class`, welche die Aktualisierung der Werte und die Übergabe an die JavaScript Methode veranlassen:

```
public void setValue() {
     value = ((xThumbLast-xMargin)*(rRight
     rLeft)+lsrAdd)/lsr+rLeft;
}
public void setElement() {
     setValue();
     if ( element != null )
               try {
                    element.setMember("value",
                    String.valueOf(value));
               }
```

```
                    catch (Exception e) {
                    }
          if ( notify != null )
                    window.eval(notify);
     }
```

Initialisierung der Arrays in JavaScript

Die Übergabe an die JavaScript Funktion erfolgt innerhalb der oben genannten `<param>`-Tags:

```
<param name="…" value="segment1(document.regler1.value);
```

Nach, oder während der Betätigung des Schiebereglers wird die `segment1()`-Funktion aufgerufen. Innerhalb des JavaScripts werden nun die einzelnen Arrays, welche die Positionsbestimmung der einzelnen Gruppierungen erlauben, einmalig initialisiert.

Den einzelnen Arrays kommen folgende Bedeutung zu:

- `kette_s_pos_x[]` (Position der Kette)
- `nabe_s_hi_pos[]` (Position der hinteren Radnabe)
- `nabe_s_vo_pos[]` (Position der vorderen Radnabe)

Diese eindimensionalen Arrays nehmen insgesamt einhundert Einträge auf, und werden je nach Wert des Schiebereglers (Skalierung von 1 – 100) an der entsprechenden Position ausgelesen.

Die Radnaben bewegen sich jeweils nur in eine Richtung (X-Koordinate). Die Y und Z Koordinaten bleiben jeweils gleich, weshalb nur die einzelnen X-Werte errechnet und initialisiert werden. Das nachfolgende Listing zeigt die Initialisierung und Zuweisung der hinteren Radnabe:

```
// Array der hinteren Radnabe (Schaltung) definieren
// Anfang: x:-10.58 y:2.503 z:-64.45
// Ende:   x:-26.38 y:2.503 z:-64.45
   nabe_s_hi_pos = new Array("-10.58");
  for(i=1; i<99; i++){
```

```
        nabe_s_hi_pos[i]=
        Math.round(nabe_s_hi_pos[i-1]*1000)/1000 - 0.158;
    }
```

Der selbe Vorgang wird bei der Kette durchgeführt, wobei sich die Kette nicht nur in X-Richtung bewegt, sondern auch in Z-Richtung (Bei Talfahrt ist die Kette relativ zum Rad gesehen weiter vorne positioniert, da sich der Umfang der vorderen Schalteinrichtung erhöht und umgekehrt). Somit werden diese Koordinaten errechnet und dem Array zugewiesen:

```
// Array der Kettenposition (Schaltung) definieren
// Anfang: x:-9.951 y:0.878 z:-274.4
// Ende:   x:-18.44 y:0.878 z:-277.9
    kette_s_pos_x = new Array("-9.951");
    kette_s_pos_z = new Array("-274.4");
    for(i=1; i<99; i++){
        kette_s_pos_x[i]=
        Math.round(kette_s_pos_x[i-1]*1000)/1000 -0.085;

        kette_s_pos_z[i]=
        Math.round(kette_s_pos_z[i-1]*1000)/1000 -0.035;
    }
```

Die Anpassung der Kette zum Umfang an die entsprechenden Schalteinheiten wird hier noch nicht berücksichtigt. Bei der Initialisierung erfolgt lediglich die Positionsbestimmung der erwähnten Elemente.

Aufruf der `segment1()`-Methode

In dieser Methode erfolgt die Positionsbestimmung und die Weiterleitung der entsprechenden Werte an den Shout3D-Szenengraphen:

```
function segment1(this_handle, call){
    var form_value=0;
```

```
    var string_tmp =" ";
    var call_funk = 0;
    form_value = this_handle;
    call_funk = call;

    // Umfang der Kette definieren
     var kette_umf =
     document.Shout3D.getNodeByName("MC_Zylinder01-
     BEZSCALAR-INTERP");
     string_tmp =form_value.toString();
     kette_umf.keyValue.setValueByString(string_tmp);
   . . .
}
```

Dieser Methode werden die Parameter des Schieberegler-Applets übertragen und ausgewertet. Im ersten Abschnitt werden die Werte des Knotens zum Umfang der Kette ausgelesen. Die zur Bezier-Skalar Interpolation animierten Kette erforderlichen Daten sind im Knoten `MC_Zylinder01-BEZSCALAR-INTERP` zu finden. Um diese Kette in die gewünschte Ausrichtung zu bringen, (der Umfang der Kette wird dem Umfang der entsprechenden Komponenten angepasst, es erfolgt keine Positionsveränderung), wird der `keyValue` dieses Knotens an die Schiebereglerstellung angepasst.

Per `getNodeByName()` werden die Werte der `kette_umf`-Variable zugewiesen und per String der jeweilige Wert (zwischen 1-100) anhand der Anweisung:

```
kette_umf.keyValue.setValueByString(string_tmp);
```

übertragen.

Die Änderung der Positionen der Kette und den Schaltelementen erfolgt nach einem ähnlichen Prinzip innerhalb der `Segment1()`-Methode:

```
// Position der Kette
//
string_tmp =kette_s_pos_x[form_value]+
             " 0.878 " +   kette_s_pos_z[form_value];
kette_pos = document.Shout3D.getNodeByName
          ("Zylinder02-BEZPOS-INTERP");
kette_pos.keyValue.setValueByString(string_tmp);
```

Obiges Beispiel zeigt die Zuweisung des entsprechenden Array-Wertes der Variable `string_tmp`. Die Konstante 0.878 repräsentiert die Position der Kette in Y-Richtung (nach „oben“). Da sich die Kette nicht in diese Richtung bewegt, bleibt sie gleich.
Abschließend werden nun die Werte anhand `getNodeByName()` und `setValueByString()` des `keyValue` innerhalb des entsprechenden Knotens verändert.
Durch die Skalierung des Schiebereglers von 1-100 ist somit keine stufenlose Schaltung möglich, sondern eine Unterteilung in 100 Teilschritte. Eine weitere Verfeinerung wäre in diesem Zusammenhang aufgrund der Animationseffizienz nur sehr schwer umzusetzen.

6.2.3.4 Fazit

Unzureichende Renderfunktion

Umfangreiche Eingriffs-Möglichkeiten

Mit Shout3D wird dem Entwickler ein mächtiges und umfangreiches Instrument zur Verfügung gestellt, Web3D-Szenen darzustellen. Die Open-Source Java Technologie setzt dem entwickeln interaktiver 3D Szenen keinerlei Grenzen. So ist es möglich unmittelbar in den Shout3D-Szenegraphen einzugreifen und einzelne Elemente nach Belieben zu manipulieren.
Die JavaScript Lösung stellt dabei nur eine Möglichkeit dar, in das Geschehen einzugreifen. So ist es bspw. ebenfalls denkbar, einzelne Panels direkt in das Applet einzubauen und somit innerhalb eines Applets die Darstellung zu verändern.
Lediglich bei der Erstellung des Modells gab es Probleme, da einzelne Animationssequenzen und Objekte nicht, oder nur in unzulänglichem Maße vom Shout3D-Applet umgesetzt wurden. Somit war ein erheblicher

Nachbearbeitungsaufwand innerhalb des 3ds Max Autorensystems, sowie den einzelnen Export-Dateien des systemspezifischen *.S3D-Formats notwendig.
Die Entscheidung Shout3D zu verwenden kam durch mein Bestreben, jeweils eine Web3D Szene anhand eines PlugIns und die zweite Darstellung innerhalb eines Applets darstellen zu wollen. Somit ist es im Vergleich zu o2c nicht notwendig ein zusätzliches PlugIn zu installieren und man ist folglich durch das Java- Applet auch weitestgehend plattformunabhängig. Die Darstellung und Rendergeschwindigkeit eines Applets, im Vergleich eines PlugIns, lässt jedoch meist zu wünschen übrig.
Dadurch eignet sich die Shout3D, oder ähnliche Lösungen, zur Darstellung einfacher Gebilde, wobei die Interaktionsmöglichkeiten und die Einflussnahme auf die 3D-Szene Vergleiche sucht.
Die Darstellung der Modelle sollte den jeweiligen Leistungen der 3D-Systeme Rechnung tragen. Ich entschied mich deshalb für die Erstellung des detaillierten, nicht animierten Modells, für o2c und die vereinfachte, jedoch für Anwender leicht verständliche und die eigentliche Funktionsweise des Schaltmodells nahe bringende Animation, für das Shout3D Modell.
[Shout3D, Pole99, Krue00, Musc01]

7 RÜCKBLICK

Durch diese Arbeit wurde es mir ermöglicht, eine anfänglich vage Idee über ein neuartiges Produkt einer Machbarkeitsstudie zu unterziehen und eine Grobkonzeption zu einem funktionsfähigen Modell zu entwickeln.
Die eigentliche Überlegung, ein Fahrradschaltgetriebe zu entwickeln, welches sich stufenlos betreiben lässt, kam durch meine unternommenen Radtouren. Nach etlichen Zwischenfällen und dem ständigen Gefühl, die gerade benötigte Übersetzung nicht zur Verfügung stehen zu haben, arbeitete ich daran, mir Gedanken über eine mögliche Alternative zu machen. Anfängliche Skizzen und Entwürfe stellten sich jedoch frühzeitig als zu kompliziert oder undurchführbar heraus.
Der endgültige Entwurf entstand zum Teil erst durch die Entwicklung des funktionsfähigen Modells. Nachdem ich der Auffassung war, ein funktions- und umsetzungsfähiges Schaltmodell entwickelt zu haben, trat ich an das Patentamt München, um den Entwurf rechtlich schützen zu lassen. An dieser Stelle gilt mein besonderer Dank Herrn Prof. Hedtstück, der mich ermutigte, diesen Entwurf in das jetzige Modell umzusetzen.
Durch die Möglichkeit, die anfängliche Konzeption in meine Diplomarbeit mit einzubinden, wurde es mir erleichtert, mich intensiver mit dieser Thematik zu beschäftigen.
Interessierten sollte es ermöglicht werden, anhand eines Web3D-Formates einen Einblick über die Funktionsweise der Fahrradschaltung zu erhalten.
Somit verbrachte ich etliche Zeit damit, mir über die Umsetzung der eigentlichen Schaltung Gedanken zu machen, Recherchen über ähnliche Konzeptionen durchzuführen und eine eventuelle Patentfähigkeit abzuwägen. Die Erfahrungen, welche ich während des Anmeldeprozesses mit dem Patentamt sammeln konnte, waren sehr lehrreich. Nach eingehender Nachforschung über ähnliche Produkte in diesem Bereich stellte sich heraus, das bereits ähnliche Entwicklungen geschützt waren, jedoch nie realisiert wurden. Die in den vorigen Kapiteln beschriebene Schaltung existiert in dieser Form noch nicht, doch sind ebenfalls in anderen Sparten entwickelte

Innovationen hierbei in Betracht zu ziehen. Aus diesem Grunde ist es zum Zeitpunkt der Fertigstellung dieser Arbeit nicht möglich, nähere Informationen über die Schutzwürdigkeit dieses Entwurfs zu geben.
Die Umsetzung des Modells in das 3D-Format erfolgte im 3ds Max. Da ich keinerlei Erfahrungen im Bereich der 3D-Modellierung hatte, war es anfänglich relativ schwierig, passable Ergebnisse zu erhalten. Es war zeitintensiv sich in ein so mächtiges Entwicklungstool einzuarbeiten und ein Modell zu erstellen.
3ds Max wird vielen Ansprüchen gerecht und nach einer gewissen Einarbeitungszeit sind auch passable Ergebnisse zu erzielen.

Der Export der 3D-Formate ins o2c-Modell und das Shout3D-Applet stellte eine weitere Herausforderung dar, da viele vordefinierte Einstellungen, Animationen, Texturen usw. nicht dargestellt werden konnten. Somit war in diesem Bereich ebenfalls ein erheblicher Nachbearbeitungsaufwand notwendig.
Die Integration der Modelle in eine HTML-Seite, sowie die Erstellung der Navigation war der nächste Schritt, was aber durch die umfangreiche Dokumentation der Shout3D- und o2c-Software erheblich erleichtert wurde. Ein Problem war die Umsetzung des stufenlosen Schaltvorgangs in Echtzeit innerhalb der Webseite. Hierbei hatte ich etliche Zeit zur Realisierung aufgebracht.
Die Arbeit hat mir sehr viel Freude bereitet, da es mir ermöglicht wurde, meine eigenen Ideen und Interpretationen mit einzubringen. Etwas verwunderlich jedoch war die Erkenntnis, dass die bestehenden Web3D-Formate zum Teil sehr wenig verbreitet sind und teilweise auch nicht mehr weiterentwickelt werden. Bestehende Entwicklungen lassen seit einiger Zeit auf die Verabschiedung eines Standards warten und daher ist die zukünftige Entwicklung in diesem Bereich nur sehr schwer einzuschätzen.
Die stagnierende Entwicklung im Web3D-Bereich, sowie die eher unwahrscheinliche Umsetzung der Fahrradschaltung in die Realität, taten meiner Motivation keinen Abbruch. Der Spaß an der Umsetzung, die kompetente Unterstützung, sowie die Sammlung an neuen Erfahrungen war der Aufwand in jeglicher Hinsicht wert.

ANHANG

A.) Abkürzungsverzeichnis

[API]	Application Programming Interface
[ASCII]	American Standard Code for Information Interchange
[ATM]	Asynchronous Transfer Mode
[BIFS]	Binary Format for Scenes
[BWG]	Browser Working Group
[CAD]	Computer Aided Design
[COM]	Component Object Model
[CPU]	Central Processing Unit
[DivX]	Digital Video Express
[DOM]	Document Object Model
[DTD]	Document Type Definition
[DXF]	Digital Exchange Format
[EAI]	External Authoring Interface
[GIF]	Graphics Interchange Format
[GLU]	Graphics Library Utility
[GLUT]	Graphics Library Utility Toolkit
[GLX]	Graphics Library Windows X Extension
[GUI]	Graphical User Interface
[HMD]	Head Mounted Device
[html]	Hypertext Markup Language
[http]	Hypertext Transfer Protocol
[IEC]	International Electrotechnical Commission
[ISO]	International Organisation for Standardization
[ISSE]	Internet Streaming SIMD Extensions
[JPEG]	Joint Photographic Experts Group
[LOD]	Level of Detail
[MIME]	Multipurpose Internet Mail Extensions

[MPEG]	Moving Picture Experts Group
[MRM]	Multi Resolution Mesh
[NURBS]	Non-Uniform Rational B-Spline und Surfaces
[o2c]	Objects to see
[OLE]	Object Linking and Embedding
[PNG]	Portable Network Graphics
[PPS]	Pictures per Second
[RTP]	Real Time Protocol
[SAI]	Scene Authoring Interface
[SGI]	Silicon Graphics Incorporated
[SDS]	Subdivision Surfaces
[SWF]	Shockwave Flash
[SMIL]	Synchronized Multimedia Integration Language
[TCP/IP]	Transmission Control Protocol/Internet Protocol
[TGS]	Template Graphics Software
[URL]	Uniform Resource Locator
[UTF]	Unicode Transformation Format
[VR]	Virtual Reality
[VRML]	Virtual Reality Modeling Language
[W3C]	World Wide Web Consortium
[WGL]	Microsoft Windows Graphics Library Extension
[X3D]	Extensible 3D
[XML]	Extensible Markup Language
[XMT]	Extensible MPEG-4 Textual
[XSL]	Extensible Stylesheet Language

B.) Glossar

[3Dnow!]	Erweiterung der x86-Prozessorfamilie von AMD, die deutlich gesteigerte Gleitkomma-Rechenleistung für dreidimensionale Grafik und Multimedia Anwendungen bietet
[ATM]	(auch als "Cell Relay" oder "statisches Zeitmultiplexverfahren" bezeichnet) ist eine paketvermittelte Übertragungstechnik, mit der sich Sprache und Video in Echtzeit übertragen lassen.
[Augmented Reality]	Erweiterte Realität Unter Augmented Reality ist die Anreicherung der realen Welt mit virtuellen Elementen zu verstehen. Dabei werden zusätzliche Informationen in das Sichtfeld des Anwenders eingeblendet, welche die realen Objekte überlagern.
[bones]	Dienen zur Animation komplexer Objekte mit Hilfe simpler Skelette (Bones – engl. für Knochen). Am Skelett lassen sich die gewünschten Posen wie an einer Gliederpuppe einstellen – unterstützt durch inverse Kinematik
[DirectX]	Multimedia-Schnittstelle für Windows
[Discreet]	3D Studio Max http://www.discreet.com AutoDesk Inc.
[Entropiecodierung]	Verlustfreies Komprimierungsverfahren durch Unterdrückung wiederkehrender Zeichen, Unterdrückung von Nullen, Lauflängencodierung, statistische Kodierung, Musterersetzung
[Extrusion]	„Ziehen“ einer zweidimensionalen Geometrie durch den dreidimensionalen Raum
[H-323]	H.323 ist ein Protokoll, das die Echtzeitübertragung von Audio- und Videodaten über Netzwerke standardisiert und von einer ganzen Reihe von Programmen unterstützt wird.
[Huffmann codierung]	ist die Generalisierung der statistischen Codierung. Sie wird z.B. auch auf Stand- und bewegte Bilder angewandt. Für jedes zu komprimierende Objekt wird ein optimaler Code ermittelt, der in einem eigenen Codebuch gespeichert wird. Dieses Codebuch wird für die Dekomprimierung benötigt und muss daher gemeinsam mit den komprimierten Daten an das Ziel übermittelt werden.
[inverse Kinematik]	Bei einem menschlichen Charakter sind die Finger, Arme und Beine aus unzähligen Gliedern zusammengesetzt. Um eine Pose zu animieren, müsste man alle Körperglieder einzeln in die gewünschte Lage bringen. Durch inverse Kinematik wird dies vereinfacht. Mit ihrer

	Hilfe verhält sich das Modell wie eine Gliederpuppe. Es genügt z.B. an einer Fingerkuppe zu ziehen, um den Finger, oder sogar den ganzen Arm, zu strecken.
[Java-Applet]	"Applet" ist abgeleitet von engl. "application", dt.: Anwendung Ein Java-Applet ist ein kleines Programm, das von einer Web-Seite aus gestartet und von einem Server heruntergeladen wird.
[Java3D]	Java3D ist eine Erweiterung von Java um Klassenbibliotheken speziell für die Anwendungsprogrammierung im 3D-Grafikbereich
[LZW-Algorithmus]	Spezieller Algorithmus zur Komprimierung von Daten. Wird unter anderem beim GIF-Format eingesetzt.
[NURBS]	(Non-Uniform Rational B-Spline und Surfaces) sind mathematische Kurven oder Flächen, die beliebige Formen von einfachen 2D Linien, Bögen oder Rechtecken bis zu organischen 3D-Freiformflächen und Volumenkörper darstellen können
[Open Inventor Format]	Das Open Inventor Format wurde speziell für interaktive Anwendungen entwickelt und enthält daher neben Klassen, die Grafikprimitiven entsprechen, auch noch Tools, die vor allem die interaktive Arbeit mit 3D-Szenen erleichtern.
[OpenGL]	Die "Open Graphics Library" (OpenGL) ist die standardisierte Software-Schnittstelle (API) der Software-Industrie zur Erzeugung von 2- und 3-D-Grafiken.
[PlugIn]	Erweiterung für Browser zur Darstellung von nicht im HTML-Standard vorgesehenen Daten
[Portabilität]	Gewährleistet die Lauffähigkeit von Anwendungssoftware auf verschiedenen Systemen
[Proprietär]	Mit Proprietär bezeichnet man Software, Protokolle oder Systeme, die in privatem Besitz stehen und deren Verbreitung und Spezifikation Copyrighteinschränkungen unterliegen
[Rendering]	Fasst alle bei der Umsetzung der Szenendaten zum Bild anfallenden Rechenvorgänge zusammen
[Shading]	Schattierung von gekrümmten Flächen. Die drei wichtigsten 3D-Shading Methoden sind Flat-Shading, Gouraud-Shading, Phong-Shading. Sie unterscheiden sich darin, wie genau die Farbverläufe innerhalb der einzelnen Polygone dargestellt werden
[Streaming]	ist eine Bezeichnung für Multimedia-Dateien wie Sound- oder Videofiles, die nicht vollständig abgespeichert werden müssen, um auf dem Rechner abgespielt zu werden. Vielmehr kann das Abspielen gleichzeitig mit dem Download der Datei erfolgen.
[Szenegraph]	Hierarchisch angeordnete, baumähnliche Struktur

[Texture Mapping]	Überlagerung einer Fläche mit einem Muster (Textur) Auch ein Video kann als Textur benutzt werden
[UTF-8]	UTF-8 ist die Codierung eines alphanumerischen Zeichensatzes im Browser. Dabei steht UCS für "Universal Character Set", ein 32-Bit-Zeichensatz, mit dem über 2 Billionen Zeichen dargestellt werden können. Die Ziffer 8 bezeichnet jedoch, dass bei diesem Kodierungsverfahren jedes Zeichen mit nur 8 Bit dargestellt wird.
[Vertex]	Knotenpunkt, Scheitelpunkt, Referenzpunkt Dieses geometrische Element verbindet die Kanten (Edges) der Gittermodelle
[Web3D]	Bezeichnet alle offenen und auch Proprietären Technologien, die interaktive 3D-Grafiken im WWW darstellen können

C.) Quellenangaben

C-1 Literatur

[Amma97] Ammann, Eckard, Programmierung animierter Welten: Java, JavaScript und VRML, Bonn: Internat. Thomson Publ., 1997

[Barg98] Bargen Bradley, Donnely, Terence Peter, Inside DirectX, Microsoft Press, 1998

[Behm00] Behme Henning, Stefan Minert, XML in der Praxis: Professionelles Web-Publishing mit der Extensible Markup Language, Addison Wesley, 2000

[Bied02] Biedorf Thomas, Leske Christophe, Regina Müller, 3D Programmierung mit Director, Galileo Press GmbH, 2002

[Birb01] Birbeck Marc, Professional XML, Birmingham: Wrox, 2001: graph. Darst. (Programmer to Programmer), 2001

[Bowm99] Doug A. Bowman, Larry Hodges, Formalizing the Design, Evaluation and Application of Interaction Techniques for Immersive Virtual Environments, Journal of Visual Languages and Computing 10, 1999

[Broll98] Wolfgang Broll, Ein Objektorientiertes Interaktionsmodell zur Unterstützung verteilter virtueller Umgebungen: GMD –Research Series, 1998 GMD-Forschungszentrum Informationszentrum GmbH

[Cohe99] Daniel Cohen-Or, David Levin, Offir Remez, Progressive Compression of Arbitrary Triangular Meshes, IEEE Visualization, 1999

[Daes02] Däßler Rolf, VRML-3D Welten im Internet, bhv Verlag, 2002

[Diehl01] Diehl Stephan, Distributed virtual worlds: foundations and implementation techniques using VRML, Java and Corba, Berlin, Heidelberg, Sprinter Verlag, 2001

[Domi97] Domik Gitta, Computer graphic Skript, Universität-Gesamthochschule Paderborn, 1997.

[Emme00] Emmel Lutz, Java3D – Der Grundkurs, Verlag Harry Deutsch, 2000

[Enge01] Engel Wolfgang, Lamothe André, Geva Amir, Beginning Direct3D Game Programming, Premier Pr, 2001

[Fey01] Jürgen Fey, XML nach Schema: W3C präsentiert Nachfolger der XML-Dokumententyp-Definition, c't 11/01, S. 102

[Goet02] Götz Frank, 3D Grafik im Web, Diplomarbeit für den integrierten Studiengang Informatik, 2002

[Gold99] Goldfarb Charles F., Paul Prescod, XML Handbuch, München: Prentice Hall, 1999

[Gros02] Gross Phil, Gross Mike, Macromedia Director 8.5

Shockwave Studio für 3D, Das offizielle Trainingsbuch, Macromedia Press, 2002

[Gumh98] Stefan Gumhold, Real time compression of triangle mesh connectivity. In Siggraph98 Conference Proceedings, 1998.

[Hand97] Chris Hand, A Survey of 3D Interaction Techniques, The Eurographics Association, 1997

[Kloss97] Kloss Jörg, Robert Rockwell, Kornél Szabó, Martin Duchrow, VRML97- Der neue Standard für interaktive 3D-Welten im Wold Wide Web, Addison Wesley Longman Verlag, 1997

[Krue00] Krüger Guido, Java2-Handbuch der Java Programmierung, Addison-Wesley, 2000

[LeaR96] Lea Rodger, Java for 3D and VRML Worlds, Indianapolis, New Riders, 1996

[Mats96] Stephen N. Matsuba, Bernie Roehl, VRML Das Kompendium, Markt und Technik Verlag München, 1996

[Maye02] Mayer Katja, Extended 3D (X3D), Diplomarbeit, Studiengang technische Redaktion an der FH Karlsruhe, 2002

[Max01] 3D Studio Max Referenzhandbuch Band I, Autodesk, Inc, 1999

[Max02] 3D Studio Max Referenzhandbuch Band II, Autodesk, Inc, 1999

[Max02] 3D Studio Max Lehrgänge, Autodesk, Inc, 1999

[Meis01] Klaus Meißner, Raimund Dachselt, 3D Grafikformate: Modellierung, Animation, Interaktion, Technische Universität Fakultät Informatik, Lehrstuhl Multimediatechnik, 2001

[Mild95] Torsten Milde, Video-Kompressions-Verfahren im Vergleich, dpunkt Verlag, 1995

[Musc01] Musicano Chuck, Bill Kennedy, HTML & XHTML, O'Reilly, 2001

[o2cC03] o2c Composer Private, MB Software AG, 2003

[Olbr00] Stephan Olbrich, Ein leistungsfähiges System zur Online-Präsentation von Sequenzen komplexer virtueller 3D-Szenen, Hannover, Universität, Fachbereich Elektrotechnik und Informationstechnik, 2000

[Pole99] Polevoi Rob, Interactive Web Graphics with Shout3D, Sybex, 1999

[Ross99] Jarek Rossignac, Edgebreaker, Connectivity compression for triangle meshes, IEEE Transactions on Visualization and Computer Graphics, 5(1), 1999.

[Rukz02] Enrico Rukzio, Formate, Technologien und Architekturkonzepte für 3D-Web-Applikationen, Belegarbeit an der TU Dresden, 08.2002

[Shre00] Shreiner, Dave, OpenGL Reference Manual – The Official

Reference Document to OpenGL, OpenGL Architecture Review Board, Addison Wesley Publishing Co., 2000

[Sowi98] Henry Sowizral, Rushforth Kevin, Deering Michael, The Java3D API Specification, Sun Microsystems Inc., Addison Wesley Publishing Co., 1998

[Taub98] Gabriel Taubin, Jarek Rossignac, Geometric Compression Through Topological Surgery, ACM Transactions on Graphics, 1998

[Toum98] Costa Touma, Craig Gotsman, Triangle Mesh Compression, Proc. of Graphics Interface, 1998.

[Tyma98] Tyma, P., why are we using Java again?, Communications of the ACM, Volume 41-Number 6, 1998.

[Velho99] Velho Luiz, Luiz Henrique de Figueiredo, Jonas Gomez, Hierarchal generalized triangle strips, The Visual Computer, Springer Verlag, 1999

[Vels02] Istvan Velsz, 3ds max 4, Grundlagen und Praxis der 3D Visualisierung und –Animation, Addison Wesley, 2002

[Wals01] Aaron E. Wals / Mikael Bourges-Sevenier, core Web3D, Prentice-Hall PTR London, 2001

[WooM99] Woo, Mason, Neider, Jackie, Davis Tom, Shreiner Dave, OpenGL Programming Guide – The Official Guide to Learning OpenGL, OpenGL Architecture Review Board, Addison Wesley Publishing Co., 1999

C-2 Web-Seiten

[3DLabs] 3dLAbs
OpenGL 2.0
http://www.3dlabs.com
3dLabs Inc. Ltd,

[3ds Max] 3D Studio Max von Discreet
http://www2.discreet.com

[Adobe] Adobe
http://www.adobe.com/
Adobe Systems Inc.

[Apple] Apple
http://www.apple.com
Apple Computer Inc.

[Arbo03] ARB, OpenGL Architecture Review Board, http://www.opengl.org/developers/about/arb.html, 2003

[ATI] ATI
http://www.ati.com
ATI Technologies Inc.

[BalsAl] Alexandra Balschun, Das Sonnensystem- erstellt mit Macromedia Director,

	http://asi-www.informatik.uni-hamburg.de/themen/mi/lehre/mi1/skripte/Director/planeten.htm
[Beyond 3D Extreme]	Beyond 3D Extreme http://www.beyond-3d.com/beyond/
[blaxxun]	Blaxxun interactive http://www.blaxxun.de
[Breu01]	Dominik Breuer, Triangle Mesh Compression, www.uni-koblenz.de/~cg/veranst/ws0001/sem/Breuer.pdf, 2001
[Cortona]	Cortona von Parallel Graphics http://www.parallelgraphics.com
[CosmoPlayer]	Cosmo Player von Cosmo Software http://www.cai.com/cosmo/
[Cybe03]	Das Cybernarium: Erlebnispark für virtuelle und erweiterte Realität, Frauenhofer Institut für graphische Datenverarbeitung, Abruf in 02/03, http://www.cybernarium.de
[Dell]	Dell http://www.dell.com/us/en/gen/corporate/general_news_news.htm Dell Computer Corporation
[Evans & Sutherland]	Evans & Sutherland http://www.es.com/
[Foru01]	Forum3D, OpenGL und Direct3D, http://www.forum-3d.de/tech/opengl.shtml 2001
[GoalaJ]	Joachim Goala, Director 8: Die Weborientierten Neuerungen, http://www.directorworkshop.de/data/workshop/d8_shock.html
[HowStu]	How Stuff Works – How Shockwave 3D Technology works http://www.howstuffworks.com/shockwave.htm
[Hewlett Packard]	Hewlett Packard http://www.hp.com/ Hewlett Packard Company
[IBM]	IBM http://www.ibm.com International Business Machines Corporation
[IICM]	IICM Hyperware http://www.iicm.edu/ Institution for Information processing and Computer supported new Media, Austria
[Inte02]	Macromedia Director 8.5 Shockwave Studio with Intel Internet 3D Graphics Software, Powerful, Scalable 3D for the Web Mainstream, Intel Corporation, http://www.intel.com/ial/3dsoftware/doc.htm, 2001

[Intel] Intel
http://www.intel.com/
Intel Corporation

[Internet Space Builder] Parallel Graphics
http://www.parallelgraphics.com

[Java03] Java3D API
http://java.sun.com/products/java-media/3D/
Sun Microsystems Inc., 2003

[Junk00] Junkins Stephen, Allen Hux, Subdivision Reality / Employing Subdivision Surfaces for Real Time Salale 3D, Intel Architecture Labs, Intel Corporation,
http://www.intel.com/ial/3dsoftware/doc.htm
2000

[Kaup01] A. Kaup, MPEG Standards Techniken und Entwicklungstrends,
http://www.lnt.de/~kaup/paper/fkt-2001.pdf
06/2001

[Koen01] Rob Koenen, Overview of the MPEG-4 Standard,
http://www.cselt.it/mpeg/standards/mpeg-4/mpeg-4.htm
2001

[Macromedia] Macromedia
http://www.macromedia.com/
Macromedia Inc.

[Matrox] Matrox
http://www.matrox.com/mga/deutsch/home.cfm
Matrox Graphics Inc.

[Maya] Maya Alias/ Wavefront
http://www.aliaswavefront.com

[Microsoft Corporation] Microsoft
http://www.microsoft.com
Microsoft Corporation

[MPEG4_Dev] MPEG-4 Tools von ENST (Telekom Paris)
http://www.comelec.enst.fr/~dufourd/mpeg-4/

[MPEG4_Toolbox] MPEG-4 Toolbox
http://uranus.ee.auth.gr/pened99/Demos/Authoring_Tool/auhoring_tool.html

[MPTU01] Tutorial Issue on the MPEG-4 Standard, Image Communications Journal,
http://www.cselt.it/leonardo/icjfiles/mpeg-4_si/
2001

[nexternet] Nexternet
http://www.nexternet.com

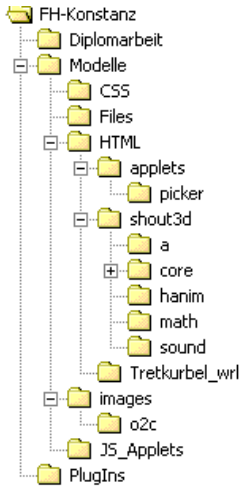
[Nvidia] Nvidia
http://www.nvidia.com/
Nvidia Corporation

[o2c03] o2c GmbH, o2c-Produkte
http://www.o2c.de ,
2003

[OpenWorlds]	Open Worlds http://www.openworlds.com
[ParallelGraphics]	Parallel Graphics http://www.parallelgraphics.com
[Schm01]	Inka-Gabriela Schmidt/dc, Shockwave oder Flash?, http://www.macwelt.de/_news/detail/director85.shtml 26.04.2001
[ScorMe]	X3D-File Converter/ Importer von Score Multimedia http://www.score.de
[SGI02]	SGI, Datasheet-OpenGL The Industry's Foundation for High Performance Graphics http://www.sgi.com/software/opengl/,
[shout3D]	Shout3D http://www.eyematic.com/products_shout3d.html
[Spazz3D]	Spazz 3D von Virtock http://spazz3d.com
[Sun]	Sun http://www.sun.com Sun Microsystems Inc.
[TechNo]	Macromedia Director Support Center TechNote, How can a Shockwave movie pass information to and from a browser? http://www.macromedia.com/support/
[TuIs01]	Tutorial Issue on the MPEG-4 Standard, Image Communication Journal, 2001 http://www.eselt.it/leonardo/icjfiles/mpeg-4_si/
[VRML97]	The Virtual Reality Modeling Language International Standard ISO/IEC 14772-1, 1997 http://www.web3d.org/fs_specifications.htm
[Web3D Konsortium]	Web3D Konsortium www.web3d.org
[X3D03]	Extensible 3D Working Group http://www.web3d.org/x3d/
[XJ3D]	XJ3D http://hypermultimedia.com/XJ3D/

D.) Abbildungsverzeichnis / Programmlisting / Tabellen

E.) CD Rom



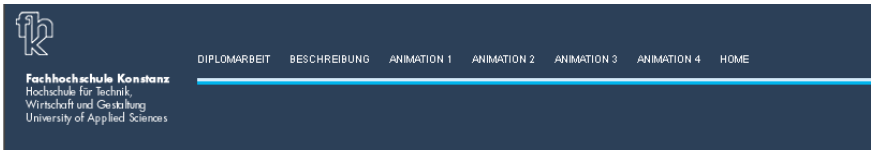
Nebenstehende Abbildung zeigt die Ordnerstruktur der CD-Rom.

Diplomarbeit: Dieser Ordner enthält die schriftliche Ausarbeitung im *.pdf Format

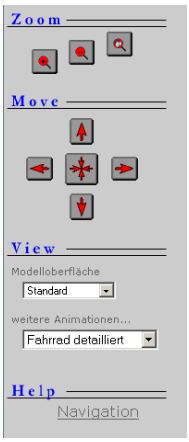
Modelle: Enthält sämtliche, zur Darstellung der Modelle erforderlichen, Dateien

PlugIns: In diesem Ordner sind die jeweils erforderlichen o2c-Player abgelegt

Die einzelnen Ordner und deren Inhalt können aber auch über den `index.html`-File aufgerufen werden, da er eine Navigationsstruktur des gesamten Inhaltes enthält:



Hier sind die Links zu der schriftlichen Ausarbeitung, die Beschreibung der Fahrradschaltung sowie die einzelnen o2c- und shout3D Modelle zu finden.



Eine Navigationshilfe zu den Modellen kann nach Aufruf der 3D-Szenen innerhalb der rechten Navigationsleiste unter „Help“ aufgerufen werden.

Die Rückkehr zur jeweiligen Animation erfolgt durch Betätigung eines der Navigationsbuttons innerhalb der dargestellten Navigationsleiste.

F.) Ehrenwörtliche Erklärung

Hiermit erkläre ich, CHRISTIAN BRAUN, geboren am 13.01.1975 in Engen, ehrenwörtlich,

(1) dass ich meine Diplomarbeit mit dem Titel:
„**Stand der Technik der Virtual Reality-Technologie im Web – dargestellt am Beispiel einer neuartigen Fahrradschaltung“**
an der FH Konstanz unter Anleitung von Herr Professor Ullrich Hedtstück selbständig und ohne fremde Hilfe angefertigt habe und keine anderen als in der Abhandlung angeführten Hilfen benutzt habe;

(2) dass ich die Übernahme wörtlicher Zitate aus der Literatur sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Konstanz, den 02.05.2003 [**Christian Braun**]