



X3D – eine Einführung

- X3D bedeutet eXtensible 3D
- ISO zertifizierter Software Standard für interaktive Echtzeit-3D-Grafik
- Open Source Standard basierend auf XML
- Entwickelt vom web3D Consortium (bestehend aus Softwareunternehmen, freien Entwicklern und Institutionen)
- Siehe <http://www.web3d.org>



Wozu dient X3D?

- Beschreibungs-Sprache für 3D „Welten“
- Darstellung von 3D Welten mit einfachen, öffentlich verfügbaren Mitteln via Internet
- Austausch von 3D Konstrukten zwischen verschiedenen Plattformen (CAD etc.)
- Animations- und Interaktions-Möglichkeiten für alle Benutzer
- Integration mit WWW und Multimedia



Warum X3D als ISO Standard?

- Ziel: Schwächen des alten „Standards“ VRML beheben und den sich ständig ändernden Anforderungen im Internet gerecht werden
- Kernpunkte X3D:
 - Modularer Aufbau (Profile je nach Anwendung)
 - Einsatz von XML zum Datenaustausch
 - Abwärtskompatibilität zu VRML97 („Profil“)
 - Definierte API und OpenSource Umgebung



Einführung in VRML

- VRML heisst Virtual Reality Modeling Language, ein Teil von “Web3D”/”X3D”.
- VRML ist einfach ein 3D Datenaustausch-Format: Es besitzt Sprachelemente für die heute in 3D-Applikationen üblichen Elemente wie Objekte, Licht, Texturen...
- VRML ist ähnlich zu HTML: Dadurch kann man es einfach im Internet als Dokument transportieren und in einem geeigneten “Browser” anzeigen lassen.



Mehr zu VRML

- VRML liefert die Technologie, drei Dimensionen, zwei Dimensionen, Text und Multimedia in einem kohärenten Modell unterzubringen.
- Dadurch entstehen in Verbindung mit Skriptsprachen und dem Internet völlig neue Anwendungsmöglichkeiten.
- VRML war eine mögliche Basis des “Cyberspace” und von virtuellen on-line Communities.
- VRML kann per se keine echten Multiuser-Simulationen oder -Welten mit Interaktion darstellen (wie z.B. Rollenspiele etc.).



Vor-Geschichte von VRML

- 1992: SGI veröffentlicht das Iris Inventor 3D Toolkit, geschrieben in C++. Hatte bereits viele Eigenschaften des heutigen VRML.
- 1994: Mark Pesce und Tony Parisi entwickeln einen frühen Prototypen eines 3D-Browsers für das WWW. Sie riefen Interessenten auf, Vorschläge für eine formale Spezifikation von 3D im WWW einzureichen.



Versionen von VRML

- Die erste Version (VRML 1.0) lässt einige Schlüssel-Eigenschaften vermissen:
 - Keine Animation,
 - Keine Interaktion,
 - Kein “Behavior”, also Verhalten von Objekten bei Aktionen.
- 1996: In Kyoto beschliesst die ISO, VRML 2.0 als Standard zu veröffentlichen.



Noch mehr VRML

- Auf dem Client wird mit einem Plug-In (z.B. Blaxxun, Octaga oder cortona) das VRML Format interpretiert und das “Rendering” besorgt, d.h. die 3D-Darstellung.
- VRML basiert auf dem SGI Dateiformat “Open Inventor”, besitzt aber kein API.
- VRML ist nutzbar mit MPEG4 und Java3D (VRML oder X3D SceneGraphs können eingebunden werden).



Was bietet X3D Neues?

- X3D Weiterentwicklung von VRML97
- X3D nutzt aktuelle Hardwaretechnologien
- Grundgerüst wurde übernommen und erweitert, um eine höhere Flexibilität zu erreichen
- Hauptunterschiede:
 - Erweiterungen der Scene-Graph Möglichkeiten
 - Überarbeitetes und vereinheitlichtes Application Programming
 - Mehrere Datei-Kodierungsvarianten (Encoder), auch verschlüsselt
 - Modulare Architektur ermöglicht stufenweise Integration von Anwendungen
 - Erweiterte Syntax in Struktur und Spezifikation



X3D und andere Formate

- X3D (und VRML) Files können Referenzen zu Files in anderen Formaten enthalten:
 - JPEG, PNG, GIF,
 - MPEG,
 - WAV, MIDI,
 - Java, JavaScript,
 - Webseiten (URL).
- Dies sind alles unabhängige Standards, die weit verbreitet sind.



X3D Programmierung

- X3D verfügt über ein standardisiertes API (Application Programming Interface)
- Im Unterschied zu VRML gibt es internes Scripting-API und externes API (SAI)
- X3D ermöglicht die Einbindung von Script-sprachen wie JavaScript und ECMAScript
- Damit ist eine vollständige Steuerung der VR durch prozeduralen Code möglich.



X3D + HTML + Scripting/Java

- X3D in HTML einbetten:
Dies geht mit Hilfe des `<EMBED>` oder `<OBJECT>` HTML Tags. Nicht jeder Browser muss das unterstützen.
- Script-Code in X3D/VRML eingebettet:
Standard-Feature schon in VRML 2.0;
benutzt einen so genannten “Script Node”.
- Java Applet kommuniziert mit Viewer:
“External Authoring Interface” (VRML) bzw.
„Scene Access Interface“ (X3D).



X3D/VRML + JAVA

- Java Klassen können Nodes des Scene Graph manipulieren, deren Identifier sie kennen.
- Proprietäre Viewer müssen entsprechende Klassenbibliotheken mitbringen.
- Man kann das Paket Xj3D benutzen, um X3D zu erzeugen bzw. per SAI zu manipulieren, siehe web3d.org



X3D Scene Graph

- Kernstück der X3D Applikation
- Besteht aus Knoten („Nodes“)
- Einführung neuer Knotenarten und Feld-Datentypen ggü. VRML
- Geringe Änderungen beim Lighting- und Event-Handling ggü. VRML

Beispiel X3D <-> VRML

```
<Transform translation = "3 0 0"> Transform {
    translation 3 0 0
    children [
        <Shape> Shape {
            <Sphere radius="2"/> geometry Sphere {
                radius 2
            }
            <Appearance> appearance Appearance {
                <Material diffuseColor = material Material {
                    "1 0 0"/> diffuseColor 1 0 0
                }
            }
        }
    ]
}
```



Weitere Features von X3D

- X3D unterstützt mehrere File-Encodings, z.B. VRML, XML und „compressed Binary“ (mit Verschlüsselungs-Option)
- CompressedBinary = erhöhter Datendurchsatz, Quellcode nicht lesbar
- Einfache Integration in Web-Anwendungen
- Einfacher plattform- und anwendungsübergreifender Datenaustausch denkbar (prüfen ob z.B. JSON aus JavaScript-Knoten möglich ist)



Modularität, Packages

- Der X3D Core stellt Grundfunktionen bereit, die durch Packages erweitert werden
- Beispiel: VRML Package
- Der Package-„Baukasten“ ist standardisiert
- Man kann eigene Packages und Profile erstellen und anbieten
- Dadurch können neue Anwendungsgebiete wie z.B. Medizin gut erschlossen werden



Erweiterungen (Teams)

- H-Anim: Humanoid Animation Modell für menschliche Avatare zwecks Realismus
- DIS-XML: Unterstützung für Distributed Interactive Simulation support für X3D, um komplexe Modellierungen zu ermöglichen
- GeoSpatial: Tools und ein ComponentModell für die Darstellung von geografischen Daten
- X3D-Shaders: Verbesserung der Darstellung in 3D-Welten, bietet ein programmierbares Shader Component Modell an



Mehr Erweiterungen...

- CAD: Component Modell, um den Austausch von 3D-Modellen aus CAD-Programmen zu ermöglichen
- Medical: Component Modell für die Darstellung von anatomischen Strukturen aus Datenbanken u.a. Quellen
- VizSim: Visual Simulation and Communication ermöglicht die Kommunikation zwischen X3D Anwendungen durch XML-WebServices



Tools für X3D (und VRML)

- Textbasiert: X3D Edit, PSPad, Vim
- Graphisch
 - ArtOfIllusion: Für einfache Strukturen
 - WhiteDune: Open Source, auch Linux
 - WireFusion: unterstützt Video/MP3 Integration
 - Vizx3D: Modeller für Avatare
 - SwirlX3D: Visual AuthoringTool
 - OctagaProducer: Tool für 3D-Welten
 - Vivaty Player (vormals Flux): Windows only



Plugins und Viewer (Auswahl)

- Octaga: Brauchbarer StandalonePlayer
- Blaxxun3D: Kleines Applet, kann durch JS konfiguriert werden, proprietär
- BS ContactVRML/3D: Encrypted Content, H-Anim Support (besonderes Profil)
- FluxPlayer: ActiveXControl, “high performance viewer”
- Xj3D: Java Package für Browser und Toolkit



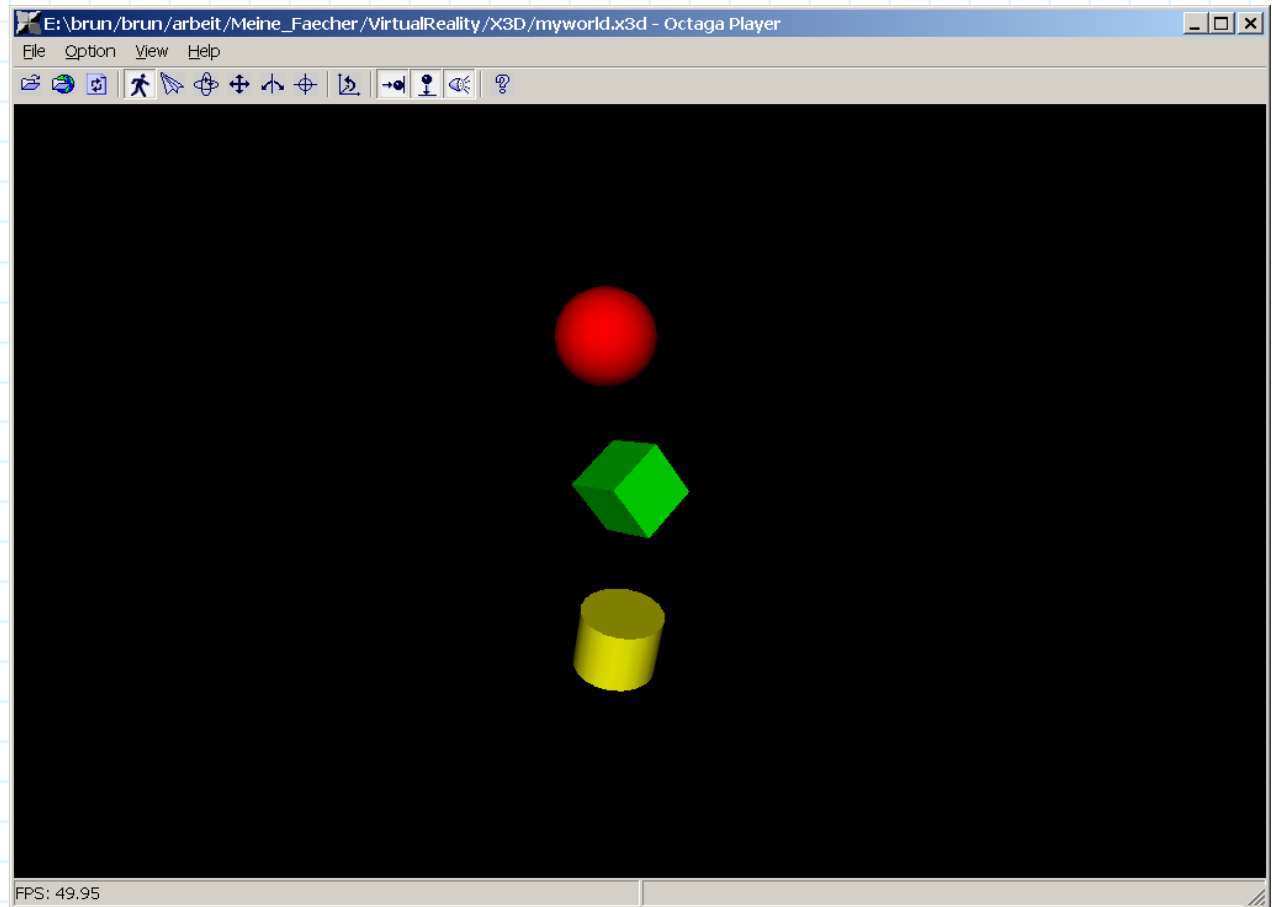
Developer Toolkits

- CyberX3D: Bibliothek für VRML/X3D für Java und C++, frei
- OctagonModellerSDK: sehr umfangreiche Bibliothek für dynamische Welten, ermöglicht Zugriff auf MPEG4 Videos
- BS SDK: Umfangreiche Bibliothek in C/C++, ermöglicht den Einsatz von Verschlüsselungen, Eventmanager
- Xj3D DeveloperToolkit: Java basiertes Toolkit, um X3D Content in Java Anwendungen zu importieren und zu manipulieren

Struktur einer X3D Datei

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN"
"http://www.web3d.org/specifications/x3d-3.0.dtd">
<X3D profile='Immersive'>
  <head>
    <componentname='Geospatial' />
    <componentname='NURBS' level='2' />
    <metaname='description' content='X3D example ' />
    <metaname='filename' content='X3dExample.x3d' />
  </head>
  <Scene>
    <!--Scenegraphnodesareaddedhere-->
  </Scene>
</X3D>
```

X3D Beispiel mit Bewegung



myworld.x3d