Szenengraphen

Proseminar Virtuelle Präsenz

Universität Ulm
Sommersemester 2005

Codruța Cosma
e-mail: cc6@informatik.uni-ulm.de

Inhaltsverzeichnis

1. Allgemeines

Definition:

„Ein **Szenegraph** bzw. **Szenengraf** ist eine Datenstruktur für 3D-Szenen, er beinhaltet die Gesamtheit aller Elemente einer Szene. Der Szenegraph ist aus graphentheoretischer Sicht ein Baum, dessen Wurzel z. B. das Universum sein kann, in dem sich die Szene befindet. Dieser Wurzel untergeordnet sind alle Objekte der Szene. Diese Objekte können wiederum Wurzel eines weiteren Baumes (also einer weiteren Hierarchie von Objekten) sein“. (*www.wikipedia.org*)

Auf dieser Weise kann man in der Computergraphik Objekte hierarchisch anordnen. Jedem Objekt eines Szenegraphen wird eine Transformationsmatrix zugeordnet. Das heißt, dass bei dem Zugriff auf dieser Matrix auch alle untergeordneten Objekte mit transformiert werden. Man unterscheidet also zwischen Objektkoordinaten (Koordinaten eines Objektes bezüglich des übergeordneten Objektes) und Weltkoordinaten (Koordinaten eines Objektes bezüglich der Wurzel des Szenengraphen, z.B. der Ursprung des Universums).

Beispiel (*www.wikipedia.org*):

Ein Auto hat vier Räder. Man hat also ein Objekt Auto und diesem Objekt werden die vier Räder untergeordnet. Wird die Position oder der Winkel des Autos manipuliert, so wirkt sich die Manipulation auch auf alle Unterobjekte (in diesem Fall die Räder) aus. Man muss nicht manuell die Position der Räder neu berechnen.

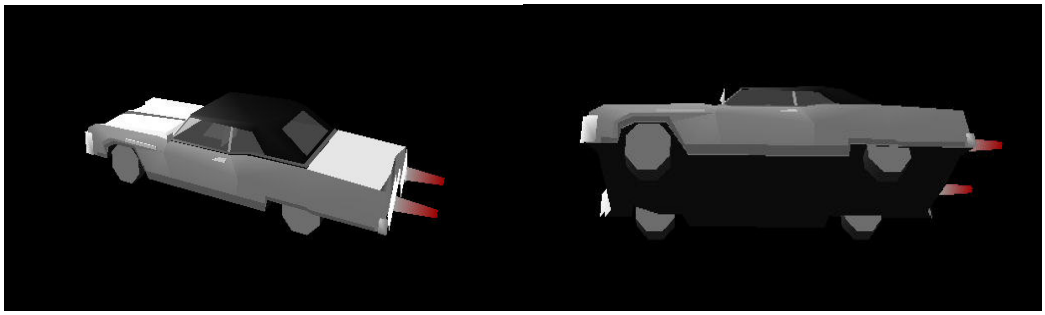

Abb.1 VRML-Auto

Bei der Drehung des Autos um die drei Achsen bewegen sich alle untergeordneten graphischen Komponenten mit. Eine Neuberechnung der Koordinaten der Unterobjekte (Räder, Lichter, usw.) ist nicht nötig.

Die virtuelle Realität stellt ein komplettes Model der 3D-Welt im Computer dar, eine perspektivische und „photorealistische“ Darstellung mit Elimination verdeckter Objekte und Lichteffekte, Texturen etc.

Abb.2 3D-Terasse mit Farben und Texturen

In dieser virtuellen Welt wird dem Benutzer ermöglicht mit Maus und Tastatur zu navigieren. Es erfolgt eine sehr schnelle Bewegung durch die Szene mit fast sofortiger Reaktion auf Benutzereingaben. Dabei bewegt sich dieser in einer Welt mir Stereo Sound in der man durch Stereovision in Zukunft auch eine Tiefenwirkung erzielen möchte. Was man ebenfalls für die Zukunft in Planung hat ist eine „natürliche Navigation“ zu ermöglichen, durch sogenanntes „Umsehen“ mithilfe eines Head Mounted Display’s (HMD) oder Gesten mithilfe eines Datenhandschuhs.

Zur Realisierung einer virtuellen Welt im Internet wird im folgenden die Graphiksprache VRML vorgestellt.

2. Graphiksprache VRML (Virtual Reality Modelling Language)

VRML ist eine Graphik(programmier)sprache zur Modellierung von 3D Objekten und Szenen , der virtuellen Realität . Eine VRML-Datei stellt einen ASCII Text dar und wird über das Internet auf den lokalen PC übertragen. Die Interpretation des VRML-files und die Erzeugung des von ihm beschriebenen Bildes erfolgt dann durch den Browser auf dem lokalen PC.

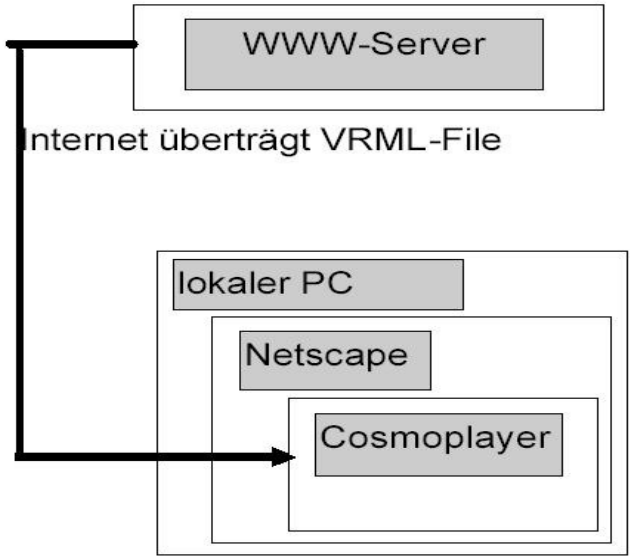

Abb.3 Übertragung und Darstellung des VRML-files

Dazu werden oft sogenannte VRML-Player benötigt, die den VRML-Code kompilieren und ausführen und das erzeugte Bild dann im Browser darstellen. Die Szene wird in verschiedenen Detaillierungsstufen dargestellt , dazu werden Methoden wie Flat-Shading , Gouraud-Shading oder das Drahtgitter-Modell benutzt. „Die Bezeichnung Flat- oder Quick-Shading kommt aus dem Englischen. Das Flat-Shading ist die einfachste Form des Shadings. Die einzelnen Facetten der gekrümmten Flächen werden in einem, je nach Lichteinfall durchschnittlich ermittelten Farbton dargestellt (kein kontinuierlicher Farbverlauf)“. Jede Oberfläche erscheint deshalb flach und matt, weil Lichtreflexe, Schatten, Transparenz usw. unberücksichtigt bleiben. „Das Gouraud-Shading, ein 1971 von Henri Gouraud vorgestelltes Verfahren, interpoliert die Eck-Farbwerte. Damit werden zwischen den Polygonflächen weiche Farbverläufe, d.h. eine gleichmäßige Schattierung erzeugt. Das Gouraud-Shading vermag lediglich matte Objektoberflächen darzustellen, die das Licht gleichmäßig und ungeordnet in alle Richtungen streuen. Folglich erhalten die Objekte ein plastikähnliches Aussehen (Transparenz, Schatten, Spiegelungen, Materialeigenschaften wie Texturen usw. werden nicht berücksichtigt)“ (*www.glossar.de*). Dabei ist es möglich mithilfe der Maus und der Tastatur in diesem Bild zu navigieren, Objekte zu bewegen oder zu drehen und ihren Platz in der Szene zu verändern (walking, flying, pointing, looking).

Der Browser ermöglicht die Erkennung von Kollisionen und die Beleuchtung der Szene erfolgt durch einen „Frontscheinwerfer“. Es existieren auch verschiedene definierte Viewpoints die der besseren Orientierung dienen.
VRML-Browser sind im Netscape Communicator und Internet Explorer bereits integriert. Es existieren auch verschiedene Plug-ins und sogenannte Stand-Alone-Browser, jedoch unterstützen nicht alle Browser die Features von VRML.

VRML-Beispiel:

```
# VRML V2.0 utf8
# Schneemann mit blauen Augen und roter Nase
# Kopf
Shape {
geometry Sphere { radius 1 }
appearance Appearance {
material Material { diffuseColor 1 1 1 }
        }
}
# linkes Auge (30° von der Nase)
Transform {
translation .5 0 .867
children Shape {
geometry Sphere { radius .2 }
appearance Appearance {
material Material {diffuseColor 0 0 1 }
                }
        }
}
# rechtes Auge (30° von der Nase) dto.
# Nase
Transform {
rotation 1 0 0 1.58 # Drehung um x-Achse um 90°
translation 0 0 1
children Shape {
geometry Cone {
        bottomRadius .5
        height 2
        }
appearance Appearance {
        material Material {diffuseColor 1 0 0}
                }
}
```

Und das Resultat :

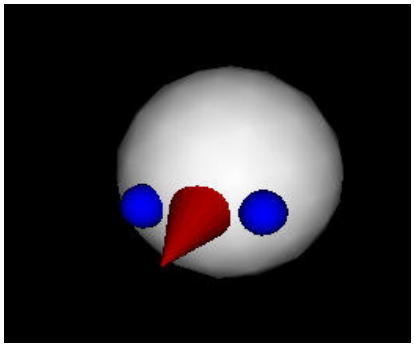

Abb.4 Schneemann

VRML baut also Modelle bzw. Szenengraphen basierend auf einer hierarchischen Anordnung von Geometrieknoten und deren korrekte Positionierung in Objektkoordinaten oder Weltkoordinaten.
Die Transformationen werden hierbei hintereinander ausgeführt , "von innen nach außen" was dazu beiträgt dass eine Gruppe geschachtelter Knoten bzw. Objekte nicht „auseinander fällt“. Das gleiche Paradigma gilt auch für Java3D und OpenSzeneGraph.
VRML bietet noch Features wie Reflexion, Texturen, Farbmodell, verschieden Beleuchtungsknoten wie z.B. ambientes Licht (als Anteil in den Lichtquellen definierbar), direktionales (paralleles) Licht, Punktlicht, oder Spotlight an. Auch kann man einen Teil einer Szene durch Angabe der URL aus dem web nachladen und Ankerknoten erstellen, das heißt die Kinder dieser Knoten werden zu „klickbaren“ Knoten bzw. Objekten. Interaktivität in der Szene wird durch Benutzung von Sensoren wie z.B. dem TouchSensor (wird eingeschaltet wenn über ein Objekt mit der Maus „berührt“ wird) , ProximitySensor (wird eingeschaltet wenn sich die Maus einem Objekt nähert) oder dem VisibiltySensor (wird eingeschaltet wenn ein bestimmtes Objekt in der Szene sichtbar ist) bereitgestellt, was zu einer event-orientierten virtuellen Welt führt. Zur Animation stellt VRML Interpolatoren wie dem PositionInterpolator oder dem OrientationInterpolator bereit. Auch verfügt VRML über einen TimeSensor mit dem man absolute oder relative Zeit setzen kann und damit per Event insbesondere die Interpolatoren steuert.

Andere VRML
Beispiele:

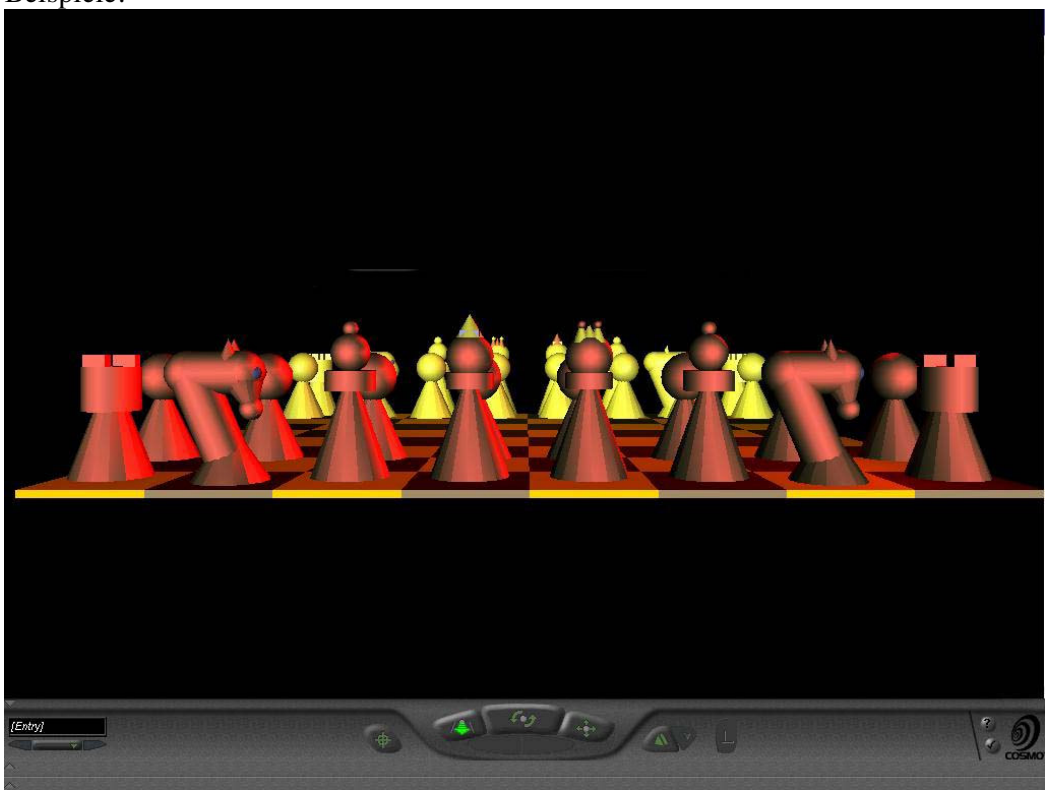

Abb.5 Komplettes Schachspiel, Figuren auf dem Schachbrett beweglich

3. Andere Graphiksprachen

3.1. OpenSceneGraph

„Der OpenSceneGraph ist eine open-source portable Plattform zur Erstellung von hochperfomanten Graphikapplikationen so wie Flugsimulatoren, virtuelle Realität, Spiele und wissenschaftliche Visualisierung“ *(www.openscenegraph.org)*. Basiert auf Szenengraphen , stellt OpenSceneGraph ein objektorientiertes System dar, aufgebaut auf OpenGL(der ersten Bezugs-Plattform zur Erstellung von 2D und 3D Applikationen) mit zahlreichen Hilfsmitteln und Features zur schnellen Entwicklung von graphischen Anwendungen.

Der Built-in Editor osgedit ermöglicht die Bearbeitung von mehreren Szenengraph features. Mit dem OsgVortex Physik Plug-in wird die Erstellung von Szenengraphen mit der Anwendung der Basis Physik, Kollisionenbehandlung und rigide Körper ermöglicht.
Analog zu dem Cosmoplayer für VRML können OpenSceneGraph Applikationen mithilfe des osgviewer’s abgespielt werden.

Die Beleuchtung kann Per-vertex oder anisotrop erfolgen, OpenSceneGraph bietet auch Texturen, Schatten und skinning an. Es enthält auch Bild-Lader die es ermöglichen .rgb, .gif, .jpg, .png, .tiff, .pic, .bmp und .dds Dateien in die virtuelle Welt zu laden. Die Graphiksprache stellt Plug-ins zur Verfügung zur Herausfilterung von Verdeckungen, Animation: Keyframe Animation (Man benutzt die Keyframe-Animation, um Elemente eines Objekts, (z. B. die Beine einer Figur) zu bewegen. Keyframes, zu deutsch Hauptphasen, bezeichnen die Endpunkte einer Bewegung, also den Schritt einer Figur.), skelettartige Animation, kombinierte Animation oder für Special Effects wie : Spiegel, Umgebungs-Mapping oder Billboarding.

Das Billboarding ist ein Verfahren in dem die Polygone in Abhängigkeit vom Ansichtspunkt orientiert werden. Verändert sich dieser, so verändert sich auch die Ausrichtung der Polygone. In Kombination mit Animation kann dieses Verfahren zur Darstellung von unsoliden Objekten so wie Feuer, Wasser, Rauch oder Wolken verwendet werden.

Terrain Rendering erfolgt via dem Demeter Terrain Engine Plug-in integriert in OpenSceneGraph, der dem Benutzer auch ermöglicht 2D -, 3D- und Streaming Sound in seine Welt einzuspielen. Rendering ist auch mithilfe der Funktion Render-to-Texture möglich.

Beispiele für einen Flugsimulator und ein Spiel:

Abb.6 und 7: Flugsimulator und Computerspiel

3.2. Java3D

Java3D ist eine Erweiterung zu Java ab Version 1.2, die es erlaubt sehr einfach 3D-Welten zu programmieren. Von dieser Schnittstelle gibt es auch eine Version für Linux. Im Gegensatz zu VRML die eher benutzt werden sollte wenn es um einfache Inhalte geht, ist Java3D besser wenn der Benutzer Operationen auf komplexe Inhalte und Applikationen in seine Welt einbringen möchte. So wie VRML baut auch Java3D auf eine Hierarchie von Knoten und Gruppen von Knoten auf, dabei stellen diese Knoten Instanzen von Java3D Klassen dar. Java3D Applikationen bauen die virtuelle Welt programmatisch auf. Es existiert kein Java3D Dateiformat, ein VRML Datei-Lader ist jedoch möglich.
Java3D und VRML teilen gleiche Konzepte wie: Formen, Design, Materiale, Gruppierung, Transformierungen und Beleuchtung. Der Übergang vom VRML zu Java3D kann sehr leicht erfolgen.
Im Vergleich zu VRML bietet Java3D jedoch komplexeren Support für: Oberflächen-Linien- und Punktgeometrie, 3D Text Geometrie, Texturen und Transparenz. Transformationen, Beleuchtung und Nebel können in Java3D auch effizienter realisiert werden. Der Sound Playback und Nachklang sind in Java3D von besserer Qualität sowie Viewing und Rendering. Im View kann durch einen Head-Tracker-Einsatz die Position und Orientierung verändert werden. Dabei erfolgt eine Aufsplittung des herkömmlichen Viewknotens in:

- Physical Environment (Input-, Sounddevices)
- Physical Body (Körpergröße, Augeneigenheiten usw.)
- Screen3D (Werte über den zu benutzenden 'Screen')
- Canvas3D (Leinwand)
- View Plattform (Position und Orientierung)
- View (Projektionsart; zentrales 'Schaltwerk')

Java3D stellt eine größere Zahl von Eingabegeräten bereit und bietet im Vergleich zu VRML bessere Auflagen für Instancing (DEF und USE auch in VRML benutzt), Hintergründe, Interpolatoren, Behaviors, Kollisionenbehandlung und Clipping(Verfahren, bei dem bestimmte Teile eines geometrischen Modells für die Darstellung am Bildschirm abgeschnitten oder als Scheibe (3D-Clipping) ausgeschnitten werden. Die abgeschnittenen Teile sind lediglich verborgen, also nicht gelöscht und können jederzeit wieder aktiviert werden.).
Was es in Java3D jedoch im Gegensatz zu VRML nicht gibt sind vordefinierte Formen der primitiven Geometrie (Zylinder, Kugel, Kegel, Würfel), Elemente der speziellen Geometrie wie Elevation grids oder Extrusion(Vorgang, durch den ein dreidimensionaler Volumenkörper durch das Auseinanderziehen einer zweidimensionalen Fläche (sweepen) entlang eines linearen Pfades erstellt wird.), Prototypen oder Routen. Routen sind eine einfache Art zur Definierung einer Bahn zwischen einem von einem Knoten generierten Ereignis und einem Knoten der ein Ereignis empfängt.

Es gibt verschiedene Interpolatoren und Behaviors in Java3D. Interpolatoren sind zum Beispiel der RotationInterpolator, der ein Objekt drehen kann, der ColorInterpolator der die Farbe eines Elementes verändert u.a. Ein Behavior beschreibt das „Verhalten“ eines Eingabegerätes oder einen Objektes beim Eintritt eines gewissen Events. Zum Beispiel kann man durch Programmieren einer Behavior-Klasse bewirken dass sich ein Objekt dreht wenn man mit der Maus darauf klickt. Weitere Bespiele von Behaviors können der oberen Abbildung entnommen werden.
Durch die fortgeschritten Interpolatoren und Behaviors kann in Java3D eine viel bessere Interaktion erzielt werden als mit VRML.
Die Aktivierungsfunktion gibt an, zu welchem relativen Zeitpunkt auf welche Art zwischen diesen Werten interpoliert wird und für den Rest ist die API zuständig. Diese ist wiederum darauf optimiert, nur den Teil der Szene neu zu berechnen, der durch die Veränderung beeinflusst wird. Es wird für diese Zwecke nicht nur ein Interpolator und eine Aktivierungsfunktion, sondern auch ein Zeitgeber benötigt. Der Zeitgeber bei Java3D ist im Scheduler mit integriert.

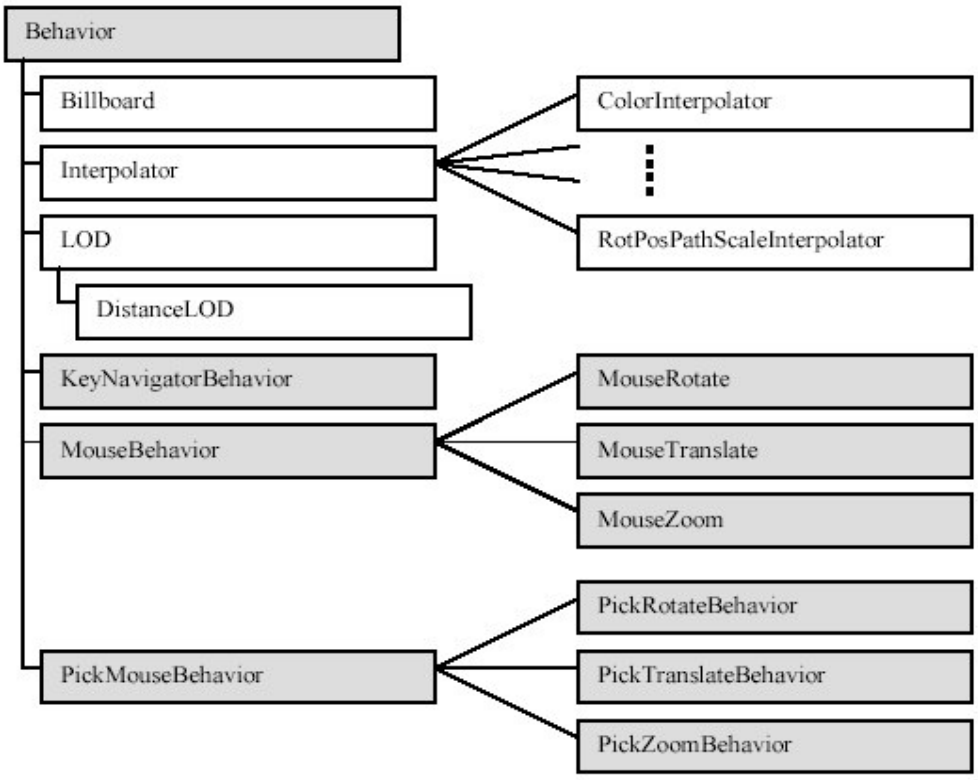

Abb.8: Erklärung von Behaviors und Interpolatoren in Java3D

Zwei weitere Verfahren die Java3D kennt sind das Morphing und das Anti-Aliasing. Mit dem Morphing-Verfahren kann man durch Überblenden einer geometrischen Struktur oder eines Bildes eine z.B. Teekanne in eine Kugel verwandeln oder mit der Mimik von Gesichtern spielen. Das Anti-Aliasing, „auch "Kantenglättung" genannt ist ein Verfahren zur Verminderung des Treppeneffekts, der durch Pixeldarstellung bei schrägen und gekrümmten Linien entsteht. Durch Interpolation, farbliche Angleichung benachbarter Bildpunkte werden die "Treppenstufen" ausgeglichen“. Dabei kann es allerdings passieren, dass die Darstellung von Linien breiter erscheint.

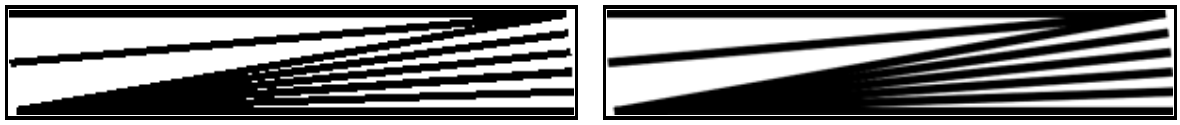

Abb. 9 Anti-Aliasing

VRML und Java3D kennen beide das LOD (Level of Detail): „Ein Objekt wird in verschiedenen Detailstufen gespeichert - z.B. für die Darstellung unterschiedlicher Detaillierungen in einer CAD-Zeichnung“. (*www.glossar.de*)

Beispiel von einem Java3D Szenengraph:

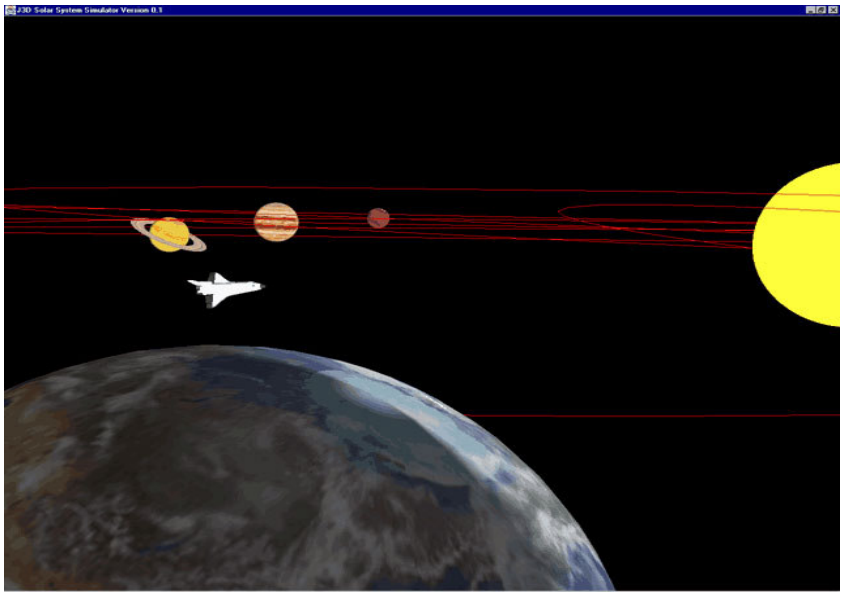

Abb. 10 Eine Solarsystemsimulation mit Java3D

Die Realisierung von Avataren kann in Java3D sehr effizient und hoch qualitativ erfolgen während man VRML eher simple und unspektakuläre „Menschen“ erstellen kann. Ein Avatar ist eine fiktive Identität, die ein Teilnehmer innerhalb virtueller Welten, zum Beispiel beim *Chatten*, annehmen kann. „Avatare sind synthetische Repräsentationen realer Menschen, werden auch natural residents genannt und gehören den Cyberian Citizens (Cyberianern) an. Ursprünglich verkörperte ein Avatar im Hinduismus einen Gott auf Erden“. (*www.glossar.de*)

4. Ein Vergleich der APIs: OpenGL, Java3D und VRML

Funktion	OpenGL	Java3D	VRML
Primitive geometrische Objekte (Punkte, Linien, Flächen)	x	x	x
Komplexere geometrische Objekte (Sphäre, Quader, ...)	x	Utils	x
Ausgefallene geometrische Objekte:			
Gebirgszüge und Extrusion (Eine Grundform, die verschieden breit ist in der Höhe)			x
komprimierte Geometrie		x	
Bezier Kurven und Flächen, Nurbs und Quadrigen	x		
Materialeigenschaften / Farben	x	x	x
Licht (direktionales, Punkt- und Spotlicht)	x	x	x
Ambientes Licht	x	x	
Nebel	schlecht	x	schlecht
Sound		sehr gut	x
Hintergrund, LOD, Billboard		x	x
2D-Text		x	x
3D-Text		x	
Morphing		x	
Anti-Aliasing	x	x	primitiv
Texturen	3D	3D	3D+movie
Wiederverwendung		x	x
Displaylist	x		
Interpolatoren (Koordinaten-, Positions-, Rotations-, Skalierungs- und Farbinterpolatoren)		x	x
weitere Interpolatoren (Transparenz-, Switch- und verschiedenste Path-Interpolatoren)		x	
Zeitgeber		x	x
Scheduler		x	
Route-Mechanismus			x
Viewpoints			x
Schnittstelle zu anderen Applikationen	x	x	x
Anchor		x	x

(*www.it-cstein.de/diplom/APIsVergleich.aspx*)

5. Zusammenfassung

Es existiert eine Vielzahl von Graphiksprachen zur Erstellung und Bearbeitung von 3D-Szenen und virtuellen Welten. Der Aufbau der Szenen und das Prinzip nachdem diese Graphiksprachen funktionieren weisen jedoch immer dieselben Eigenschaften auf: hierarchisch und objektorientiert. Jede Sprache bietet ihre eigenen Features und gleiche Features haben bei den verschiedenen Sprachen unterschiedliche Qualität. Aus diesem Grund kann man nicht behaupten dass eine Graphiksprache besser ist als die andere , die Wahl sollte projekt- und ressourcenorientiert getroffen werden.

Literatur / Quellen

www.openscenegraph.org
www.wikipedia.de
www.glossar.de
www.it-cstein.de/diplom/APIsVergleich.aspx
Vorlesung Graphische Datenverarbeitung SS03 Universität“ Transilvania“ Rumänien , Prof. Dr. Ch. Schulz FH Wiesbaden