2 3D-Echtzeitgrafik im World Wide Web

Zur Heranführung an die Thematik dieser Arbeit ist dieses Kapitel der Internet-basierten 3D-Echtzeitgrafik gewidmet. Dazu erfolgt zunächst eine kurze Einführung in interaktive Echtzeitgrafik allgemein und damit verbundene Begriffe rund um das Gebiet Virtuelle Realität (VR), darunter schwerpunktmäßig *Desktop-VR*. Zu dieser Klasse von VR-Anwendungen unter Nutzung von Standardcomputern ohne spezielle Ein- und Ausgabegeräte gehört auch die im Fokus der Arbeit liegende 3D-Echtzeitgrafik im World Wide Web, nachfolgend kurz *Web3D-Grafik* genannt. Es erfolgt deshalb zunächst eine Charakterisierung und Darstellung von Besonderheiten und Problemen, die sich durch die Übertragung von komplexen 3D-Grafikdaten über Internetverbindungen und Anzeige auf heterogenen Client-Plattformen ergeben. Die sich anschließende Betrachtung von aktuellen Anwendungsdomänen für Web3D-Grafik zeigt die Zunahme an Einsatzmöglichkeiten für dreidimensionale, interaktive Grafik bei Web-Anwendungen und motiviert somit weitere Forschungsarbeiten und die Entwicklung von Standards in diesem Bereich.

Aktuelle Web3D-Formate werden in diesem Kapitel im Überblick vorgestellt, wobei sich eine grobe Einteilung vornehmen läßt in proprietäre Formate kommerzieller Anbieter und Standardformate, die durch institutionelle Gremien entwickelt werden. Einen Schwerpunkt bilden dabei die Formate rund um das künftige Web3D-Standardformat *Extensible 3D (X3D)*, auf das in folgenden Kapiteln Bezug genommen wird. Wenn sich die beiden Lager auch gegenwärtig noch gegenüberstehen, sind doch Synergieeffekte und Technologietransfers zwischen proprietären und standardisierten Formaten zu beobachten. So schließt dieses Kapitel mit einer Betrachtung zum Entwicklungsstand und aktuellen Trends bei Web3D-Grafik. Kommerzielle Aspekte werden ebenso erläutert wie Standardisierungsbemühungen, die gesteigerte visuelle Qualität von Web3D-Grafik, die Erweiterung der Zielplattformen auf mobile Endgeräte und schließlich der nach wie vor ungenügend unterstützte Autorenprozeß für 3D-Grafikanwendungen im World Wide Web.

2.1 Einführung und Abgrenzung

2.1.1 Grundlagen interaktiver 3D-Grafik

Im Unterschied zu Verfahren aus den Bereichen CAD, 3D-Modellierung und Animation wird bei *3D-Echtzeitgrafik* ein computerinternes 3D-Modell einer Szene abhängig von einer virtuellen Kamera in Echtzeit dargestellt. Der Nutzer kann in der Regel seinen Blickpunkt und damit die virtuelle Kamera verändern und auch auf andere Weise interaktiv die Darstellung der dreidimensionalen Szene beeinflussen. Es werden also nicht vorgefertigte Bilder von Objekten dargestellt, sondern transformierte, beleuchtete und triangularisierte 3D-Objekte in die zweidimensionale Ebene eines Ausgabegerätes projiziert. Dabei sorgt moderne Grafikhardware auch bei hohen Polygonzahlen der Modelle für eine Darstellung realistisch anmutender Szenen mit für interaktive 3D-Anwendungen akzeptablen Bildwiederholraten.

> *Eine 3D-Applikation läßt sich allgemein als ein Programm definieren, das dem Benutzer eine dreidimensionale Szene präsentiert, mit der er in Echtzeit interagieren kann [Pierc01].*

Dazu kann ein breites Spektrum von Ein- und Ausgabegeräten verwendet werden. Der interaktive Teil einer 3D-Anwendung, also die dreidimensionale Benutzungsoberfläche, wird ausführlich im Kapitel 3 dieser Arbeit behandelt. Dabei spielt der Begriff *Freiheitsgrad,* auch als *Degrees Of Freedom (DOF)* bezeichnet, eine Rolle. So ergeben sich beispielsweise aus den Aufgaben Positionierung (Translation) und Orientierung (Rotation) in der dritten Dimension 6 Freiheitsgrade. Diese müssen entweder über geeignete Eingabegeräte (z.B. *Spacemouse*) kontrolliert oder aber durch die Benutzungsschnittstelle unterstützt werden.

Während sich bei der 3D-Modellierung primär Austauschformate für Szenenbeschreibungen finden lassen, die mit komplexen Autorenwerkzeugen generiert werden, dominieren im Echtzeitgrafikbereich neben einigen 3D-Austauschformaten vor allem APIs und 3D-Toolkits zur Programmierung von 3D-Anwendungen. Neben einer Szenenmodellierung bzw. Animation von Objekten sind auch häufig komplexe Simulationen, Interaktionen und Datenreduzierungstechniken nötig. Außerdem müssen verschiedenste spezialisierte Ein- und Ausgaberäte unterstützt werden, wofür imperative Programmierung das erste Mittel der Wahl ist. Dabei lassen sich Low-Level- und High-Level-APIs unterscheiden. *Low-Level APIs* bilden standardisierte Software-Schnittstellen zur Grafikhardware und umfassen Bibliotheken von Funktionen, Prozeduren oder auch Klassen zur Erzeugung geometrischer Primitive, für Blickpunkttransformationen, Farbzuweisungen, Szenenbeleuchtung, Anti-Aliasing, Alpha-Blending etc. Die wichtigsten Vertreter sind *OpenGL* als hardwarenahe Schnittstelle und plattformübergreifende Abstraktionsschicht für Anwendungsprogrammierung sowie *DirectX* bzw. *Direct3D* im *Immediate Mode* als Realisierungsbasis für die Windows-Plattform.

Zur Gruppe der *High-Level APIs* gehören zumeist objektorientierte Toolkits, die mit der Einführung einer Szenengraphebene eine Abstraktion zu Low-Level-APIs darstellen und zudem teilweise Interaktionsfunktionalität anbieten.

> *Ein Szenengraph ist ein gerichteter azyklischer Graph (Directed Acyclic Graph) von Knoten zur effizienten hierarchischen Beschreibung einer dreidimensionalen Szene. Knoten (Nodes) können u.a. Gruppierungskonstrukte, geometrische Objekte, Transformationen, Materialien und Texturen sein, deren Zustand in Form von Feldern (Fields) beschrieben wird.*

Ein Szenengraph wird zur Laufzeit traversiert, wobei Informationen (z.B. Transformationen und Farben) für das Rendering akquiriert werden und Funktionsaufrufe der Low-Level-API erfolgen. Da eine Szene üblicherweise als Transformationshierarchie von 3D-Objekten mo-

delliert werden kann und im Szenengraph zudem Eigenschaften von Knoten (z.B. Materialdefinitionen) an nachfolgende Subhierarchien delegiert werden, lassen sich 3D-Anwendungen auf Szenengraphbasis gut strukturiert erstellen. Toolkits bieten typischerweise Klassen für jeden Knoten an, in denen Methoden zum Ändern der Felder und des Objektzustandes definiert sind. Verschiedene APIs wurden in der letzten Dekade entwickelt, darunter so erfolgreiche Toolkits wie *Iris Performer* für performante Multiprozessor VR-Simulationen und *Open Inventor* [Strau92] als anwendungsfreundliches, erweiterbares und objektorientiertes Toolkit. Andere 3D-APIs, wie *Cosmo3D, OpenGL Performer, OpenGL Optimizer, OpenGL++* oder *Fahrenheit* waren nur in bestimmten Anwendungsdomänen erfolgreich oder existierten nur für kurze Zeit. Mit *OpenSG Plus* [OpenSG@] ist ein vielversprechendes OpenSource-Projekt begonnen worden, das mehrere Aktivitäten und Leistungsmerkmale zurückliegender API-Entwicklungen bündelt. Natürlich existieren auch im akademischen Bereich zahlreiche Toolkits zur Programmierung von Anwendungen der Virtuellen Realität.

2.1.2 VR, VE, DVE und weitere Begriffe

Für die Arbeit wichtige Begriffe, Abkürzungen und Synonyme sollen im folgenden kurz erläutert werden. Die klassische Domäne interaktiver 3D-Echtzeitgrafik ist die *Virtuelle Realität (Virtual Reality – VR)*.

> *Virtuelle Realität ist blickpunktabhängige Echtzeitberechnung von 3D-Grafik, die es Nutzern ermöglicht, mit Hilfe einer immersiven Mensch-Computer-Schnittstelle durch eine computergenerierte, künstliche räumliche Umgebung realistischer oder abstrakter Natur zu navigieren und mit dieser zu interagieren.*

Dabei sollen die verschiedenen Sinnesreize mit der VR-Schnittstelle so stimuliert werden, daß der Benutzer das Gefühl hat, in die Umgebung einzutauchen *(immerse)*. VR-Anwendungen werden auch synonym als VR-Umgebungen, immersive virtuelle Umgebungen bzw. Welten *(virtual worlds)* oder auch *immersive virtual environments* bezeichnet. Dabei sind immersive Schnittstellen und entsprechende VR-Ein-/Ausgabegeräte jedoch keine Notwendigkeit für virtuelle Umgebungen (auch englisch VE abgekürzt).

> *Virtuelle Umgebungen sind ein Überbegriff für virtuelle Welten oder interaktiv manipulierbare, navigierbare 3D-Szenen mit verschiedenen Hardware-Konfigurationen.*

Charakteristisch für echte, also immersive, VR-Umgebungen sind spezialisierte höherdimensionale Eingabegeräte (6 DOF-Tracker, Datenhandschuhe etc.) und häufig stereoskopische Ausgabegeräte, die ein Immersionsgefühl ermöglichen (*HMD - Head Mounted Display*[1], *CAVE*[2] etc.). Auch wenn in den letzten Jahren ein Transfer vom High-End- zum Low-End-Sektor für einige VR-Geräte stattfand (Bsp. Stereobrillen *Chrystal Eyes* und *Elsa Relevator*), sind diese nach wie vor unkomfortabel und finden nur geringe Akzeptanz außerhalb der häufig Experten vorbehaltenen VR-Anwendungsdomänen Medizin, Militär, Fahrzeug- und Maschinenbau, Naturwissenschaften, Erdölindustrie etc. Aus der Historie militärischer Einsatzsimulationen entwickelten sich auch die verteilten virtuellen Umgebungen *(Distributed Virtual Environments – DVE)*.

[1] Datenhelm mit CRT/LCD-Schirmen vor jedem Auge, erlaubt die Stereodarstellung für einen Nutzer.

[2] Würfelförmiger Raum mit bis zu 6 Rückprojektionswänden, in dessen Mitte sich ein oder mehrere Nutzer befinden.

> *Distributed Virtual Environments, auch Networked Virtual Environments (Net-VE) genannt, sind Internet-basierte dynamische Mehrbenutzer-Umgebungen, bei denen die prinzipiell gleiche virtuelle Welt auf verschiedenen, miteinander vernetzten Rechnern betrachtet werden kann und gleichzeitige Navigation und Interaktionen mehrerer Nutzer möglich sind.*

Dabei kommen bei Net-VEs zu den bestehenden VR-Problemen (Immersionsgefühl, Performance etc.) noch Herausforderungen, wie z.B. gemeinsames Raumempfinden, Präsenzgefühl, Zeitgefühl sowie gemeinsame Kommunikations- und Interaktionsmöglichkeiten hinzu [Singh99]. Verteilte virtuelle Anwendungen, die eine Mischung von verteilten Systemen und grafischen Anwendungen darstellen, stehen aber wie VR-Applikationen nicht im Mittelpunkt dieser Arbeit. Die im Kapitel 3 vorgestellten Metaphern und Widgets interaktiver 3D-Anwendungen haben jedoch häufig ihren Ursprung in diesen Bereichen, zu denen es bereits zahlreiche Forschungsarbeiten gibt.

Ein jüngeres Gebiet ist die *Erweiterte Realität (Augmented Reality – AR),* die beispielsweise im Servicebereich der Autoindustrie Einsatz findet, wo Handlungsschritte zur Wartung von Teilen eingeblendet werden können.

> *Augmented Reality ist die Überlagerung realer Szenerien durch virtuelle, meist dreidimensionale Darstellungen. Möglich wird dies durch Mensch-Computer-Schnittstellen, die eine gleichzeitige Wahrnehmung der Realität und von computergenerierten Bildern gestatten, z.B. durch semi-transparente HMD.*

Für diesen Bereich ist neben der Lösung von Problemen zur Markierung der realen Szene und Überblendung virtueller Objekte auch die Entwicklung spezieller Interaktionstechniken und Benutzungsschnittstellen nötig. Eher im Fokus dieser Arbeit stehen jedoch Anwendungen der sogenannten *Fishtank-VR* [Ware93], wo Nutzer an normalen Arbeitsplätzen sitzen, die Ausgabe auf dem Monitor stereoskopisch erfolgt und über Head-Tracking die Kopfposition registriert wird. Auch wenn autostereoskopische Monitore inzwischen bereits weiter entwickelt und preiswerter sind, konnten sich Anwendungen mit Nutzer-Tracking bisher auch nicht etablieren. Anders ist dies mit dem Sektor *Desktop-VR,* der virtuelle Umgebungen für einen vergleichsweise großen Personenkreis zugänglich macht.

2.1.3 Schwerpunkt Desktop-VR

PCs werden zunehmend leistungsfähiger, sie dringen mit ihrer Grafik-Performance in Bereiche vor, die bisher ausschließlich High-End Workstations vorbehalten waren. Während immersive VR-Applikationen durch spezialisierte, teure Hardware ein realistisches Eintauchen in virtuelle Welten erlauben, gestatten Desktop-VR Lösungen die nicht-immersive Betrachtung von 3D-Grafik.

> *Desktop-VR-Anwendungen lassen sich als nicht-immersive Virtuelle Realität auf Standardcomputern und ohne Nutzung spezieller Ein-/Ausgabegeräte definieren, wodurch sie für einen breiten Einsatz prädestiniert sind.*

Dabei ist die Nutzung dreidimensionaler Eingabegeräte oder stereoskopischer Ausgabegeräte nicht ausgeschlossen. Wenn interaktive Echtzeitgrafik eine echte Erfolgsgeschichte schreiben sollte, dann wahrscheinlich über den Bereich Desktop-VR, der im Zentrum dieser Arbeit steht. Aus Gründen der Bedienbarkeit, Übersicht und Navigationskontrolle favorisieren auch andere Wissenschaftler, z.B. Ben Shneiderman, Desktop-VR-Anwendungen, also das *„looking at rather than being in“* [Shnei02]. Der PC wird zunehmend als VR-Plattform der Zukunft betrachtet, wobei das Internet hierbei eine entscheidende Rolle spielen wird [Bulli98]. Vorteile von Desktop-VR gegenüber VR-Anwendungen sind (teilweise aus [Zelez97]):

- Sehr gute Anzeigequalität durch CRT-Displays im Gegensatz zu HMD-Auflösungen;
- Nutzer muß keine unbequemen Display-Geräte tragen, ist nicht isoliert;
- Kombination verschiedener Anzeigegeräte möglich;
- Höhere Präzision und Kontrolle als mit VR-Trackern;
- Ermüdungserscheinungen werden durch Hand- und Armauflage vermieden;
- Kostenfaktor günstiger, Hardware überall verfügbar.

Als Nachteile sind das fehlende Immersionsgefühl und die Bewegungseinschränkung im Vergleich zu CAVEs oder HMD-Lösungen zu nennen. So lassen sich auf aktuellen Standard-PCs Anwendungen mit interaktiver 3D-Echtzeitgrafik rein technisch gut benutzen. Während jedoch seit ca. 15 Jahren im Forschungsbereich VR eine Vielzahl von Präsentations- und Interaktionstechnologien entwickelt wurden, sind für einen größeren Nutzerkreis Mehrwert versprechende 3D-Anwendungsklassen noch nicht richtig etabliert. Erst mit der Verbreitung von 3D-Grafik über das Internet in Kombination mit Desktop-VR auf dem Client-Rechner besteht eine Chance auf eine breitere Anwendung von interaktiver 3D-Grafik. Im folgenden wird Web3D-Grafik darum näher charakterisiert und anschließend anhand von typischen Anwendungsklassen und aktuellen Formaten der gegenwärtige Stand dargestellt.

2.1.4 Charakterisierung von Web3D-Grafik

> *Der Begriff Web3D-Grafik umfaßt sowohl proprietäre als auch offene Technologien und Formate für 3D-Echtzeitgrafik im World Wide Web, wobei typischerweise von einem Server Grafikdaten ohne spezialisierte Netzwerkprotokolle zum Client übertragen werden, um dort in einem Webbrowser betrachtet und manipuliert werden zu können.*

In Abgrenzung zu DVEs bzw. Net-VEs handelt es sich bei Web3D-Grafik in der Regel um Anwendungen, die nur von einem Nutzer verwendet werden. Zudem stehen meistens keine besonderen Ein- oder Ausgabegeräte zur Verfügung, womit Web3D-Grafik als verbreitete Form von Desktop-VR aufgefaßt werden kann.

Mit der Einführung der Plug-In-Technologie im Jahre 1994 war es möglich geworden, Webbrowser um Anzeigemöglichkeiten auch für 3D-Grafik zu erweitern. Im gleichen Jahr wurde mit der VRML-Spezifikation begonnen. Das bald danach gegründete *VRML Consortium* reichte 1997 die überarbeitete Spezifikation 2.0 zur Standardisierung ein, woraus der VRML97 genannte ISO/IEC-Standard wurde. 1998 folgte mit Java3D eine 3D-Grafik-API auf Java-Basis, ab 1999 die Entwicklung von MPEG-4/BIFS auf VRML-Basis als 3D-Teil des Standard-Frameworks MPEG-4, ab 1999 dann die Weiterentwicklung von VRML97 in Form von X3D. Aus dem VRML-Consortium wurde das *Web3D-Consortium* [Web3D@], um der nunmehr breiteren Palette an 3D-Standards bzw. Quasi-Standards gerecht zu werden. Parallel dazu entstanden zahlreiche konkurrierende, proprietäre Web3D-Technologien mit sehr unterschiedlichem Erfolg. Für ausführlichere Darstellungen der interessanten und unmittelbar mit dem Internet-Boom verknüpften Entwicklungsgeschichte von Web3D-Grafik wird z.B. auf [Götz02] und [Walsh01] verwiesen.

2.1.4.1 Technologie

Prinzipiell wird interaktive Web3D-Grafik in Form plattformunabhängiger, häufig textbasierter Szenenbeschreibungsdateien abgelegt, die diverse, teilweise verteilte Dateien für Medien, Skripte und Programme referenzieren. Auf einem Server können diese Daten je nach Technologie als Datei oder bei spezialisierten Serverarchitekturen in Datenbanken für 3D-Objekte, Medienassets und Programmodule gespeichert werden.

Nach der Übertragung von einem Web-Server über Standardprotokolle (z.B. HTTP) zum Client werden die Dateien im Browser mit Hilfe eines Web3D-Players für das jeweilige Format dargestellt. Dieser kann als Plug-In oder Applet realisiert sein.

Plug-Ins für Web-3D-Grafik sind problemlos zu integrieren und erweitern die Funktionalität eines Webbrowsers. Sie erlauben kleinere Dateigrößen für zu ladende 3D-Szenen als Applets, da Grundfunktionalität bereits im Player integriert ist. Die Grafik-Performance ist aufgrund der direkten internen Nutzung von OpenGL oder DirectX sehr gut. Ernst zu nehmende Probleme waren jedoch die Größe von Plug-Ins und die oft hinderliche Installation. Beide konnten aber durch modulare Plug-In-Technologien mit sukzessivem Download und Automatikinstallation von Zusatzfunktionalität nahezu beseitigt werden. Noch immer besteht jedoch die Schwierigkeit, daß beim Betrachten von 3D-Inhalten auf Webseiten Verzögerungen für Download und (automatisierte) Installation von Plug-Ins nur ungern in Kauf genommen werden. Von einer weiten Verbreitung von 3D-Plug-Ins kann insbesondere bei kleineren proprietären Technologien nicht ausgegangen werden. Weitere wirtschaftliche Aspekte dieser Realisierungsform werden in einer Web3D-Studie unter [Salient@] beschrieben. Für die führenden proprietären Formate – darunter Viewpoint, Shockwave 3D und Cult3D – sowie für die Standards VRML97 und MPEG-4 existieren Plug-In-Lösungen.

Java-Applets stellen eine elegante Realisierung zur Anzeige von 3D-Grafik ohne zusätzliche Plug-In-Installationen dar. Durch schnellere PCs und leistungsfähigere Grafikkarten sind inzwischen auch schnelle und qualitativ hochwertige Darstellungen möglich. Dennoch sind Java-Applets in der Regel ladezeitenunfreundlich, da die wenigsten tatsächlich dynamisch in den Browser geladen werden, und somit stete Client-Server Abfragen auslösen [Salient@]. Auch stellt sich die Entwicklung von Applets aufwendiger dar als die Erstellung von Austauschformat-Dateien, die einfach durch ein Plug-In zur Anzeige gebracht werden. Applets sind bei kommerziellen Formaten eher selten anzutreffen, dafür existieren mit Shout3D, Blaxxun3D und Cortona Jet gleich mehrere Lösungen auf VRML97-Basis.

2.1.4.2 Probleme und Lösungen

Mitunter stehen nur geringe Bandbreiten zwischen Server und Client zur Verfügung, wobei Web3D-Client-Plattformen sehr leistungsschwach und heterogen sein können. Daraus ergibt sich eine hohe Startup-Latenz, also Zeit für Abruf, Transport, Dekodieren und Rendern, bevor ein Nutzer eine Szene betrachten und mit ihr interagieren kann. Reduzierte Interaktionsmöglichkeiten und geringere Dateigrößen durch detailärmere, unrealistischere 3D-Modelle als z.B. bei VR stellen eine simple Lösung dieser Probleme dar, was jedoch insbesondere kommerziellen Ansprüchen häufig nicht genügt. Somit liegt die Entwicklung von skalierbaren Lösungsansätzen zur Modularisierung, Kompression und zum Streaming von 3D-Daten nahe (s.a. [Rukzi01]).

Modularisierung kann auf vielfältige Weise und meistens bereits während des Autorenprozesses erfolgen. Typischerweise werden Szenen in kleinere Einzelwelten zerlegt und Hyperlinks, Inline-Knoten oder ähnliche Konzepte zum Nachladen von Geometrien und Medien zur Verringerung der Startup-Latenz eingesetzt. Bestimmte Medien können alternativ eingesetzt werden, z.B. ein Bild statt einem Video oder eine Textur statt einer Detailgeometrie. Beim *Universal Media* – Ansatz [UniversalMedia@] wird ein Standardsatz verschiedenartiger Texturen, Modelle und Sounds auf dem Client lokal als Bibliothek gespeichert. Auf diesen wird dann, ähnlich wie bei Fonts, beim Rendern zurückgegriffen. Eine klassische Modularisierungsform stellen unterschiedliche Detaillierungsstufen *(Level of Detail - LOD)* dar. Auch für die Animation von Charakteren kann Übertragungskapazität eingespart werden, indem nur für bestimmte Gelenkpunkte Differenzpositionen übertragen werden und die eigentliche Berechnung erst auf dem Client erfolgt.

Kompressionsverfahren spielen bei Web3D-Grafik sowohl für Geometrien als auch Einzelmedien eine Rolle. Neben bekannten Bild- und Videokompressionsverfahren sind vor allem leistungsfähige Geometriekompressionsverfahren von Bedeutung. Eine verlustfreie Entropiekodierung für Austauschformate kann als einfache Form verwendet werden. Der Aufteilung von Oberflächen in Dreiecksnetze werden die verlustfreien Kodierungsverfahren *Generalized Triangle Strips* und *Generalized Triangle Mesh* gerecht, bei denen nicht für jedes Dreieck alle Eckpunkte in Form von X,Y,Z-Koordinaten abgespeichert werden müssen. Auch mit verlustbehafteter Kompression lassen sich Polygonnetze vereinfachen *(Mesh Simplification Technologies)*, um z.B. LOD-Stufen automatisch generieren zu können. Solche *Multiresolution Meshes* sind in der Regel Dreiecksnetze, innerhalb derer Dreiecke durch *Split*-Operationen erneut unterteilt werden können oder umgekehrt mit *Collapse*-Operationen vereinfacht werden (s. z.B. [ModNav3D@]). Dazu gibt es mehrere Verfahren der „Topologischen Chirurgie". Andere aus der Computergrafik bekannte Techniken, wie *Subdivision Surfaces* oder *Boundary Representations,* lassen sich nur bedingt für Web3D-Grafik einsetzen, weil Dreiecksnetze und bei einigen 3D-Formaten auch NURBS-Oberflächen die primäre Repräsentationsform darstellen. Interessante Kompressionsformen auf Basis von ASCII-Dateiformaten wurden z.B. von Isenburg entwickelt [Isenb02]. Dabei werden für Listen von Vertexpunkten und Indexlisten, die typischerweise den Hauptteil eines Szenengraphen ausmachen, binäre Kodierungen innerhalb der Textdateien erstellt, was Dateigrößen drastisch schrumpfen läßt.

Streaming von Objektgeometrien wird unter anderem durch *Progressive* oder auch *Multiresolution Meshes* möglich. Dabei erfolgt für ein Modell üblicherweise die Übertragung eines detailarmen LOD und eine spätere sukzessive Verfeinerung auf dem Clientrechner. Das Originalmodell wird durch Eckpunktvorhersage bei den *Split*-Operationen rekonstruiert, wobei nachträglich übertragene Verschiebungsvektoren eine immer genauere Darstellung ermöglichen. Neben der verbesserten Bandbreitenausnutzung und Verringerung der Startup-Latenz läßt sich mit solchen Verfahren auch eine Skalierung auf leistungsschwächere Endgeräte erzielen und die Framerate bei gleichzeitigem Verlust an Details erhöhen. Auch zeitabhängige, durch den Server gesteuerte Updates einer 3D-Szene sind möglich, bei denen jeweils Teile der Szene durch z.B. detailliertere oder alternative Szenenbeschreibungen ersetzt werden. Auch für verknüpfte Medien, z.B. Video- und Audiodateien läßt sich über Streamingserver klassisches Video- oder Audio-Streaming durchführen. Bisher wurden solche Verfahren jedoch selten mit 3D-Geometriestreaming kombiniert. Das liegt unter anderem daran, daß Audio und Video isochrone Übertragung verlangen, 3D-Grafik jedoch nicht oder selten.

Neben den aufgeführten technischen Problemen ist der Sektor Web3D-Grafik jedoch auch durch einen Mangel an Standards und Richtlinien für die Entwicklung von 3D-Anwendungen und die Interaktion mit ihnen auf Nutzerseite geprägt. Besonders die Navigation und Interaktion in 3D-Playern gestalten sich von Format zu Format sehr unterschiedlich, was die ohnehin schwierigere Navigation in der dritten Dimension zusätzlich erschwert. Neben teilweise unzureichenden Autorenwerkzeugen werden nur selten Konzepte von Wiederverwendbarkeit unterstützt. Auch zur Realisierung komplexeren Verhaltens jenseits von einfachem Anklicken und Betrachten von 3D-Objekten bieten nur wenige Formate geeignete Konzepte.

2.2 Anwendungsdomänen

Seit Mitte der 90er Jahre bildeten sich mit der Entwicklung verschiedener Web3D-Grafikformate auch zahlreiche Anwendungsdomänen interaktiver 3D-Grafik im Internet heraus. Naheliegend ist einerseits die Darstellung von Objekten oder Szenen, die a priori räumlich sind und deren Räumlichkeit erfahren bzw. betrachtet werden soll. Daraus ergeben sich Anwendungen in den Bereichen CAD, Modellierung und Animation, Produktpräsentation / E-Commerce, Lernumgebungen, Bedienungsanleitungen, Architekturvisualisierungen etc. Andererseits werden auch abstrakte Daten im World Wide Web visualisiert, bei denen 3D-Grafik Vorteile gegenüber zweidimensionalen Präsentationen hat, z.B. bei Informations- und Hierarchievisualisierungen, Navigationsunterstützung, Diagrammen etc. Die folgende Zusammenstellung gibt einen Überblick zu den vielfältigen Web3D-Anwendungsfeldern, wobei die Bereiche sich teilweise überlappen. Betrachten lassen sich Anwendungen unter verschiedenen Aspekten, wie Darstellungsqualität, Geschwindigkeit, Nutzerfreundlichkeit, Medienintegration bzw. Integration in Webseiten oder den Interaktionsgrad für Nutzer. Mit dieser Darstellung soll das Potential von Web3D-Grafik deutlich gemacht und die Entwicklung von Standards in diesem Bereich motiviert werden. Ausgangspunkt der folgenden Abschnitte ist die Arbeit von Rukzio [Rukzi01], in der Formate, Technologien und Architekturkonzepte für 3D-Web-Applikationen im Detail untersucht wurden. Für den Bereich Produktpräsentationen / E-Commerce wurde der Einsatz von Desktop-VR in [Dachs99b] dargestellt.

2.2.1 Produktpräsentationen / E-Commerce

Diese Gruppe ist das primäre Anwendungsfeld für 3D-Grafik im Netz [Leavi99] und verspricht kommerziellen Nutzen. Es ist sinnvoll, 3D-Produktdarstellungen einzusetzen, wenn Produkte komplexe Funktionen haben, erklärungsbedürftig sind sowie viele Varianten und Konfigurationsmöglichkeiten aufweisen. Zu den typischen Branchen für Produktvisualisierungen gehören Medizintechnik, Bau-, Möbel-, Automobil- und Maschinenindustrie. Zunehmend werden auch Kosmetika, Raumdekor, Einrichtungsgegenstände und Kleidung dreidimensional vermarktet. Ende der 90er Jahre wurden virtuelle Shopping Malls (meist mit VRML) eingesetzt, bei denen die Navigation aufwendig und zeitraubend war (Beispiel: *Active Worlds Shopping Mall @mart* [@mart@] s. Abbildung 1, 1). Inzwischen dominieren Produktpräsentationen als 3D-Fenster innerhalb multimedial angereicherter HTML-Seiten bekannter Shop-Lösungen. Zwingend notwendig ist die Anbindung solcher 3D-Lösungen an kommerzielle Shopsysteme. Zu den Anwendungen in diesem Bereich zählen:

- Interaktive Produktkonfiguratoren: z.B. für Autos [Cybelius@] (s. Abbildung 1, 2), Kinderwagen [INBY@] (3) oder PDAs [RealityBuy@] (4)
- Virtuelle Einrichtungsplaner im Wohn-, Küchen- und Badbereich: z.B. *Cera Design Fliesen- und Badplaner* [Cera@] oder Küchenkonfigurator [RealityBuy@] (s. Abbildung 1, 5)
- Bereich *Customer Relation Management (CRM)* und Service: z.B. *ServiceDirect* [Lunatic@] oder Online-Service für Maschinen [CriticalReach@] (s. Abbildung 1, 7)
- Bereich Kleidung und Mode: z.B. virtuelle Mannequins bei [Macys@] (s. Abbildung 1, 6)

Der Interaktionsgrad ist bei dieser Anwendungsklasse relativ gering und erstaunlich einheitlich. Üblicherweise können Produkte von allen Seiten betrachtet und vergrößert bzw. verkleinert werden; Tooltips erläutern einzelne Teile; Animationen von Teilobjekten lassen sich abspielen oder aktivieren; visuelle Eigenschaften – wie z.B. Farbe oder Texturen – sind zumeist über Steuerelemente außerhalb des 3D-Fenster einstellbar. Bei Einrichtungsplanern lassen sich zudem geometrische Transformationen durchführen, teilweise sogar im 3D-Fenster. Aufgrund der von Herstellern geforderten hohen visuellen Qualität von Produkten kommen hier statt VRML97 primär proprietäre Formate (z.B. Viewpoint VET oder Cult3D) zum Einsatz.

Abbildung 1: Web3D-Anwendungsklassen *Produktpräsentationen / E-Commerce* und *Unterhaltung und Freizeit*

2.2.2 Unterhaltung und Freizeit

Neben Produktpräsentationen / E-Commerce ist dies der wichtigste Anwendungsbereich, der von 3D-Online-Spielen dominiert wird. Dazu zählt das wohl bekannteste Online-Multiplayer-Spiel, „The Sims Online“ [SimsOnline@] (s. Abbildung 1, 8). Auch andere *Massively Multi-user Persistant Worlds* in grafisch aufwendigen Spielen wie „Everquest“ oder „Ultima

Online“ stellen mit über 100.000 simultanen Teilnehmern extreme Anforderungen an Multi-Server-Architekturen [Gehor03]. Eine Ausweitung auf diverse Endgeräte und Home-Cinema-Qualität mit lebensgroßen Avataren sind Zukunftsvisionen, an denen intensiv gearbeitet wird. Fast immer besitzen diese Spiele proprietäre Formate oder spezielle 3D-Engines, die oft in Stand-alone-Browsern laufen. Auch die weiter unten vorgestellten 3D-Chat-DVEs lassen sich zu dieser Kategorie zählen. Zu weiteren Anwendungen in diesem Bereich gehören:

- *Advergames* zur Verkaufsförderung, zumeist ohne Plug-Ins [3DGroove@] (s. Abbildung 1, 9)
- 3D-Cliparts für Einladungen, Glückwünsche, virtuelle Geschenke (z.B. Blumen) etc.
- Virtuelle Charaktere: synthetische Nachrichtensprecher, Trainer, anthropomorphe Interaktionsagenten etc. (z.B. Tutor und „Wahrsagerin“ [Eyematic@] in Abbildung 1, 10). Verfahren und Formate für synthetisches Video bzw. 3D Gesichtsanimationen existieren von Charamel [Charamel@], Eyematic [Eyematic@] und Pulse [Pulse@].
- Virtuelle Museen: z.B. Science Museum London [ScienceMuseum@], Helsinki-Museum [VirtualHelsinki@] (typisch dabei die Medienkombination, s. Abbildung 1, 11)
- Online-Galerien und Web3D-Kunst: z.B. unter [3D-Art@]
- Virtuelle Fitneßtrainer, Reisebüros oder Kartenreservierungssysteme mit 3D-Ansichten des Zuschauerraumes (Bsp. 3D-Ticket-Buchungssystem [Stella@] s. Abbildung 1, 12)

Für diese Web3D-Anwendungen kommen häufig Java-Applets, aber auch Plug-Ins für Standard- und proprietäre Formate zum Einsatz. Aufgrund des großen Nutzerkreises sind meistens nur wenige Interaktionen möglich, um eine einfache Bedienung zu erleichtern. Ausnahmen bilden natürlich Spiele. Gemeinsam ist allen Anwendungen eine hohe visuelle Qualität und meistens eine Integration in Webseiten mit zusätzlichen 2D-Inhalten.

2.2.3 Lehr- / Lernanwendungen

Diese Anwendungsklasse gewinnt zunehmend an Bedeutung, da sich komplexe Sachverhalte mit Hilfe von interaktiver 3D-Grafik leichter erklären und nachempfinden lassen. Im Web existieren zahlreiche Bau- oder Reparaturanleitungen bzw. virtuelle Bedienungshandbücher, womit sich Kunden, Händler und Servicetechniker schulen lassen können. Einzelne Schritte können animiert abgespielt, jedoch auch interaktiv selbst nachvollzogen werden. Daraus folgt, daß für diese Anwendungsklasse 3D-Navigation allein nicht ausreicht, sondern Web3D-Formtate geeignete Möglichkeiten zur Verhaltensmodellierung und Realisierung komplexer Animationen bereitstellen müssen. VRML97 kommt deshalb – auch aufgrund der weiten Verbreitung – häufig zum Einsatz, im Geschäftsbereich jedoch auch proprietäre Lösungen. Beispiele sind:

- Virtuelle Handbücher: z.B. für Flugzeug-Turbine [VirtualManuals@] (s. Abbildung 2, 1) oder Schrankmontage [INBY@] (s. Abbildung 2, 2)
- Virtuelle Charaktere: Beispieltutorial (s. Abbildung 1, 10) oder zur Vermittlung von Gebärdensprache [Sims02]
- 3D-Lernsoftware: z.B. für Naturwissenschaften, virtuelle Chemie etc.
- Geschichtsvisualisierungen (auch Edutainment): interaktives Nachempfinden historischer Ereignisse (z.B. detailtreue 3D-Raumfahrt-Geschichte [Cosmos@])

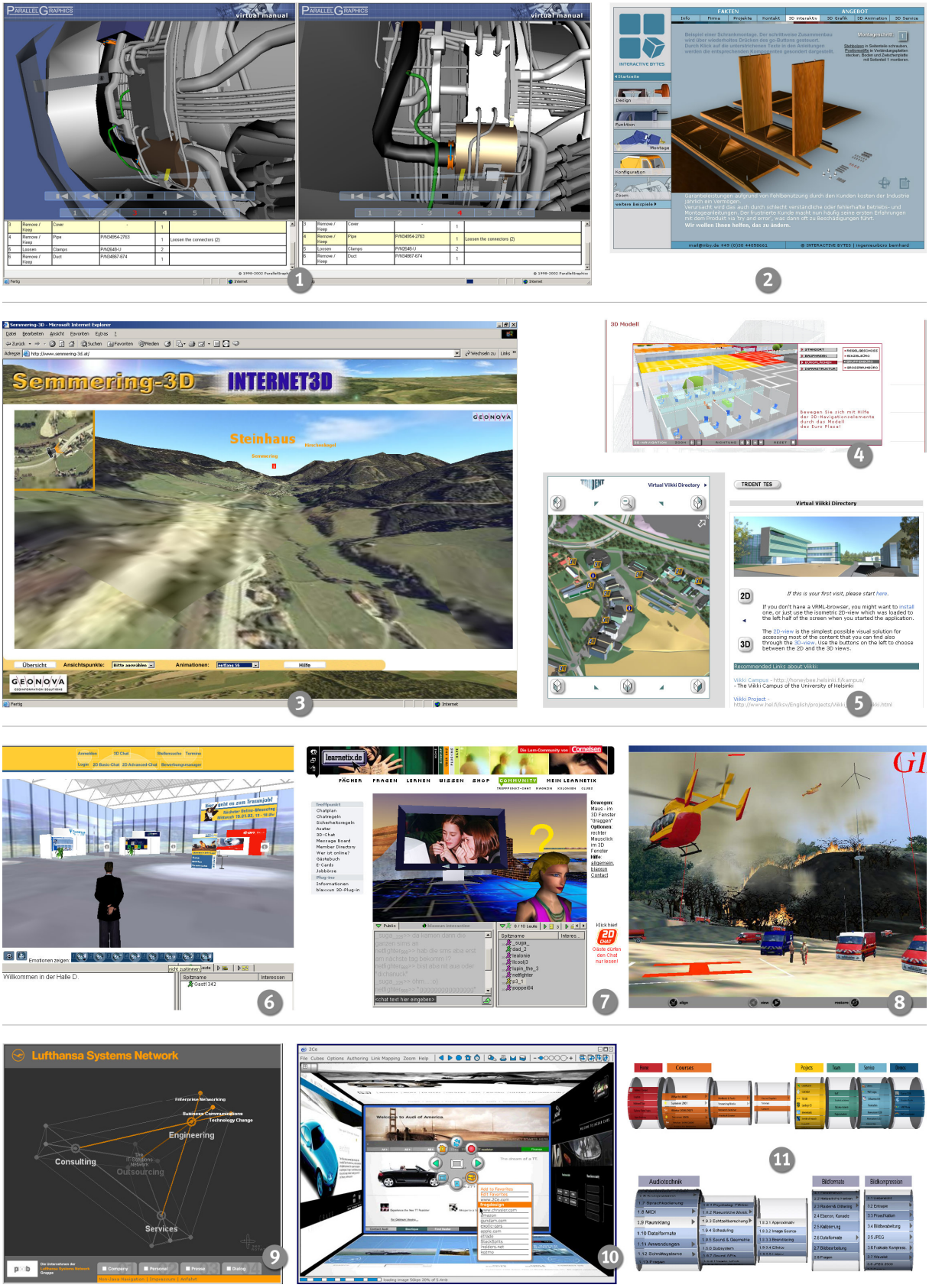

Abbildung 2: Web3D-Anwendungsklassen *Lehr-/Lernanwendungen, Architektur-, Städte und Landschafts-visualisierung, DVEs, Informationsvisualisierung und Navigation*

2.2.4 Architektur-, Städte- und Landschaftsvisualisierung

Hierbei handelt es sich um eine Anwendungsdomäne mit Zukunft, da die realistische Visualisierung von digitalen Geländemodellen bzw. 3D-Daten zur Städte- und Bebauungsplanung, Gebäudevermarktung, Bürgerinformation und nicht zuletzt für den Tourismus ein wichtiges Feld darstellt, das über andere Medien (z.B. 2D-Karten) nicht so gut erschlossen werden kann. Besonders wichtig sind realistische Visualisierungen der komplexen Modelle. Häufig sind die 3D-Ansichten innerhalb der Webseiten noch zu klein (s. Abbildung 2, 4), so daß kein echter Mehrwert gegenüber Fotos oder Bildpanoramen zu erkennen ist. Bei den Geoinformationssystemen der Firma Geonova [Geonova@] werden als Texturen Satellitenfotos eingesetzt, wobei das Nachladen Probleme bereiten kann (s. Abbildung 2, 3). Mehrere Firmen sind auf diesem Gebiet aktiv, darunter UMA Real Estate [UMA@] oder Mischek Bau AG [Mischek@] für Gebäudevisualisierung und Bebauungsplanung, andere im Tourismusbereich mit virtuellen Stadtmodellen (Bsp. [Finnet@], s. Abbildung 2, 5). Häufig werden alternative 2D-Ansichten angeboten oder 3D-Grafik mit Grundrissen bzw. Landkarten kombiniert. Die meisten Modelle sind mit VRML97 und GeoVRML [GeoVRML@] realisiert, einige Geovisualisierungen auch mit dem Format G-Vista [G-Vista@], für das ein Player vorliegt. Auch Applet-Lösungen, z.B. Shout3D, kommen zum Einsatz. Interaktivität ist primär auf Navigation und auf einfache Animationen, Feedbacks und Informationsabruf beschränkt.

2.2.5 Distributed Virtual Environments und Virtual Communities

Verteilte virtuelle Umgebungen werden hauptsächlich für Spiele, Simulationen und zur Kommunikation eingesetzt, jedoch auch für CSCW-Anwendungen und zur Kundenbindung. In [Templ99a] werden virtuelle Gemeinschaften auf Basis von DVEs im Detail untersucht. Für verteilte Multibenutzerumgebungen müssen zahlreiche technische Probleme (s.a. 2.1.2) geklärt werden. Dieses Anwendungsgebiet hat eine lange Tradition, denkt man z.B. an Klassiker wie *Cybertown* (vormals *Colony City*) [Cybertown@] oder die *Blaxxun Multiuser-Plattform* [Blaxxun@]. Dabei handelt es sich um Onlinewelten, die Nutzer häufig selbst gestalten können, wo sie sich treffen und u.a. 2D/3D-Chats erleben können. Für die meisten DVEs wird die VRML-basierte *Blaxxun Contact* Technologie [Blaxxun@] als Plug-In eingesetzt, jedoch existieren auch alternative Stand-alone-Lösungen (z.B. *moove romancer* [Moove@]). Diese Anwendungsklasse benötigt vielfältige Interaktionsformen und zusätzliche Funktionalität. Die Handlungen der Nutzer erstrecken sich von der Navigation zwischen verschiedenen Plätzen über die Informationsbeschaffung bis hin zu komplexen Tätigkeiten beim Bau virtueller Gebäude und Siedlungen. Weitere Beispiele:

- Virtuelle Jobmessen: z.B. *Jobfair24* [Jobfair@] (s. Abbildung 2, 6)
- Virtuelle Lerngemeinschaften: z.B. *Cornelsen Learnetix Community* [Learnetix@] (s. Abbildung 2, 7)
- CSCW-Bereich: 3D-Videokonferenzsysteme, militärische und medizinische Anwendungen, verteilte Simulationen (z.B. zur Waldbrandbekämpfung [CIFSC@], wobei VRML97 und GeoVRML verwendet werden, s. Abbildung 2, 8)

2.2.6 Informationsvisualisierung und Navigation

Bei Anwendungen dieser Kategorie werden abstrakte, nicht-räumliche Informationen visualisiert, wobei neben 3D-Diagrammen zumeist navigierbare Hierarchievisualisierungen anzutreffen sind. Auch die Visualisierung abstrakter Prozesse, z.B. wissenschaftlicher Theorien, Zellbildungen oder Prozeßabläufe, gehört zu dieser Gruppe, deren Anwendungen jedoch nicht sehr verbreitet sind. Manche Navigationsformen sind spezialisierte Insellösungen, die häufig auch als Applet realisiert wurden. Dazu zählt eine 3D-Navigation für das *Lufthansa Systems Network* [LSN@] (s. Abbildung 2, 9). Generischere Lösungen zur 3D-Hierarchienavigation

werden häufig mit VRML97 umgesetzt. Ein Beispiel dafür ist die Visualisierungstechnik *Collapsible Cylindrical Trees* [Dachs01c], mit der insbesondere die schnelle Interaktion und Navigation innerhalb einer mittelgroßen Hierarchie möglich ist (s. Abbildung 2, 11). Andere Technologien stellen mehrere 2D-Dokumente in einem 3D-Raum gleichzeitig dar. Der *CubicEye Viewer* [CubicEye@] ist z.B. ein Webbrowser, der navigierbare Webseiten auf den 5 sichtbaren Innenseiten eines Würfels darstellt (s. Abbildung 2, 10). Primäre Interaktionsformen bei diesen Anwendungen sind die Navigation in Hierarchien und das Auswählen von Verknüpfungen.

2.3 Internetbasierte 3D-Formate und -Technologien

Bei der Beschreibung der Anwendungsdomänen wurden bereits typische Web3D-Formate erwähnt. Zwei große Lager lassen sich dabei gegenwärtig ausmachen [Götz02]. Das sind einerseits die im Einflußbereich des Web3D-Consortiums und in Zusammenarbeit mit *Sun* bzw. der *Moving Picture Expert Group (MPEG)* entstehenden Standardformate VRML97, X3D, Java3D und MPEG-4. Diese bieten offene, teilweise XML-basierte ASCII-Formate, die mit Ausnahme von Java3D deklarativen Charakter tragen. Andererseits handelt es sich um die kommerziellen, meist binären Formate der Firmen Macromedia, Adobe und Viewpoint sowie zahlreicher anderer proprietärer Anbieter. Da eine babylonische und in ständiger Veränderung begriffene Vielfalt von über 40 Formaten existiert, kann die folgende Darstellung nur unvollständig sein und soll nur das breite Spektrum und wesentliche Vertreter aufzeigen. In den Arbeiten von Rukzio [Rukzi01] und Stündel [Stünd00] wurden 3D-Formate, Technologien und Architekturkonzepte für 3D-Web-Applikationen im Detail untersucht und miteinander verglichen. Listen und Beschreibungen von den meisten Technologien findet man zudem in [Walsh01] oder im Web unter [About3D@].

2.3.1 Standardformat VRML97

Die *Virtual Reality Modeling Language (VRML)* wurde 1997 als ISO/IEC-Standard 14772-1 [VRML97@] verabschiedet. Im Kern handelt es sich um ein textbasiertes, auf Open Inventor aufbauendes 3D-Austauschformat für statische und dynamische 3D-Szenen und -objekte mit Hyperlinks zu oder Integration von anderen Medien (Text, Sound, Pseudo-3D-Sound, Videos, Bilder). Umfangreich ist die Literatur zum ersten Web3D-Standardformat, so daß für eine detaillierte Darstellung auf die Quellen (z.B. [VRML97@], [Walsh01], [Götz02]) verwiesen wird.

VRML97 folgt dem Prinzip einer Szenengrapharchitektur und definiert 54 verschiedene Knoten, die sich in die Knotengruppen geometrische Primitive, Erscheinungsbild, Sound, Gruppierung und Transformation einordnen lassen. Für jeden Knoten werden über Felder seine Eigenschaften gesetzt und Namen und Typen von Ereignissen festgelegt, die ein Knoten generieren oder empfangen kann. 20 verschiedene Feldtypen sind insgesamt definiert. ROUTES sind Ereignispfade zwischen Sender und Empfänger und verbinden somit geeignete Felder in der Form `ROUTE NodeName.eventOutName TO NodeName.eventInName`. Sensoren stellen die Nutzerinteraktions- und Animationsprimitive in VRML dar und generieren Ereignisse, so z.B. ein *TouchSensor* nach dem Berühren der ihm zugeordneten Objekte. Erst über die Verbindung zu Feldern anderer Knoten mittels Route-Statements können Änderungen der Szene hervorgerufen werden. Interpolatoren empfangen, verarbeiten und generieren Ereignisse und können als *built-in Scripts*, aufgefaßt werden, die Berechnungen für einfache Animationen vornehmen. Skriptknoten können zwischen Ereignis-Generatoren (meist Sensoren) und Empfängern plaziert werden und erlauben die Definition eigener Funktionalität in Java und den Skriptsprachen JavaScript und ECMA-Script. Neben Skript-Knoten erlaubt das *External Authoring Interface (EAI)* einen programmtechnische Zugriff auf die VRML-Szene zur Laufzeit. Eine Form der Kapselung und Wiederverwendbarkeit ist mit Prototypen möglich. Einzelne Subszenengraphen lassen sich zusammenfassen, mit einer Schnittstelle versehen und an anderer Stelle – evtl. über Felder parametrisiert – wiederverwenden.

Das ASCII-Dateiformat mit eigener VRML-Syntax umfaßt die Deklaration der Szenenhierarchie, Route-Statements und Skripte. Für die Erstellung bieten sich einfache Texteditoren, syntaxgesteuerte VRML-Editoren (z.B. VrmlPad [VRMLPad@]) und natürlich visuelle Autorenwerkzeuge (z.B. CosmoWorlds [CosmoWorlds@]) an. In jedem Fall kann man VRML97 als Austauschformat Erfolg bescheinigen, was die zahlreichen Exportfunktionen aus gängigen CAD- und 3D-Modellierungsprogrammen beweisen. Zudem haben verschiedene Arbeits-

gruppen des Web3D-Consortiums interessante Vorschläge und Erweiterungen des Standards hervorgebracht, darunter GeoVRML [GeoVRML@], Humanoid Animation [H-anim@] oder Universal Media [UniversalMedia@].

Das Format weist jedoch auch eine Reihe von Schwächen auf, die zu seinem teilweise nur geringen Erfolg und der Entwicklung des Nachfolgeformates X3D führten. Dazu zählen die monolithische Architektur und dementsprechend großen Plug-Ins, die ungenügenden Erweiterungsmechanismen, kein ausgereiftes und objektorientiertes Programmiermodell zum Laufzeitzugriff, zu große Dateien durch fehlende Komprimierung und fehlendes Binärformat, keine Streaming-Unterstützung und nicht zuletzt eine unzureichende visuelle Qualität im Vergleich zu guten proprietären Lösungen.

2.3.2 Standardisierungsvorschlag X3D

Als Konsequenz aus den Mängeln von VRML97 entwickelte eine Arbeitsgruppe des Web3D-Consortiums *Extensible 3D (X3D)* [X3D@], vormals *VRML Next Generation (VRML-NG)* genannt. Nach mehrjähriger Spezifikationsarbeit, Implementierung und Evaluation ist das zu VRML97 voll kompatible Format inzwischen als *Final Committee Draft* 19775:200x bei der ISO/IEC eingereicht worden. Die Standardisierung ist für April 2004 geplant. Parallel dazu wird gegenwärtig für X3D noch ein Kodierungsstandard (ISO/IEC 19776) zur Festlegung der XML- und klassischen VRML-Kodierung vorgelegt. Dazu kommt außerdem noch der Sprachbindungsstandard (ISO/IEC 19777), in dem für das X3D-API *Scene Access Interface (SAI)* Sprachbindungen an ECMA-Script und Java definiert werden.

Aus dem Kodierungsvorschlag wird der wichtigste Unterschied zu VRML97 deutlich. Es handelt sich um die Kodierung mit XML, die parallel zum klassischen VRML-ASCII-Dateiformat eingesetzt werden kann und künftig auch um eine binäre Variante ergänzt werden soll. Zwar ist die Performance beim Einlesen von XML-Dateien geringer als im optimierten VRML97-Format, dafür eröffnet sich durch den Einsatz von XML eine Vielzahl von einsetzbaren Werkzeugen, Datenbanklösungen und Interoperabilität zu zahlreichen XML-basierten Medienformaten im Web. Eine enge Kooperation mit dem *World Wide Web Consortium* [W3C@] bietet erfolgversprechende Synergien. Das folgende Codebeispiel zeigt den Einsatz der X3D-XML-Syntax für einen einfachen Tisch.

```
<X3D>
    <head>…</head>
    <Scene>
        <ProtoDeclare name="Zweifarbentisch">
            <ProtoInterface>
                <field name="FarbeBeine" type="SFColor" value=".8 .4 .7" accessType="initializeOnly"/>
                <field name="FarbePlatte" type="SFColor" value=".6 .6 .1" accessType="initializeOnly"/>
            </ProtoInterface>
            <ProtoBody>
                <Transform>
                    <Transform translation="0.0 0.6 0.0"> <!-- Tischplatte --> … </Transform>
                    <Transform translation="-0.5 0.0 -0.5"> <!-- Erstes Tischbein -->
                        <Shape DEF="Bein">
                            <Appearance>
                                <Material DEF="TischbeinMaterial" diffuseColor="1.0 0.0 0.0">
                                    <IS><connect nodeField="diffuseColor" protoField="FarbeBeine"/></IS>
                                </Material>
                            </Appearance>
                            <Cylinder height="1.0" radius="0.1"/>
                        </Shape>
                    </Transform>
```

```
                <Transform translation="0.5 0.0 -0.5"> <!-- Nächstes Tischbein -->
                    <Shape USE="Bein"/>
                </Transform>
                ...
            </Transform>
        </ProtoBody>
    </ProtoDeclare>
    <!-- Der gerade definierte Prototyp wird instanziiert, wobei die Standardfarben überschrieben werden -->
    <ProtoInstance name="Zweifarbentisch">
        <fieldValue name="FarbeBeine" value="1 0 0"/> <fieldValue name="FarbePlatte" value="0 1 0"/>
    </ProtoInstance>
  </Scene>
</X3D>
```

Deutlich ist zu erkennen, wie Knoten auf XML-Elemente und Felder auf XML-Attribute abgebildet sind. Im Beispiel ist zudem der Einsatz des DEF-/USE-Konzeptes und von Prototypen als Form der Wiederverwendung zu erkennen. Für die Definition der X3D-XML-Syntax stehen eine *Document Type Definition (DTD)* und ein *XML Schema* zur Verfügung.

Ein weiterer wichtiger Unterschied zu VRML97 ist die modulare Struktur von X3D in Form der Konzepte *Components, Profiles* und *Levels*. *Profiles* sind benannte Gruppen mit bestimmter Funktionalität und konkreten Anforderungen, die Implementationen (z.B. Player) dieses Profils erfüllen müssen. Ziel war es, für Player eine genau definierte Funktionalität festzuschreiben, die Kompatibilität und Verläßlichkeit für einen Web3D-Autor gewährleistet. Sechs Profile sind gegenwärtig definiert: *Core, Interchange, Interactive, MPEG-4 Interactive, Immersive* und *Full*. Dabei umfaßt der Kern eine Minimaldefinition von nur 26 Knoten, in jedem Profil kommen weitere hinzu, zunächst für einfache Player-Implementierungen auf z.B. mobilen Endgeräten, dann mit zusätzlicher Interaktivität, MPEG-4-Kompatibilität und schließlich voller VRML97-Funktionalität. Erweiterungen und neue Profile sind möglich.

Components sind typischerweise Kollektionen von funktional in Beziehung stehenden X3D-Knoten, können jedoch z.B. auch bestimmte API-Dienste oder Kodierungen umfassen. So werden in der X3D-Komponente *Interpolators* sämtliche Interpolationsknoten zusammengefaßt. In Analogie zur Java-Programmierung kann man *Components* auch als *Packages* auffassen.

Die X3D-Spezifikation kann von Implementationen auf verschiedenen *Levels* unterstützt werden. So kann jede X3D-Komponente eigene Dienstgüte-Ebenen definieren, wobei *Levels* mit höheren Zahlen z.B. auf striktere Semantik, erweiterte Funktionalität oder Genauigkeit hinweisen. Leider ist zur Zeit noch weitestgehend ungeklärt, wie die Definition und Registrierung eigener Profile, Komponenten oder Dienstgüteebenen konkret erfolgen kann oder wie ein automatisiertes Nachladen benötigter Teil-Implementierungen in einem X3D-Player angestoßen wird.

Auch im Bezug auf die visuelle Qualität und Darstellungsgenauigkeit sind in X3D Verbesserungen vorgenommen worden, darunter die Unterstützung von NURBS-Oberflächen, Multi-Texturing, *Humanoid Animation* und *Geospatial Data*. Beispielimplementierungen für Autorenwerkzeuge (z.B. X3D-Edit [X3DEdit@]), erste X3D-Player und Applet-Lösungen (z.B. Shout3D [Shout3D@]) runden die Entwicklungen um den künftigen Web3D-Standard ab.

2.3.3 Standardformat MPEG-4

Seit 1999 ist das von der *Moving Picture Experts Group (MPEG)* [MPEG@] entwickelte MPEG-4 [MPEG-4@] der ISO/IEC 14496 Standard für die Definition, Komprimierung, Übertragung und Präsentation komplexer interaktiver Medienströme natürlicher und synthetischer Herkunft. Neben der Definition der flexiblen Schichtenarchitektur und von Mechanis-

men zur Skalierung der Medienströme je nach verfügbarem Übertragungskanal und Endgerät wird im Standard vor allem auch festgelegt, wie sich die Medien Text, Bild, Audio, Video, Vektorgrafik und 3D-Grafik repräsentieren und in audiovisuellen Szenen gemeinsam anordnen lassen.

So wurde als Beschreibungssprache zur Komposition der Medienobjekte das *Binary Format for Scenes (BIFS)* entwickelt, mit dem sich Szenenbeschreibungen getrennt von elementaren Medienströmen kodieren lassen. Die Szenengraph-basierte Beschreibungssprache entstand auf der Grundlage von VRML97. Zusätzlich zu den VRML97-Knoten wurden Knoten für zweidimensionale Objekte und Kompositionen, zur Ebenendefinition, für zusammengesetzte Texturen, zur Definition von 3D-Audio und zur Darstellung von Gesicht und Körper eines Menschen *(Facial and Body Animation)* definiert. Um auch eine textuelle Szenenrepräsentation zu ermöglichen und die Interoperabilität zu anderen XML-Medienstandards sowie den Austausch von Daten zwischen Autorenwerkzeugen zu erleichtern, wurde das *Extensible MPEG-4 Textual Format (XMT)* als XML-Beschreibung für MPEG-4 konzipiert. Dabei umfaßt XMT zwei Ebenen: das BIFS repräsentierende und zu X3D kompatible XMT-A sowie das zu SMIL interoperable XMT-Ω als höhere Abstraktion von MPEG-4-Eigenschaften aus Autorensicht. Ein wichtiges Ergebnis der Zusammenarbeit der Standardisierungsgruppen war die Akzeptanz des *X3D Interactive Profile* als 3D-Grundlage für den MPEG-4 Standard durch die MPEG. Das bereits erwähnte X3D-Profil *MPEG-4 Interactive* [MPEG-4-IP@] sorgt künftig für die Kompatibilität zwischen X3D- und XMT-A-Szenen.

Für die einzelnen Bereiche und Medien des MPEG-4-Standards wurden ebenfalls Profile festgelegt, die in ihrer Funktionalität denen von X3D vergleichbar sind. Der Standard MPEG-4 ist sehr mächtig, umfassend und vielversprechend, bedarf jedoch noch einer breiteren Unterstützung durch die Industrie, um zu einem Durchbruch zu gelangen. So existieren erst sehr wenige Player (auch nur für bestimmte Profile) von MPEG-4 und bisher kaum Autorenwerkzeuge.

2.3.4 Quasi-Standard Java3D

Mit Java3D [Java3D@] steht eine objektorientierte Klassenbibliothek zur imperativen Erstellung von skalierbaren und plattformunabhängigen 3D-Grafikanwendungen zur Verfügung. Java3D ist Bestandteil der Java-Media-Technologien, gilt somit als Standarderweiterung von Java und kann für die Programmierung von Stand-alone-Anwendungen und Web-basierten 3D-Applets verwendet werden. Dabei wird standardmäßig auf OpenGL aufgesetzt, bei der Windows-Plattform wahlweise auch auf Direct3D. Als Szenengraph-API ist Java3D architektonisch an Open Inventor und VRML97 angelehnt, nimmt jedoch eine innovative Trennung des Szenengraphen in *content branch* und *viewing branch* vor. Durch das *View Model* wird zwischen virtueller Welt – also Positionierung, Orientierung und Skalierung von Objekten und Räumen durch den Applikationsentwickler – und physikalischer Welt – also Objektdarstellung durch den Java3D-Renderer auf der Zielplattform – unterschieden. Damit können verschiedene VR-Ausgabegeräte und Stereodarstellungen flexibler angesteuert werden als mit traditionellen Kamera-basierten Modellen. Java3D stellt kein eigenes Dateiformat zur Verfügung, bietet jedoch sogenannte *Runtime-loaders* für VRML und andere 3D-Datenformate. Im Gegensatz zu VRML97 wird eine binäre Geometriekompression spezifischer Objekte durch die API selbst unterstützt.

2.3.5 Proprietäre Streamingformate

Hinter den führenden proprietären 3D-Formaten mit Streamingfähigkeit stehen inzwischen größtenteils namhafte Firmen, so Adobe beim Format *Atmosphere* [Atmosphere@], Macromedia in Kooperation mit Intel bei *Shockwave 3D* [Shockwave@] und Viewpoint in Fusion mit MetaCreations bei der *Viewpoint Experience Technology (VET)* [VET@]. Aber auch weniger bekannte Firmen haben erfolgreiche Streamingformate hervorgebracht, namentlich Cy-

core mit *Cult3D* [Cult3D@] oder Pulse mit *Pulse3D* [Pulse@]. Während Pulse3D primär für Charakteranimationen und Atmosphere für realistische 3D-Welten und virtuelle Mehrbenutzerumgebungen konzipiert wurden, sind Shockwave 3D, VET und Cult3D vorrangig auf die Sektoren E-Commerce, Infotainment und Lehr-/Lernanwendungen ausgelegt. Die Formate bieten Streaming, Kompression, und teilweise auch Geometrieskalierung an. Es existieren Exportmodule für die verbreiteten 3D-Modellierungs- und Animationswerkzeuge Maya, 3D Studio Max und trueSpace. Zusätzlich steht für jedes Streamingformat ein visuelles Autorenwerkzeug zur Verfügung, mit dem interaktive Szenen erstellt und für die Übertragung über das Internet optimiert werden können. In Unterkapitel 6.1.3 ist eine nähere Beschreibung dieser Werkzeuge zu finden. Für eine ausführliche Darstellung der einzelnen Formate wird auf die genannten Originalquellen oder auf die Arbeiten [Rukzi01] und [Götz02] verwiesen.

Die proprietären Streamingformate haben den Stand der Technik bei Web3D-Grafik vorangetrieben und Maßstäbe gesetzt. So wird bei Viewpoint eine innovative Trennung vorgenommen in XML-Dateien zur deklarativen Szenenbeschreibung und streamingfähige Binärdateien zur Speicherung von Medienobjekten in *Wavelet*-komprimierter Form. Weitere Besonderheiten der Technologie sind die Integration zahlreicher Medien (darunter Shockwave und Flash), die transparente Integration von 3D-Objekten in HTML-Seiten bzw. über den Browser-Rahmen hinaus (s. Abbildung 3, rechts) sowie eine exzellente visuelle Qualität. Diese wird durch *Fullscene Anti-Aliasing, Soft Drop Shadows, Phong Shading, Bump Mapping, MIP-Mapping* und bilineare Texturfilterung möglich. Abbildung 3 unterstreicht die für Web3D-Grafik hohe optische Qualität.

Abbildung 3: Darstellungsqualität der Formate Cult3D und Viewpoint (Demos von [Cybelius@] und [VET@])

Zu den Innovationen bei Shockwave 3D zählen die Technologien *Multi-Resolution Mesh* und *Subdivision Surfaces* zur Skalierung der Objektgeometrien für verschiedene Zielplattformen. Dazu kommen physikalische Simulationen, *Keyframe- & Bones Animation*, *Non Photorealistic Rendering (NPR)* und Partikelsysteme. Cult3D macht ebenfalls *Fullscene Anti-Aliasing* möglich und bietet neben diversen Beleuchtungsverfahren auch Schattenberechnung und spezielle Codecs für Sound- und Sprachkompression.

2.3.6 Weitere Formate und Technologien

Viele der weiteren proprietären und auch frei verfügbaren Web3D-Formate sind nicht streamingfähig, auf bestimmte Spezialanwendungen ausgelegt (z.B. Verhaltensmodellierung, Charakter- und Gesichtsanimation, Videospiele, physikalische Simulation oder einfach als 3D-Scannerformat) oder werden nach dem Internet-Boom mit seinem „me-to-Syndrom“ nicht weiterentwickelt. Sie sollen hier nur namentlich aufgeführt werden, Quellenangaben lassen sich unter [About3D@] finden. Zu den erfolgreicheren gehören ohne Zweifel AXEL [AXEL@], Virtools [Virtools@], SGDL *(Solid Geometric Design Logic)* und 3D Groove [3DGroove@]. Weitere Formate sind 3d Anywhere, 3DML, 3space, Anfy3D, Atomic3D, B3D, Hypercosm, NxView, o2c, OpenSPX, RichFX, Virtue3D, Wild Tangent, Zap3D u.a.

Interessant ist vor allem die Gruppe der Pseudo-3D-Formate, zu denen bildbasierte *(Image Based Rendering – IBR)* und Vektorformate gehören. Bekanntestes IBR-Format ist wohl *Quicktime-VR* [QTVR@], mit dem 3D-Objekte oder Szenenpanoramen auf Basis von Fotos erstellt werden können. Auch mit dem Format *Flash MX* [Flash@] sind inzwischen IBR-Renderings und 3D-Vektorformate möglich. Dazu zählt z.B. die Vektorlösung *Swift3D* [Swift3D@], mit der sehr kompakte Drahtgitter- oder einfach schattierte Renderings in Flash-Szenen integriert werden können.

Diese Methoden sind vergleichsweise kostengünstig, da Fotos oder Designskizzen als Grundlage für 3D-Modelle dienen können. Auch befördert die allgegenwärtige Verbreitung von Plug-Ins wie Flash, RealVideo oder Quicktime die Darstellung und Verbreitung solcher pseudo-dreidimensionaler Inhalte. Nachteil ist bei den IBR-Verfahren die Einschränkung auf die Bildvorlagen. Trotz möglicher Interpolationen für Blickpunktänderungen lassen sich z.B. nicht ohne weiteres Farben oder andere Eigenschaften ändern. Nachteil der Vektorformate ist ihre Beschränkung auf bestimmte Schattierungsarten.

2.3.7 Gesamtübersicht

Für existierende Formate lassen sich zahlreiche Vergleichskriterien bestimmen. Dazu zählen interne Struktur, Dateiformat, Größe, Skalierbarkeit, Erweiterbarkeit, Medienintegration, 3D-Soundfähigkeiten, Skriptprogrammierung, Streamingunterstützung, Kompatibilität zu anderen Standards, Plug-In- / Appletrealisierungen, Lizenzkosten, Verbreitung und nicht zuletzt die visuelle Qualität. In der Tabelle 1 werden zur Abrundung dieses Unterkapitels wichtige Formate anhand grundsätzlicher Kriterien miteinander verglichen. In den ersten Zeilen sind die Standardformate aufgeführt, von denen lediglich MPEG-4 ein Binärformat und Streamingmöglichkeiten bietet. Die in den unteren Zeilen aufgelisteten proprietären Formate sind alle streamingfähig und besitzen unveröffentlichte, binäre Dateiformate. Eine ergänzende Übersicht mit weiteren Kriterien ist in [Rukzi01], S. 75 zu finden.

	Kodierung					*Zugriff*			*Performance / Größe*		
	UTF-8	Binärcode	XML	Dateiendung	API	Skripts	Streaming	Skalierbarkeit	Plug-In/Applet	Standard
VRML97	x			vrml, wrl, wrz	EAI	x			Plug-In 0.5-3.2 MB	ISO/IEC 14772
X3D	x	in Arbeit	x	x3d, x3dz, x3dv, x3dvz	SAI	x	geplant	x	Plug-In 50-100 kB	ISO/IEC FCD19775
Shout3D, Blaxxun3D	x			wrl, vrml, x3d, s3d, bx3d	Shout-API	x		(x)	Applet 150/55 kB	VRML-Basis
MPEG-4 (BIFS & XMT-A)	x	x	x	mp4, mpeg	MPEG-J SG-API	Java Script	x	x	Plug-In	ISO/IEC 14496
Java3D	x	x		java, class	x				/	quasi
Viewpoint	x	x	x	mts, mtx, mtz		Java Script	x	x	Plug-In 0.3-1.85 MB	verbreitet
Shockwave3D		x		w3d, obj, dcr	Lingo	Lingo	x	x	Plug-In 3.4 MB	verbreitet
Cult3D		x		c3d,co,c3p		JavaScript, VBScript	(x)	x	Plug-In 1.1 MB	
Pulse3D		x		pwc,pws		Pulse Script, Joescript	x		Plug-In 370 kB	

Tabelle 1: Vergleich von standardisierten und proprietären Web3D-Formaten

2.4 Entwicklungsstand und Trends

Es ist in den vergangenen Unterkapiteln deutlich geworden, wie vielfältig die Aktivitäten rund um das Thema Web3D-Grafik sind. Gleichzeitig sind sie jedoch auch unübersichtlich, wenig konsistent und wirtschaftlichen Schwankungen unterworfen. Die überzogene Erwartungen wurden nach dem ersten Internet-Boom inzwischen korrigiert, und große Firmen beginnen sich in das vormals von kleineren Unternehmen besetzte Web3D-Geschäft einzumischen. Laut einer Marktstudie von Jon Peddie Associates [JPA@] werden 2007 voraussichtlich über 1 Millionen Websites 3D-Inhalte anbieten. Gleichzeitig wird die Zahl der verfügbaren 3D-Player von etwa 150 Millionen im Jahr 2003 bis zum Jahr 2007 auf 559 Millionen gestiegen sein (siehe Tabelle 2).

	2001	2002	2003	2004	2005	2006	2007
Seiten mit 3D-Inhalten	0.016	0.036	0.080	0.177	0.341	0.620	1.0
3D-Player	21.3	68.7	150.2	227.5	326.7	453.1	559.1
	Alle Angaben in Millionen.			© 2001 Jon Peddie Associates			

Tabelle 2: Prognostiziertes Wachstum der Webseiten mit 3D-Inhalten bis 2007 (aus [JPA@])

Dabei ist der wirtschaftliche Erfolg von 3D-Grafik nach Meinung von Peter Ryce (Macromedia) nicht in ihrer Nutzung um der Sache selbst willen („3D ist cool“) oder als eigener Datentyp bzw. Web-Entität zu sehen, sondern nur, um ganz konkrete Probleme zu lösen [Ryce02]. Daß der Nutzen von 3D-Grafik für E-Commerce-Anwendungen zwar unbestritten ist, ändert z.B. nichts an den nach wie vor hohen Erstellungskosten, die bedacht werden müssen. An der Sinnfälligkeit von 3D-Anwendungen auch für Lehr-/Lernanwendungen, Simulationen, Training und die Unterhaltungsbranche gibt es ebenfalls keine Zweifel. So macht Ryce fünf Erfolgsfaktoren für Web3D-Grafik aus [Ryce02]:

- Gute 3D-Inhalte *(„content matters“),*
- Skalierbarkeit, um ein breites Publikum zu erreichen, ohne jedoch nur das kleinste gemeinsame Vielfache zu bedienen,
- Interaktivität und Medienintegration *(„3D has to escape the rectangular prison“),*
- Breite Unterstützung durch die Industrie,
- Distribution, Distribution, Distribution… *(„we need real-world applications“).*

Aus Sicht marktdominierender Firmen sind Standardisierungsbemühungen natürlich ein zweischneidiges Schwert. Wie wichtig sie jedoch gerade für das World Wide Web sind, zeigen Gremien, wie das World Wide Web Consortium, Web3D-Consortium oder die Moving Picture Experts Group, mit Standards für ein multimediales und flexibles Web, darunter XHTML, SMIL, SVG, MPEG-4 und X3D. Dabei zeichnet sich die Bedeutung von XML für die Interoperabilität der einzelnen Standards deutlich ab. Künftig wird die flexible Integration verschiedener Medien – z.B. von 2D-SVG und Flashfilmen als Texturen in einer 3D-Szene – eine technische Selbstverständlichkeit sein müssen. Die Akzeptanz und Unterstützung dieser jungen Standards ist jedoch noch nicht im angestrebten Maße erreicht, was auch die Wichtigkeit von Industrie-Gremien zu ihrer Verbreitung (z.B. dem *MPEG-4 Industry Forum* [MPEG-4-IF@] oder der *Browser Working Group* des Web3D Consortiums) unterstreicht.

Ein wichtiger Erfolgsfaktor ist auch die visuelle Qualität interaktiver Echtzeitgrafik im WWW. Diese hat in den letzten Jahren kontinuierliche Verbesserungen erfahren, was natürlich auch auf die performantere Grafikhardware und breitbandigere Netzwerkverbindungen zurückzuführen ist. Neben den Qualitätsverbesserungen bei proprietären 3D-Formaten erwei-

tern aber auch VRML97/X3D-Playerhersteller den VRML-Standard um Eigenentwicklungen und neue Knoten, z.B. die Firma Open Worlds um solche für Bump Mapping, Schatten, Reflection Mapping, Spiegel, NURBS, Streaming-Unterstützung und optische Effekte, wie Lens Flares, Feuer, Rauch und Wellen [OpenWorlds@]. *Multi Resolution Meshes,* verbesserte Metall- und Transparenzdarstellungen, *Full Scene Anti-Aliasing,* zahlreiche Texturierungsarten und Stereodarstellungen runden das Repertoire des zur Zeit bei Web3D-Grafik Möglichen ab.

Auch bei Web3D-Grafik macht der Trend zu mobilen Endgeräten nicht halt. Dabei werden Spiele, Multimedia Messaging, virtuelle Charaktere und einfache 3D-Grafikanwendungen nicht nur auf Mobilfunkgeräten und PDAs realisiert, sondern auch bereits auf japanischen Mikrowellen, Kühlschränken und anderen Küchengeräten. Zahlreiche, zumeist Java-basierte 3D-Engines stehen zur Verfügung, z.B. der *Superscape Swerve* Player mit Größen von 80-200 kB [Swerve@]. Neben proprietären Lösungen wird auch bereits VRML unterstützt und künftig das *Interchange-Profile* von X3D. Ein Beispiel ist der erste VRML-Browser für mobile Endgeräte namens *Pocket Cortona* von Parallel Graphics [PocketCortona@].

Die zunehmende Vielfalt an Zielplattformen erfordert natürlich auch geeignete Autorenwerkzeuge und skalierbare, modulare Ansätze. Trotz der Existenz zahlreicher proprietärer Autorenwerkzeuge mangelt es hier noch an Unterstützung, insbesondere für die 3D-Standardformate. Aber auch bei kommerziellen Formaten sind die Erstellungskosten aufgrund von mangelnden Ansätzen zur Wiederverwendbarkeit (z.B. auch für interaktive Verhaltensbausteine) und unzureichender Unterstützung eines multidisziplinären Autorenprozesses zu hoch. Inzwischen ist die Akquisition von 3D-Daten in Form von dreidimensionalen Scans oder CAD-Modellen für die meisten Produkte Stand der Technik, zumal alle gängigen 3D-Modellierungswerkzeuge Exportmöglichkeiten nach VRML97 und in wichtige proprietäre 3D-Formate anbieten. Immer noch sind jedoch Expertenwissen, viel Handarbeit und aufwendige Optimierungen und Anpassungen nötig, um interaktive 3D-Szenen zu erstellen. Dies zeigen auch aktuelle Diskussionen in Forschungskreisen (z.B. ein internationaler Workshop zum Thema effizientes Authoring für Web3D-Grafik [Dachs03b], [X3DCPWorkshop@]). Im Kapitel 6 wird diese Thematik vertieft.

Zusammenfassend läßt sich festhalten, daß die Anwendung von Web3D-Grafik zwar bisher nicht die in sie gesetzten, sicher überzogenen Erwartungen erfüllt hat, aber ein klares Entwicklungspotential besitzt. Dabei kann die technische Seite (Grafikhardware, Netzwerkverbindung) für Standard-PCs als größtenteils zufriedenstellend gelöst betrachtet werden. Anders sieht es bei der einfachen Erstellung von Web3D-Grafik, den eigentlichen 3D-Anwendungen und ihrer intuitiven Bedienung durch unerfahrene Nutzer selbst sowie der Medienintegration und Formatinteroperabilität aus.