Gymnasium der Abtei Schlierbach

3D-Modellierung am Computer mit Blender

Fachbereichsarbeit aus Informatik und Naturwissenschaften

vorgelegt bei:
Mag. Robert Schürz

von:
Daniel Knittl-Frank

Vorderstoder, 28. Februar 2007

1 Eidesstattliche Erklärung

Ich, Daniel Knittl-Frank, erkläre hiermit, die Fachbereichsarbeit eigenhändig verfasst und nur die angegebenen Quellen verwendet zu haben.

Die Fachbereichsarbeit wurde auf Rechtschreibfehler kontrolliert.

Vorderstoder, Mi, 28. Feb 07 Daniel Knittl-Frank

2 Vorwort

Über eine Fachbereichsarbeit dachte ich schon Ende der siebten Klasse nach, ein Fach weniger bei der Matura und viel Zeit hörte sich verlockend an.

Die Themenwahl fiel anfangs ganz klar auf „Eine Community-Plattform im Internet“, da ich durch das Projekt Sigma-Schlierbach (http://www.sigma-server.com) bereits mit dem Thema vertraut war, und auch der praktische Teil der FBA großteils schon existierte, also eine Online-Plattform im Internet.

Kurze Zeit später entdeckte ich Blender – noch in der Version 2.41 – und es dauerte nicht lange, bis ich meine ersten Bilder mit Blender erschuf. Angespornt durch die Bilder von Profis in diesem Gebiet, wie sie in der Gallerie auf blender.org, oder auf blenderartists.org zu finden sind, arbeitete ich mehr und mehr mit Blender.

In den darauf folgenden Sommerferien verbrachte ich meine Freizeit fast ausschließlich vor dem Bildschirm und vor Blender – nun in der Version 2.43. So kam mir dann auch die Idee, ob es nicht besser wäre, eine FBA über Blender und 3D-Computergraphik zu schreiben, jedoch war ich noch sehr hin und her gerissen zwischen diesen beiden Themen.

Anfang des Schuljahres fiel dann meine Entscheidung zugunsten Blender, hauptsächlich dadurch bedingt, weil das Thema „PHP & mySQL“ schon so oft von anderen behandelt worden war, dass es letztendlich jeglichen Reiz für mich verlor. Zudem konnte ich mich so intensiver mit Blender beschäftigen, und gleichzeitig bei meiner FBA weiter schreiben. Ein Bild, das in den Sommerferien als Freizeitprojekt begann, wurde von mir dann auserkoren, als praktisches Beispiel zu dienen.

Zwar konnte ich meine FBA nicht ganz dem Programm Blender widmen, was ich gerne gemacht hätte, sondern musste auch 3D-Computergraphik im Allgemeinen behandeln, um den Ansprüchen einer FBA gerecht zu werden. Dies erforderte eine erneute intensive Beschäftigung mit dem Thema, aber das war nicht wirklich ein Problem.

Zur FBA selbst ist noch zu erwähnen, dass von mir aus ästhetischen Gründen viele Anglizismen verwendet werden, und Fachtermina hauptsächlich in englischer Sprache aufscheinen. Viele deutsche Übersetzungen erscheinen mir einfach zu plump und teilweise zu unpassend oder zu ungenau um eine Sache exakt zu beschreiben. Das Graphische Interface von Blender lässt sich zwar in viele Sprachen umstellen, diese sind jedoch meist unvollständig und lückenhaft.

Danken möchte ich all jenen, die mich vor, während und nach dem Schreiben dieser Arbeit unterstützt haben, und auch denen, die sich die Zeit nehmen sie zu lesen.

Ganz besonderer Dank gilt der Blender Foundation, die es erst ermöglichte dieses Programm frei verwenden zu können, und allen Entwicklern, die dieses Projekt weiterentwickeln.

Inhaltsverzeichnis

3D-Modellierung am Computer
mit Blender

3 Computergrafik und 3D-Modellierung

3.1 Definition

Computergrafik wird definiert als:

> *"The set of technologies used to create art with computers oder Art or designs created using such technologies."*[1]

Sie beschreibt also nicht nur die fertige Arbeit, sondern auch die Technologie, die eingesetzt wird, um das gewünschte Ergebnis zu erreichen.

3.2 Meilensteine

Die erste Computergrafik entstand im Jahre 1949 im MIT auf dem Whirlwind Computer, mittels des „Bouncing Ball" Programms von Charlie Adams.

Auch die Darstellung von Flugobjekten fällt unter die Kategorie der Computergrafik, dies geschah das erste Mal 1952.

1962 war es dann soweit, die ersten 3D-Computergrafiken erblickte das Licht der Welt im MIT auf dem TX2-Computer von Lawrence G. Roberts.

1963 entwickelte Ivan Sutherland das erste „Grafikprogramm", damit war es bereits möglich, ein Bild aus Standardelementen zusammenzusetzen, sowie mit einem Lichtgriffel und der Tastatur in Interaktion mit dem System zu treten. Auch konnten die Bilddaten für die spätere Verwendung gespeichert werden.

Im weiteren Verlauf der sechziger Jahre wurden die ersten kommerziellen Bildschirme produziert, die jedoch kaum erschwinglich waren. Des Weiteren wurde vermehrt im Bereich der CG (Computer Graphics) geforscht, und die bereits bekannten Techniken verbessert.

Anfang der Siebziger kamen dann die ersten CAD[2] (Computer Aided Design) und CAM (Computer Aided Manufacturing) Programme auf den Markt, es wurde das Raster-Scan-Verfahren zum Rendern von 3D-Szenen vorgeschlagen, und der Altair 8800, der erste Heimcomputer – kam als Bausatz in die Läden.

1 Vgl. http://www.answers.com/computer+graphics&r=67
2 Vgl. http://www.cad.com.au

1974 veranstaltete die ACM SIGGRAPH (**A**ssociation for **C**omputing **M**achinery **S**pecial **I**nterest **G**roup on Computer **Graph**ics and Interactive Techniques) das erste Mal die SIGGRAPH Konferenz[3], die bis heute jedes Jahr veranstaltet wurde, und seitdem jedes Jahr eine Konferenz abhält. 1999 war Blender das erste Mal auf dieser Konferenz vertreten.

Eine Ikone der Computergrafik – der „Utah teapot“ - wurde 1975 von Martin Newell modelliert, als Vorlage diente die Melitta-Teekanne seiner Frau. Diese Teekanne existiert in vielen 3D-Programmen als verwendbares Primitivobjekt, in Blender kann man sie in den Demo Dateien zur jeweilig aktuellen Version finden.

Fast zeitgleich entstanden auch viele der heute noch gebräuchlichen Algorithmen zur Berechnung von Beleuchtung (Phong, Blinn), Schattierung (Gourard, Phong), Texturierung (Catmull) und Schattenwurf (Crow).

Es begann die Zeit des 3D, die neuen Algorithmen wurden immer leistungsfähiger und besser. So dauerte es auch nicht lange, bis 1979 die ersten Spiegelungen und Lichtbrechungen mit Raytracing berechnet wurden.

Auf der SIGGRAPH'80 wurde der Film „Vol Libre“ von Loren Carpenter, der einen Flug durch eine künstliche Fraktallandschaft zeigt, vorgestellt. Er begeisterte das Publikum, wurde jedoch von der Jury disqualifiziert, weil er nicht wie eine Computer Graphik aussah.

1981 wurde von Loren Carpenter das Render System „Reyes“[4] (Render everything you ever saw) entwickelt, welches später zu RenderMan wurde, und noch immer in großen Filmproduktionen verwendet wird (Toy Story 1+2, The Incredibles, A Bugs Life, Finding Nemo, Monsters Inc., Cars, uvm.). Renderman wurde später zum Industriestandard erhoben.

1982 wurde auch die erste längere Computeranimation in dem Film „Tron“ verwendet, die zirka 30 Minuten dauert. Der Film floppte jedoch, und die Filmindustrie stand CG (Computer Graphics) in Filmen daraufhin sehr kritisch gegenüber.

Auf der SIGGRAPH'82 präsentierte Tom Brigham das erste Mal eine Filmsequenz mit Morphing, in dieser Filmsequenz verwandelte sich eine Frau in einen Luchs. Diese Technik erhielt jedoch kaum Aufmerksamkeit, bis sie 1988 im Film „Willow“ von George Lucas erneut verwendet wurde.

1982 wurde Silicon Graphics Inc. von Jim Clark gegründet, die sich darauf spezialisierte, Computer für graphische Anwendungen herzustellen.

3 Vgl. http://www.siggraph.org/publications/newsletter/v32n3/contributions/machover2.html
4 Vgl. http://renderman.pixar.com/

1983 entwickelte Jaron Lanier den ersten Datenhandschuh, Jaron Lanier gründete 1985 die Firma VLP um kommerzielle VR-Produkte (Virtual Realtiy) zu entwickeln.

1984 wurde die Radiosity Technik entwickelt, die es ermöglicht globale Beleuchtung zu simulieren.

Die bekannte Firma Pixar wurde 1986 von Ed Catmull und A.R. Smith gegründet, nachdem sich Lucas Film Ltd. aufspaltete.

1989 zeigte ILM[5] (Industrial Light and Magic) im Film „The Abyss" eine Szene mit einer computergenerierten Schlange aus Wasser, die die Gesichter anderer Menschen imitierte.

1989 wurde das Motion Capturing von Jim Henson eingeführt, welches es ermöglicht, Bewegungen aufzuzeichnen und in ein für den Computer verständliches Format umzuwandeln.

Mit „Terminator 2" (1991) wurden neue Maßstäbe im Bereich von computergenerierten Spezialeffekten gesetzt, in ihm wurde die Flüssigmetall-Oberfläche des T-1000 mit dem Computer animiert.

In „Jurassic Park" (1993) wurden alle Dinosaurier computeranimiert, anstatt, wie anfänglich geplant, durch Puppen dargestellt.

1995 erschien „Toy Story", der erste komplett mit Computern generierte Film von Pixar. Für das Rendern wurden 177 Sun Sparc Computer verwendet, ein Frame zu berechnen dauerte zwischen 45 Minuten und 2 Stunden.

2001 wurde im Film „Shrek" die Gesichtsanimation bei den künstlichen Charakteren sehr gut umgesetzt. Im selben Jahr erschien auch noch „Final Fantasy: Die Mächte in dir", der nur am Computer erzeugt wurde, und realistisch wirkende menschliche Charaktere einführte. Der Entwicklungsprozess dauerte 4 Jahre und beschäftige 170 Animatoren.

3.3 Verwendung

Computergrafik wird heutzutage fast ständig verwendet, im CAD/CAM Bereich, in Computerspielen, in Werbetrailern, in Kinofilmen, in der Medizin zur Visualisierung von klinischen Daten. Schier überall kann man sie erblicken und wird sie auch erblicken, jede Homepage und Computerprogramm ist mit Grafiken geschmückt.

5 Vgl. http://www.ilm.com/

3.4 Wichtige Begriffe

Die folgende Liste von Begriffen ist bei weitem nicht vollständig und soll nur einen kurzen Überblick verschaffen. Eine längere Liste ist im Anhang zu finden.

3.4.1 Modellieren[6]

Die Szene wird mit Hilfe entsprechender Software meist von Hand erstellt, kann aber auch vollständig durch den Computer erzeugt worden sein, zum Beispiel bei der Computertomografie. Als Modellieren wird der Vorgang bezeichnet, in dem die 3D-Daten erstellt werden.

3.4.2 Rendern[7]

Als Rendern wird der Prozess bezeichnet, in dem die 3D-Daten in ein zweidimensionales Bild „umgerechnet" werden. Dazu werden zuerst alle Flächen indiziert, Normalvektoren für die Beleuchtung berechnet, Materialien zugewiesen. Bei komplexen Szenen kann dieser Vorgang, auch abhängig von der verwendeten Renderengine und -technik und den eingesetzten Shadern einige Stunden bis Tage dauern. Die entstandenen Bilddaten werden anschließend als Rastergrafik in eine Grafikdatei geschrieben.

Dieses Bild kann dann unabhängig von der installierten Software auch auf anderen PCs betrachtet werden. Es handelt sich nur mehr um ein Standardgrafikformat[8], wie zum Beispiel JPEG, GIF oder PNG. Mehrere Bildsequenzen können als Video gespeichert werden, wiederum in einem softwareunabhängigem Format, bekannte Formate sind AVI, MPEG und MOV.

3.4.3 World

Die Welt ist das Superobjekt, das alle anderen Objekte beinhaltet, sowie auch die globalen Einstellungen wie Himmelsfarbe und Himmelstextur.

3.4.4 Object

Ein Objekt ist vergleichbar mit einem Container, der Informationen enthält. In unserem Fall also ein Container für das Mesh, Modifier, Constraints und in einigen speziellen Fällen auch für das Material.

6 Vgl. http://de.wikipedia.org/wiki/3D-Computergrafik#Modellierung
7 Vgl. http://de.wikipedia.org/wiki/3D-Computergrafik#Rendern
8 Vgl. http://de.wikipedia.org/wiki/Grafikformat#Liste_von_Dateiformaten_f.C3.BCr_Rastergrafiken

3.4.5 Material

Das Material bestimmt die Farbe, die Reflexionseigenschaften, Transparenz, Textur, und vieles mehr, was mit dem Visuellen zu tun hat.

3.4.6 Shader

Shader bestimmen, wie stark und in welcher Form das auf Flächen auftreffende Licht gebrochen, reflektiert und absorbiert wird.

3.4.7 Mesh

Das Mesh bestimmt die Form des Objekts, festgelegt durch Vertices, Edges und Faces. Weiterhin sind Materialien standardmäßig mit den Faces eines Meshes verknüpft.

3.4.8 Faces: Polygone, Triangles

Faces sind die Flächen zwischen mindestens drei Vertices. Beim Rendern wird anhand der den Flächen zugewiesenen Materialien ein zweidimensionales Bild berechnet.

3.4.9 Edges

Edges sind Kanten, die sich zwischen zwei Vertices aufspannen.

3.4.10 Vertices

Ein Vertex ist ein Punkt im 3D-Raum mit einer festgelegten Position.

3.5 Techniken

3.5.1 Box-Modelling und Polygon-Modelling

Beim Polygon-Modelling werden einem Modell einzelne Polygone hinzugefügt, im Gegensatz dazu steht das Box-Modelling, bei dem von einem Würfel (Box) ausgehend, durch Extrudieren von einzelnen Flächen oder Flächenverbänden das gewünschte Ergebnis erreicht wird.

3.5.2 Parametrisches Modellieren

Beim parametrischen Modellieren wird die Geometrie bemaßt, und lässt durch die geschlossene Oberfläche eine eindeutige Definition von Innen und Außen zu.

Geometrie kann auch durch Bool'sche Operationen definiert werden, indem Grundkörper voneinander abgezogen oder miteinander kombiniert werden.

3.5.3 Andere Arten des Modellierens

Einen Spezialfall stellt das Modellieren mit NURBS, Splines oder anderen Kurventypen dar, hier übernimmt das 3D-Programm das „Interpolieren“ der Flächen, der Anwender gibt dem Programm lediglich einige Ankerpunkte vor. Durch Parameter kann nachträglich die entstandene Geometrie geändert werden, unter anderem der Feinheitsgrad des Meshes, oder wie Endpunkte gehandhabt werden.

Auch unter diese Kategorien fällt das Modellieren mit Metaballs. Metaballs oder allgemein Metaelemente sind mathematische Beschreibungen von Primitiven, die sich gegenseitig beeinflussen, sie können miteinander verschmelzen oder sich gegenseitig eindrücken und auflösen, dies geschieht durch positiv oder negativ wirkende Metaballs.

3.5.4 Low-Poly-Modelling und High-Resolution-Modelling

Es wird auch unterschieden zwischen Low-Poly-Meshes (oder -Modelling) und High-Resolution-Meshes (oder -Modelling), erstere eignen sich besonders für Computerspiele und zeichnen sich durch schnelle Verarbeitung durch den Prozessor aus, High-Resolution-Meshes werden für fotorealistische Renderings oder ähnliches verwendet. Diese sind auch in der Erstellung total unterschiedlich, beim Low-Poly-Mesh wird versucht mit möglichst wenig Details ein Objekt erkennbar zu machen, beim High-Resolution-Mesh ist genau das Gegenteil der Fall, es wird versucht ein Objekt möglichst originalgetreu nachzubilden.

Bei Low-Poly-Meshes wird meistens mit Texturen ein zusätzlicher Detailgrad hinzugefügt, so dass oft gar nicht auffällt, wie simpel die Modelle gehalten sind.

4 Die Open-Source Software Blender

4.1 Geschichte[9]

Im Jahre 1988 wurde in den Niederlanden das Animationsstudio „NeoGeo“ gegründet, welches schnell das größte in den Niederlanden wurde. NeoGeo verwendete intern entwickelte Animationsprogramme. 1995 wurde beschlossen, diese Programme von Grund auf neu zu schreiben, eine Aufgabe, welcher sich Ton Roosendaal annahm, der zugleich Art Director war. Diese Anwendung kennen wir heute alle unter dem Namen Blender, welcher nun von der Firma NaN (Not a Number) unter der Leitung von Ton Roosendaal weiterentwickelt wurde. NaN wollte ein plattformunabhängiges, frei erhältliches und kompaktes 3D Paket („*compact, cross platform 3D tool for free*“) entwickeln und vertreiben. Die Einnahmen der Firma wurden durch Support, kommerzielle Produkte und Services um und für Blender eingenommen.

1999 wurde Blender auf der SIGGRAPH bekannt gemacht, und erhielt viel Aufmerksamkeit, sodass die Entwicklung von Blender im Jahr 2000 mit 4,5 Millionen Euro unterstützt wurde. Dies führte zu einem starken Entwicklungsschub von Blender, NaN bestand nun aus 50 Angestellten, und im Sommer 2000 erschien Blender in der Version 2.0, die eine integrierte Game Engine besaß. Weiters stieg die Zahl der registrierten Benutzer der Software auf über 250 000.

Leider überschätzte NaN die Möglichkeiten, woraufhin NaN 2001 neu gegründet wurde, diesmal kleiner und mit neuen Investoren. NaN brachte dann die erste kommerzielle Umsetzung von Blender, den „Blender Publisher“, auf den Markt, die aber nicht den erhofften Erfolg brachte. Sie sollte speziell auf den interaktiven Internetmarkt zielen. Niedrige Verkaufszahlen und die noch immer schwierige wirtschaftliche Lage zwang die Investoren kurze Zeit später dazu NaN nicht weiter zu unterstützen und die Firma zu schließen, was auch den Entwicklungsstop von Blender zur Folge hatte.

Doch obwohl Blender eine unübersichtliche innere Struktur, viele unfertige Features hatte und ein eigenartiges GUI bot, war es der Community und den Käufern von Blender Publisher zuwider, Blender in Vergessenheit geraten zu lassen. Die erneute Neugründung einer ausreichend großen Firma war nicht mehr möglich, und so gründete Ton Roosendaal im März 2002 die Stiftung „Blender Foundation“, deren Hauptziel es ist, Blender wieder zu entwickeln. Sein Vorhaben stellte Ton Roosendaal im Juli 2002 vor und überzeugte die ehemaligen Investoren von NaN, das Projekt zu

9 Frei nach http://www.blender.org/blenderorg/blender-foundation/history/

bewilligen: Blender unter eine Open-Source Lizenz zu stellen, jedoch nur wenn die Blender Foundation bereit wäre 100 000€ zu zahlen, um das geistige Eigentum an Blender zu erkaufen. Viele begeisterte Freiwillige und auch viele der ehemaligen NaN Angestellten spendeten Geld und so wurde die 100 000€ Marke innerhalb kürzester Zeit erreicht, nach 7 Wochen – am 13 Oktober 2002 – wurde Blender erschreckender- und überraschenderweise unter die GNU-GPL gestellt („*To everyone's shock and surprise the campaign reached the 100,000 € goal in only seven short weeks*“).

Heute wird Blender von einer Vielzahl internationaler Entwickler, noch immer unter der Leitung von Ton Roosendaal ständig verbessert und erweitert. Durch die GPL kann Blender sowohl privat als auch kommerziell benutzt werden, ohne dafür bezahlen zu müssen, die Software kann auch verändert und weiterverteilt werden, wenn dadurch die GPL nicht verletzt wird. Insbesondere muss der Quelltext auch veröffentlicht, beziehungsweise auf Anfrage zur Verfügung gestellt werden.

4.2 Die GPL[10]

> *“…When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things…”*
>
> *[GNU GENERAL PUBLIC LICENSE Version 2, June 1991]*

Die GNU General Public License ist eine Lizenz, welche ausschließlich für Open-Source-Projekte verwendet wird. Der größte und wichtigste Unterschied zu anderen Lizenzen besteht darin, dass durch ihre Verwendung sichergestellt wird, dass bei Wiederverwendung des Quelltextes, das daraus resultierende Projekt wieder durch die GPL lizenziert werden muss, der Quelltext muss somit weiterhin frei verfügbar bleiben.

Die GPL schließt aber keinesfalls die kommerzielle Verwendung von Programmen aus, sie ist sogar ausdrücklich erlaubt.

10 Vgl. http://www.gnu.org/copyleft/gpl.html

4.3 Anwendungsbereiche

Blender findet in den verschiedensten Bereichen Anwendung, klarerweise in der Computergrafik. Mit Blender können 3D-Modelle erstellt, bearbeitet und gerendert werden, sowohl als statisches Bild als auch in einer Animation. Aber auch Filme und Bilder können bearbeitet und Spiele programmiert werden. Seit kurzem ist es auch möglich Simulationen zu kreieren: Softbodies und Fluids sind hier bekannte Begriffe. Mit diesen ist es möglich gummiähnliche Körper und Flüssigkeiten darzustellen, die mit der Umgebung interagieren.

Ich werde mich am Ende dieses Textes mit dem Erstellen und Rendern von 3D-Modellen beschäftigen.

4.4 Blender im Vergleich zu anderen Softwarepaketen

Es gibt viele Programme um 3D-Szenen zu erstellen und zu rendern, Blender ist nur eines davon. Bekannte Namen sind: 3ds Max, Maya, Cinema 4D, Rhinoceros und viele mehr[11]. Viele davon kosten mehrere tausend Euro.

Deshalb ist auch ein ganz klarer Pluspunkt von Blender, dass es ein Open-Source Projekt und infolgedessen frei erhältlich ist. Der Quelltext kann jeden Tag neu bezogen werden, vorkompilierte Testversionen erscheinen zirka einmal in der Woche auf graphicall.org und einmal im Monat auf blenderbuilds.com. Auch können auf der Seite projects.blender.org (der Entwicklerseite) gefundene Bugs im Bugtracker gemeldet werden, oder von anderen Benutzern und Entwicklern gemeldete Bugs untersucht und behoben werden. Weitere Informationen sind im Kapitel 4.6 Entwicklung zu finden.

Ganz anders im Vergleich zu anderen Programmen ist das grafische Interface von Blender. Es ist nicht statisch, sondern unglaublich dynamisch und praktisch. Im Vordergrund der Entwicklung stand die Optimierung des Workflows und nicht die Anwenderfreundlichkeit, da Blender anfangs noch ein firmeninternes Softwarepaket war. Doch hier divergieren die Meinungen unter den Anwendern ziemlich stark, ob und wie das Interface verbessert werden kann, oder wie Neueinsteigern der Zugang erleichtert wird. Nun wird sogar diskutiert[12] zwei Interfaces mitzuliefern – ein „Anfänger“ und ein „Fortgeschritten“-GUI. Nichtsdestotrotz steht in nächster Zeit eine Überarbeitung des ganzen GUI-Konzepts ins Haus, hierzu führt das Kapitel 4.7 GUI weiter.

11 Vgl. http://de.wikipedia.org/wiki/3D-Grafik-Software
12 http://www.blender.org/forum/viewtopic.php?t=10540

Ein Problem ist die unvollständige und oft nicht aktuelle Dokumentation, besonders zu neuen Entwicklungen finden sich selten ausführliche Beschreibungen. Auf der Website mediawiki.blender.org wird versucht, diesem Problem entgegenzuwirken.

4.5 Besonderheiten

Blender verwendet nicht wie andere Programme einzelne überlappende Fenster, und besitzt einen eigenen Window-Manager der auf OpenGL aufbaut, siehe 4.7 GUI. Das Hauptfenster kann bis zu 64 Mal unterteilt werden, und die einzelnen Unterfenster können schnell und einfach in ihrer Funktion geändert werden. So kann das Hauptfenster schnell nach eigenen Wünschen und je nach geforderter Aufgabe umstrukturiert werden. Das Layout wird dann mit der Datei gespeichert und beim nächsten Öffnen der Datei auf Wunsch wieder hergestellt.

Die Art wie Dateien intern strukturiert sind, ist seit den ersten Versionen kaum geändert worden. Schon damals wurde versucht die Dateien sowohl vorwärts, als auch rückwärtskompatibel zu halten. Dieses Konzept ist in meinen Augen gut gelungen, Dateien lassen sich ohne Probleme mit neueren und älteren Version bearbeiten und wieder abspeichern. Natürlich gehen manche Neuerungen dadurch verloren, wenn sie von der verwendeten Version noch nicht unterstützt wurden, die Datei bleibt jedoch weiterhin verwendbar.

Dies wird erreicht durch eine so genannte SDNA, eine Struktur, die an das Ende einer jeden Datei angehängt wird. Sie wird beim erneuten Öffnen mit der SDNA der aktuellen Version verglichen, um zu wissen, um welche Version es sich bei der Datei handelt. Durch diese äußerst ausgeklügelte Methode ist es möglich jede Art von Information in einer .blend-Datei zu speichern, auch Bilder, Videos und vieles mehr, ohne die Datei in älteren Versionen unbrauchbar zu machen.

4.6 Entwicklung[13]

Blender wird von vielen unabhängigen Privatpersonen, die über die die ganze Welt verstreut sind, verbessert und weiterentwickelt. Die Einnahmen des Blender e-shops und Spenden ermöglichen es Gründer Ton Roosendaal, seine ganze Zeit der Entwicklung Blenders zu widmen, aber auch viele Treffen zu organisieren oder auch Dokumentationen schreiben.

13 Vgl. http://blender.org/development/

Jeder, der will und genug Erfahrung hat, kann natürlich auch mitarbeiten, es werden ständig fähige Leute gesucht[14].

Dazu muss zuerst das CVS-Repository ausgecheckt werden, eine genaue Anleitung dazu liegt auf http://www.blender.org/development/building-blender/getting-sources/ vor. Mit der lokalen Kopie des Quelltextes kann jetzt nach Belieben experimentiert werden, ohne die offiziellen Quellen zu beschädigen. Auch jemand der Blender nur benutzen, und den Quelltext nicht verändern will, kann von dieser Methode profitieren, die aktuellen CVS-Dateien können selbst kompiliert werden, sodass man immer eine aktuelle Entwicklerversion besitzt, und neue Features testen kann (siehe http://www.blender.org/development/building-blender/). Jedoch wird ausdrücklich darauf hingewiesen, dass dies eine Entwicklerversion sei und nicht in produktiven Umgebungen eingesetzt werden soll, da manche Entwicklungen jederzeit wieder verworfen werden könnten oder unbekannte Bugs Arbeitsdateien unbrauchbar machen könnten.

Wer über die fortlaufenden Entwicklungen im Quelltext interessiert ist, oder nur wissen will, worüber die Entwickler gerade diskutieren, für den empfiehlt sich eine Mailinglist zu abonnieren. Eine vollständige Liste der für Blender verfügbaren Mailinglists ist auf http://projects.blender.org/mailman/listinfo vorhanden.

4.7 GUI[15]

Die graphische Oberfläche wurde mit der Idee entworfen, sich nicht überlappende Teilbereiche zu entwickeln. Sie baut vollständig auf OpenGL auf, und ist somit unabhängig vom Betriebssystem, solange dieses und die Grafikkarte OpenGL unterstützt. Ein weiterer Pluspunkt der Verwendung von OpenGL ist, dass sich die einzelnen Unterfenster schnell und zentral verwalten lassen, ohne auf den Windowmanager des Betriebssystems zurückgreifen zu müssen.

Blender verwendet intern einen eigenen Windowmanager, dieser ist zur Zeit wie folgt aufgebaut: Auf oberster Ebene befindet sich der „Screen“, der unter normalen Umständen das „Window“ eines Betriebssystems beschreibt, in dem Blender ausgeführt wird, wobei dieser Screen weiters in „Areas“ unterteilt ist. Diese Bezeichnungen wurden hauptsächlich gewählt, weil das Wort „Window“ viel zu leicht zu Verwirrungen führen würde.

14 Vgl. http://blender.org/development/current-projects/project-openings/
15 Vgl. http://blender.org/development/architecture/window-manager/

Blender verwendet noch zwei weitere Wrapper, insgesamt sind es dann fünf Ebenen von Fenstern, die aber dringend überarbeitet werden sollen, beziehungsweise mitten in einem Überarbeitungsprozess sind. Zurzeit setzen sie sich wie folgt zusammen:

System Window <-> GHOST_Window <-> Window <-> bWindow <-> ScrArea

Das Ziel in nächster Zeit ist es aber einen Aufbau zu erhalten, der ähnlich diesem ist:

System Window <-> GHOST_Window <-> Screen

Weiters muss das Farbschema der Icons vereinheitlicht werden. Es wird versucht einheitliche Farben für Icons zu finden, die vom Sinninhalt her zu einer gemeinsamen Gruppe gehören. So sollen zum Beispiel modifizierende von erzeugenden Icons anhand der Farbe gleich unterschieden werden können.

Die Areas besitzen alle eine eigene Event Queue, welche Ereignisse wie zum Beispiel Tastendrücke oder Mausbewegungen verarbeitet, oder Funktionen aufruft. So kann jede Area einzeln behandelt und neu gezeichnet werden, was sich in einem Geschwindigkeitsgewinn bemerkbar macht.

Um die Behälter, in diesem Fall die Areas von dem eigentlichen Inhalt zu trennen, gibt es auch noch „Spaces“. Dadurch wird es möglich, in der selben Area unterschiedliche Arbeiten zu verrichten. Der Typ der Area kann aus einer Dropdownliste gewählt werden, die Eigenschaften des vorherigen Typs bleiben aber erhalten. So kann man zum Beispiel eine Datei speichern und nachher wieder zur 3D-Ansicht zurückkehren, ohne die Orientierung der Ansicht zu verlieren.

Weiterhin kann jeder Space lokal, ohne andere Fenster laufen, er bildet eine abgeschlossene Einheit. Ausnahmen bestätigen die Regel, der Outliner und das Buttons Window sind abhängig vom 3D-Fenster und senden auch Events an dieses.

Noch zu wenig dokumentiert und auch kaum verwendet ist die Möglichkeit, andere Applikationen in einem Space als Plug-in zu starten.

4.8 Quicklinks - Blender in the Media

Blender ist gar nicht so selten in den Medien aufzufinden, wie ich anfangs glaubte, einmal im Monat wird auf blendernation.com eine Zusammenfassung von Blenders Medienauftritten veröffentlicht. So wird aufgezeigt, wo Blender verwendet und oftmals auch gelobt wird.

Besondere Medienaufmerksamkeit erhielt Blender durch das Open-Movie-Projekt „Orange“[16], welches den Film „Elephants Dream“ animierte, der fast ausschließlich mit Open-Source-Software produziert wurde und Blender als Hauptanwendung einsetzte. Dieser 10-Minuten-Kurzfilm wird in letzter Zeit immer öfters für Technik-Präsentationen oder zum Testen von neuen Techniken verwendet, so war er auch der erste Film, der in Europa auf HD-DVD erhältlich war.

Durch Spendenaktionen[17] sieht man oft den guten Geist, der unter den Blender-Benutzern lebt. So wurden in einer Woche 4120$ für OXFAM gesammelt, und für einen Klassenraum und diverse Hilfsmittel aufgewendet.

Gerade vor kurzem erst kam auch ein Buch auf den Markt, welches Charakter-Animation in Blender behandelt. Es trägt den Titel „Introducing Character Animation with Blender“[18] und wurde von Tony Mullen verfasst, die Blender Community erhofft sich durch die Verbreitung dieses Buches eine weitere Steigerung des Bekanntheitsgrads von Blender.

4.9 Zukunft

Die Geschwindigkeit, mit der Blender entwickelt wird, nahm vor allem in den letzten Jahren dramatisch zu, nicht zuletzt wegen der oben genannten öffentlichen Auftritte.

Immer mehr Leute sind interessiert an der Philosophie von Open-Source-Projekten im Generellen, speziell aber an Blender, da es doch eines der wenigen gratis erhältlichen 3D-Softwarepakete ist. So entschließen sich immer mehr Entwickler einen Beitrag zu leisten und Blender zu verbessern.

Durch andere Projekte wie Google Summer of Code[19, 20] wird die Entwicklung von Blender zusätzlich unterstützt und beschleunigt. Auch durch Sponsorengelder der EU[21] könnte die Entwicklung angespornt werden, jedoch geriete Blender dadurch in Abhängigkeit anderer, die meisten möchten Blender aber weiterhin frei und unabhängig halten.

16 http://orange.blender.org, http://elephantsdream.org
17 Vgl. http://www.blendernation.com/2007/01/03/blendernation-birthday-present-a-whole-school/
18 Vgl. http://www.blendernation.com/2007/02/22/introducing-character-animation-with-blender-has-hit-the-shelves/
19 Vgl. http://google-code-updates.blogspot.com/2007/02/speaking-of-summer.html
20 Vgl. http://projects.blender.org/pipermail/bf-committers/2007-February/017532.html
21 Vgl. http://projects.blender.org/pipermail/bf-committers/2007-February/017423.html

Doch sollte die Entwicklung weiterlaufen, wie sie jetzt läuft, können wir uns Großes erhoffen. Und auch dadurch, dass die Heimcomputer immer leistungsfähiger werden, können auch immer leistungsfähigere Algorithmen in Blender eingesetzt werden, die den Arbeitsaufwand reduzieren und Bilder immer realistischer rendern lassen.

5 Praktische Beispiele

5.1 Idee

Diese Idee war mein erstes größeres Projekt mit Blender, und mit diesem Bild machte ich auch beim Splash-Screen-Contest[22] auf blenderartists.org mit, der für jede neue Version von Blender veranstaltet wird.

Der nächste Abschnitt behandelt das Erstellen dieser einfachen 3D-Szene.

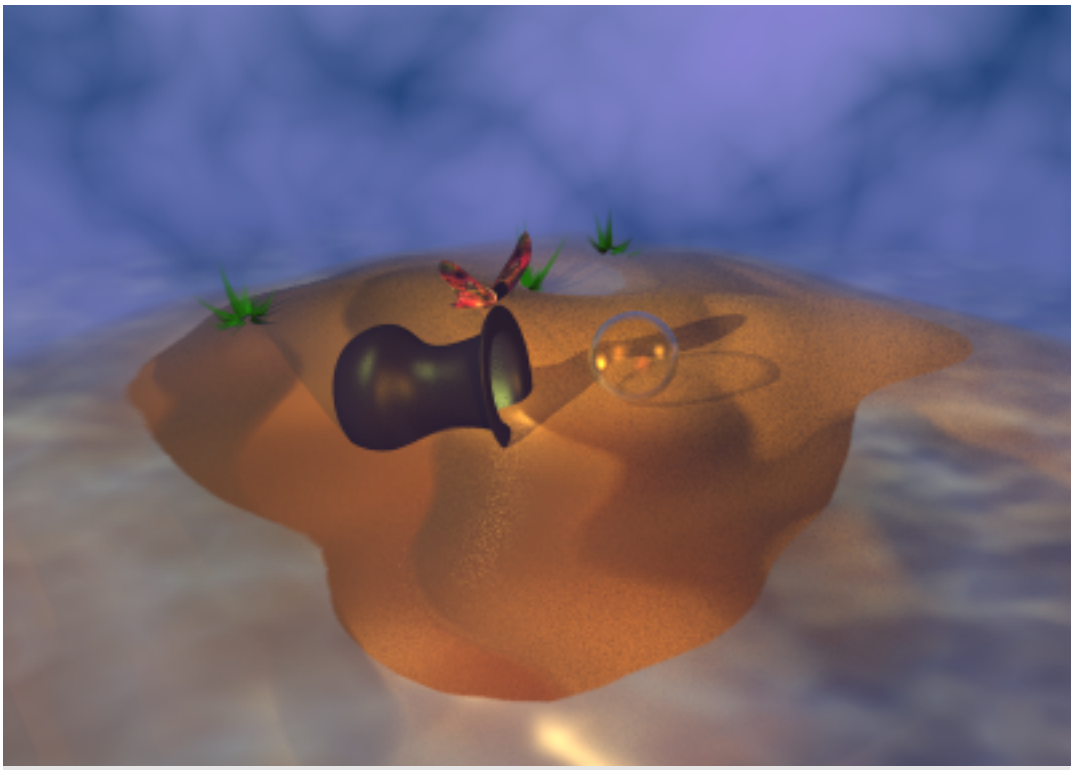

Abbildung 1: Das fertig gerenderte Bild

5.2 Vase auf einer einsamen Insel

So ähnlich wird unser Endergebnis aussehen, gerendert mit dem blenderinternen Raytracer. Ich werde Materialien und Texturen, Beleuchtung, Partikel, Nebel und Transparenz erklären.

Zu den einzelnen Arbeitsschritten ist auf der beigelegten CD jeweils eine .blend-Datei enthalten. So kann man seine Arbeit zu jedem Zeitpunkt kontrollieren, und im Notfall mit einer der mitgelieferten Dateien weiterarbeiten.

22 http://blenderartists.org/forum/showthread.php?t=81725

5.2.1 Modellieren

Zuallererst muss natürlich die entsprechende Geometrie modelliert werden, dies werde ich in den folgenden Schritten erklären.

5.2.1.1 Die Vase: Modellieren mit einem Profil

Angefangen wird mit der Vase, welche durch Rotation eines Profils erzeugt wird. Nachdem ein neues Projekt begonnen wurde, wird mittels TAB und selektiertem Würfel in den Edit Mode geschaltet.

Um mit dem Profil anzufangen wird mit W das Spezialmenü aufgerufen und *Merge > At Center* ausgewählt.

Dieser so entstandene einzelne Vertex wird zu dem gewünschten Profil mit E extrudiert. Das Profil sollte nicht größer als zwei Blenderunits sein.

5.2.1.1.1 Rotieren des Profils

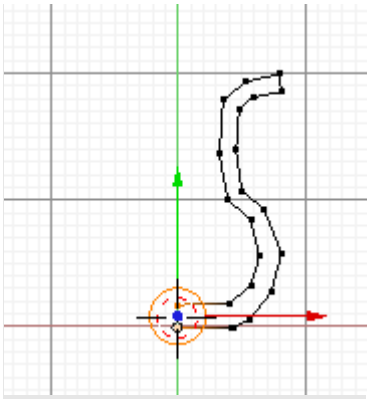

Abbildung 2: Das fertig extrudierte Profil

Der letzte und erste Vertex werden nun ausgewählt (RMT) und mit den Tasten S > X > 0 in X-Richtung ausgerichtet. Durch Drücken von SHIFT+S und Auswählen von Cursor -> Selection wird auch noch der 3D-Cursor in die Mitte unserer Vase gesetzt.

Danach werden durch zweimaliges Betätigen von A alle Vertices ausgewählt, schließlich soll das ganze Profil rotiert werden, und nicht nur die Endvertices. Zusätzlich wird mit 1 auf dem Ziffernblock in die Front-Ansicht umgeschaltet und mit A werden alle Vertices ausgewählt, denn ein Profil wird immer normal zur Ansichtsebene rotiert.

Nun wird es Zeit unser Profil zu rotieren und einen Körper daraus zu machen. Per Druck auf F9 erscheint im Buttonswindow der Editingkontext. In unserem Fall ist das Panel Mesh Tools interessant. Hier finden sich Werte für Degrees und Steps und einige weitere Buttons.

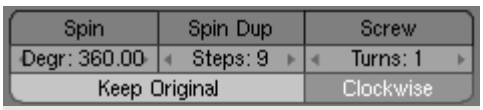

Abbildung 3: Spineinstellungen

Für die Vase stellen wir Degrees auf 360 und belassen Steps auf 9. Noch ein Klick auf Spin und im 3D-Fenster befindet sich eine Vase, die zwar noch etwas kantig aussieht, aber das wird

sich nach ein paar Handgriffen ändern. Vorerst sind aber noch zwei Dinge wichtig: mit A alle Vertices auswählen und anschließend mit W das *Specials*-Menü aufrufen und mit *Remove Doubles* doppelte Vertices entfernen, denn durch die 360 Grad Rotation befinden sich an der ehemaligen Postion des Profils die selben Vertices ein zweites Mal. Mit allen Vertices noch immer ausgewählt werden die Flächennormalen nach außen gerichtet, indem man CTRL+N drückt und anschließend die Frage mit einem Klick (LMT) oder RETURN bestätigt. Dies ist wichtig, um später keine Artefakte zu erhalten.

Der Edit Mode kann jetzt wieder mit TAB verlassen werden. Eine glatte Oberfläche erhalten wir, indem wir im *Links and Materials* Panel den Button Set Smooth klicken, und den Anzeigemodus für dieses Objekt (oder für dessen Flächen) auf „Gouraud“ umstellen, so dass durch Interpolation der Vertexfarben der Eindruck von verrundeten Flächen entsteht.

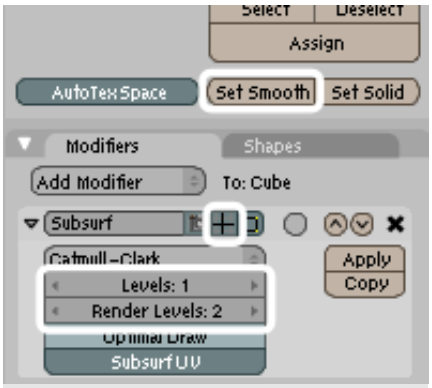

Abbildung 4: Subsurfeinstellungen

Die Silhouette bleibt dadurch aber kantig, deshalb wird zusätzlich im *Modifiers Panel* der Subsurf Modifier hinzugefügt, der jede Kante halbiert, und die entstehende Geometrie abrundet. Die Stärke kann mit zwei Reglern eingestellt werden, einer steuert die interaktive 3D-Ansicht und einer den Rendervorgang. Je höher der Wert, um so runder wird ein Objekt, jedoch bedeutet dies auch, dass die Renderzeiten mit zunehmendem Wert explosionsartig zunehmen, besonders für viele Objekte, die so verrundet wurden. Für die 3D-Ansicht kann der Wert ohne Bedenken auf 1 gelassen werden oder mit dem kleinen Button in der Mitte kann das Subsurfing während des Arbeitens ausgeschaltet werden.

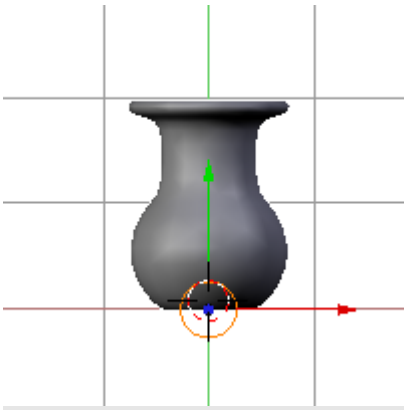

Abbildung 5: Subsurf Modifier und Set Smooth

5.2.1.2 Die Insel: Modellieren mit Noise

Nun fügen wir ein Grid mit SPACE > *Mesh* > *Grid* ein und bestätigen die Einstellungen jeweils mit einem Klick. Diesem Mesh wird nun sogleich ein Material zugewiesen, F5 öffnet die entsprechenden Einstellungen. Mit Add New erscheinen sofort die weiteren Materialeinstellungen. Mit F6 wird in den Texturenkontext geschaltet, mit Add New wird auch hier eine neue Textur eingefügt, und im Dropdownmenu wählen wir Clouds aus. Die Standardeinstellungen sind für unser Beispiel völlig ausreichend.

Zurück in den Editingbuttons (F9) wird nun im *Mesh Tools Panel* der Button Noise zirka drei Mal geklickt. Hierbei ist es ratsam eine perspektivische Ansicht vom Mesh zu haben, die Ansicht kann mit der mittleren Maustaste (MMT) gedreht werden.

Das entstandene Gitter wird nun auf die achtfache Größe skaliert (S > 8). Auch hier ist wieder ein Subsurf Modifier zu empfehlen, sowie die Set Smooth Option.

Nachdem eine Insel normalerweise nicht flach ist, sondern aus dem Wasser schauen soll, selektieren wir einen der Vertices in der Mitte. Mit O wird auf Proportional Editing umgeschaltet, auf nicht ausgewählte Vertices werden so durch einen wählbaren Algorithmus indirekt proportional zum Abstand von den ausgewählten Vertices Operationen durchgeführt, in unserem Fall das Bewegen von Kontrollpunkten.

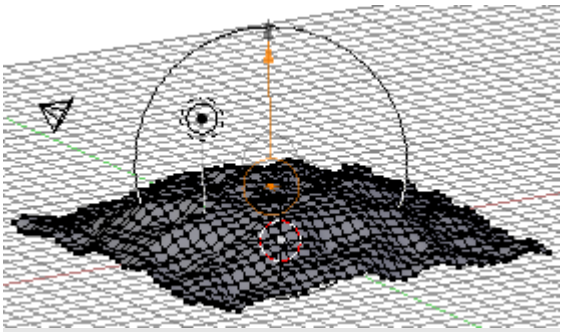

Abbildung 6: Der Wirkkreis von Proportional Editing

Der ausgewählte Vertex (RMT) wird nun mit dem Widget um 3 Blenderunits nach oben gezogen, während der nun sichtbare Kreis, der die Stärke des Effekts darstellt, fast die ganze Insel umschließen sollte. Seine Größe kann mit dem Mausrad verändert werden. Gleich nach Abschluss dieser Operation wird der noch immer ausgewählte Vertex wieder nach unten bewegt, diesmal aber nur um eine Blenderunit, und der Kreis sollte diesmal nur die vorher entstandene Kuppel umschließen.

Nach Abschluss dieses Vorganges sollte Proportional Editing wieder deaktiviert werden (O), da es sonst später zu unerwünschten Effekten kommen kann.

5.2.1.3 Das Meer

Das Meer wird wieder aus der Draufsicht eingefügt (7 auf dem Ziffernblock), es bleibt vorerst nur eine einfache Plane (SPACE > *Add* > *Plane*), die auf das zehnfache vergrößert wurde (S > 10). Das Objekt wird noch nach oben bewegt, bis die Insel nur noch ein wenig aus dem Wasser schaut.

5.2.1.4 Die Glaskugel

Hierfür wird der 3D-Cursor leicht neben der Vase platziert, dies geschieht mit der linken Maustaste. SPACE > *Add* > *Icosphere* fügt eine Kugel aus Dreiecken zur Szene hinzu. Als Subdivision Level genügt zwei. Alle Vertices werden zunächst auf die Hälfte skaliert (S > .5) und danach mit SHIFT+D dupliziert, die Bewegung aber mit Rechtsklick abgebrochen. S > .99 skaliert die Kugel ein klein wenig, sodass sie nun als eine Innenwand für die äußere fungiert.

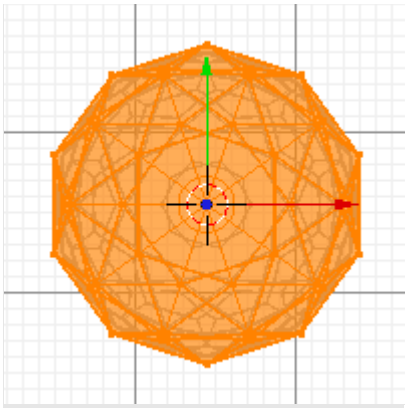

Abbildung 7: Die Kugel in der Wireframeansicht

Mit CTRL+SHIFT+N werden die Flächennormalen nach innen gerichtet, denn das sollen sie ja auch bei einer hohlen Glaskugel.

5.2.1.5 Die Grashalme: Partikelsystem

Für die Grashalme wird ein einfacher Würfel eingefügt (SPACE > *Add* > *Cube*), dessen Deckfläche zu einem Vertex in der Mitte vereinigt wird (W > *Merge* > *At Center*). Er wird auf die Hälfte seiner Größe (S > .5) und zusätzlich in Z-Richtung auf die Hälfte (S > Z > .5) skaliert.

Dieser Würfel fungiert als Emitter für die Partikel, die später unsere Grashalme bilden werden. Unter *Object* > *Physics* (2x F7) lassen sich diverse Simulationen für Objekte einstellen, auch Partikel fallen unter diese Kategorie.

Mit New wird nun ein Partikelsystem erzeugt, die entsprechenden Einstellungen müssen aber noch bearbeitet werden. Der Button Static erzeugt statische Partikel, perfekt geeignet für Haare, Grashalme oder ähnliches. 25 Partikel sind in diesem Fall absolut ausreichend, und mit dem Button Vect wird den einzelnen Partikeln eine Richtung gegeben, was für die spätere Beleuchtung sehr wichtig ist. Rand und Even sorgen zusätzlich für eine gleichmäßige Verteilung. Der Life Wert sollte 35 betragen.

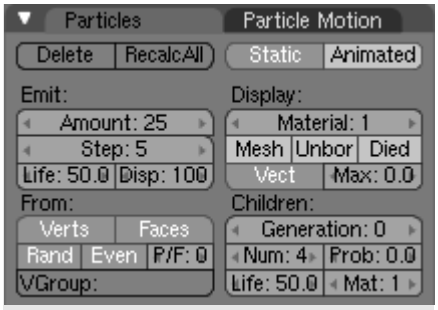

Abbildung 8: Partikeleinstellungen

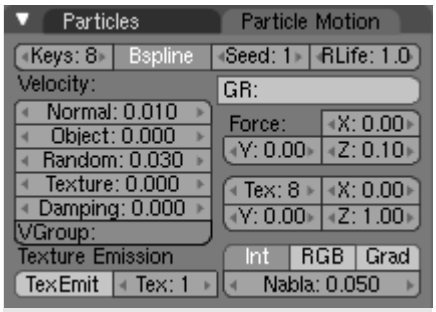

Abbildung 9: Bewegungseinstellungen

Im Tab *Particle Motion* kann man nun Eigenschaften wie Bewegungsrichtung und Geschwindigkeit einstellen. Unsere Partikel bekommen eine Anfangsgeschwindigkeit von 0.01 in Richtung der Flächennormalen (Normal) und 0.03 in zufällige Richtung (Random). Die Werte unter Force bestimmen, wie stark und in welche Richtung die Partikel abgelenkt werden, da Grashalme normalerweise nach oben aus der Erde wachsen, stellen wir Z auf 0.10.

Der Wert Keys bestimmt, wie viele Schlüsselpositionen für die Partikelpfade verwendet werden, 8 ist gut geeignet, und auch der Button B-Spline sollte aktiviert sein, er bewirkt eine Interpolation der Pfade mittels der B-Spline-Formel, was sich in einem rundlicheren und natürlicheren Verlauf bemerkbar macht. RLife lässt die Grashalme verschieden lang werden, 1 ist ein ausreichender Wert. Mit Seed wird die Zufallsvariable eingestellt, hier gibt es keinen Vorgabewert, jeder soll nach seinem eigenem Empfinden entscheiden, wie es ihm am besten gefällt.

5.2.1.6 Positionieren

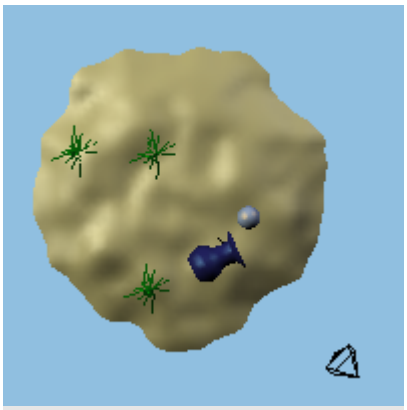

Abbildung 10: Die platzierten Objekte

Alle Modelle müssen in der Szene noch an günstigen Stellen platziert werden. Dies kann entweder über das Widget geschehen oder mit der Taste G.

Die Vase sollte in den Vordergrund gerückt werden und leicht mit der Öffnung in die Kamera schauen. Indem sie ein wenig nach unten verschoben wird, scheint es, als ob sie schon länger im Sand liegen würde und eingesunken wäre.

Die Kugel wird etwas neben der Vase platziert, sollte die Inseloberfläche aber nur berühren.

Der Grasbüschel kann irgendwo im Hintergrund platziert werden, mit ALT-D können abhängige Duplikate erstellt werden, damit die Szene nicht so leer wirkt.

Die Kamera wird an der Stelle 8, -8, 5 positioniert, mit den Rotationswerten 70, 0, 45.

5.2.2 Einfärben

Mit F5 gelangen wir zu den vielen Materialeinstellungen für unsere Objekte.

5.2.2.1 Die Vase

Unserer Vase verpassen wir als diffuse Farbe ein dunkles Blaues mit den RGB-Werten von 0.15, 0.15 und 0.4. Als Glanzlichtfarbe verwenden wir ein helles Cyan mit den Werten 0.6, 1 und 0.9.

Als Name vergeben wir „Dunkelblau“, um es später einfacher wieder zu finden.

5.2.2.2 Die Insel

Die Insel bekommt ein sandiges Gelb. Die Werte sind hier 1, 0.95 und 0.6. Der Name sollte „Sand“ sein.

5.2.2.3 Das Meer

Das Meer braucht ein helles Blau (0.6, 0.7, 0.9). Speziell hier ist jedoch noch zu beachten, dass der Emit Wert auf 0.5 und der Alpha Schieberegler auf 0.8 gestellt wird. Auch hier wählen wir einen aussagekräftigen Namen: „Wasser“

5.2.2.4 Die Glaskugel

Unser Glas bekommt ein sehr helles Blau (0.8, 0.9, 1) und als Glanzlichtfarbe ein kräftiges Gelb (1, 0.6, 0).

Die Glanzlichteinstellungen werden noch auf 1.1 für die Stärke und 90 für Hard (Hardness) gestellt.

Unter dem Tab *Mirror Transp* werden nun die Einstellungen für das Raytracing, also die Spiegelungen und Lichtbrechungen vorgenommen.

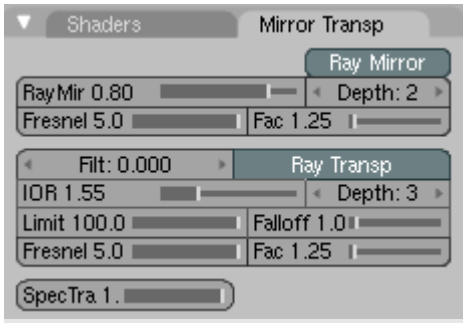

Abbildung 11: Klares Glas

Die beiden Buttons Ray Mirror und Ray Transp müssen aktiviert werden, um überhaupt einmal Raytracing einzuschalten. Die Fresnel-Werte werden beide Male auf 5 gestellt, sie sind für die Blickwinkelabhängigkeit verantwortlich. 5 entspricht hierbei der Realität am ehesten.

Die Stärke der Spiegelung wird mit dem Schieberegler RayMir eingestellt, 0.8 erzeugt für unser Bild eine passende Spiegelung.

Die Lichtbrechung wird durch einen wichtigen Schieberegler beeinflusst. IOR (Index of refraction) bestimmt den Brechungsgrad, dieser kann für verschiedenste Materialen aus Tabellen[23] entnommen werden, und entspricht der physikalischen Brechzahl *n*.

$n_1 \cdot \sin(\alpha) = n_2 \cdot \sin(\beta)$ ist die Formel, die die Brechzahl beschreibt. α entspricht hierbei dem Einfallswinkel des Lichtstrahls und β dem gebrochene Winkel im Medium, n_1 und n_2 sind die einzelnen Brechzahlen der jeweiligen Materialien. n_1 wird in Blender als 1 betrachtet und repräsentiert die ungefähre Brechzahl von Luft (1,000292), physikalisch ist die Brechzahl 1 für Vakuum definiert.

Unsere Glaskugel erhält einen Index von 1.55, was der Brechzahl von etwas stärker brechendem Glas entspricht, Werte für Glas liegen zwischen 1.45 und 1.93.

Wie auch schon die vorigen Materialien erhält es einen Namen: „Glas“

5.2.2.5 Die Grashalme

Die Grashalme bekommen ein sattes Grün (0.05, 0.5, 0.05) mit hellem Grün (0, 1, 0) als Glanzlicht. Unsere Gräser sind aber keine normalen Objekte, sondern Partikelstrands, die einige Sondereinstellungen benötigen, um akzeptabel auszusehen.

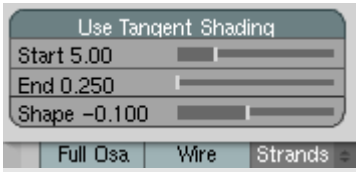

Abbildung 12: Strands

Unter *Links & Pipeline* findet sich ein Button mit der Aufschrift Strands, der es ermöglicht, die Dicke der Grashalme festzulegen. Sie sind am Anfang dicker und werden zum Ende hin dünner, mit den Schiebereglern Start und End kann dies sehr präzise gesteuert werden. Der Regler Shape bestimmt, auf welche Art der Strand seine Dicke ändert, positive Werte bewirken eine runde Form, negative Werte eine spitze Form.

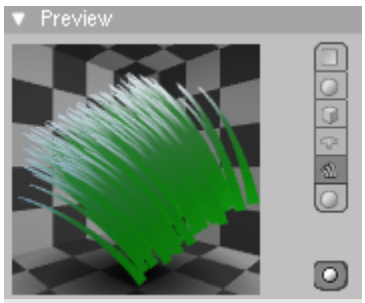

Abbildung 13: Vorschau

Unsere Grashalme erhalten deshalb die Einstellungen Start: 5, End: 0.250, und Shape: -0.100. Im Panel *Preview* kann auch auf Strands umgeschaltet werden, so erhält man gleich einen besseren Überblick, wie sich die Einstellungen auswirken. Auch hier wieder ist ein Name wichtig: „Gras“.

5.2.2.6 Der Himmel

Die Eigenschaften des Himmels können über die Materialeigenschaften der Kamera bearbeitet werden.

23 http://de.wikipedia.org/wiki/Brechzahl#Brechzahl_der_Luft_und_anderer_Stoffe

Die Horizontfarbe kann bei dem Standardblau belassen werden, die Zenithfarbe sollte aber auf ein helles Blau (0.4, 0.4, 0.8) gestellt werden.

5.2.3 Texturieren

Die Texturen können mit F6 erstellt und mit F5 den Materialen zugewiesen werden.

Generell kann man allen Materialien eine Normalmap zuweisen, so wirkt das Bild nicht so steril und etwas realistischer, ohne AO (Ambient Occlusion) zu verwenden, was unter Umständen sehr lange Renderzeiten mit sich zieht. So hat man einen klaren Geschwindigkeitsgewinn, der sich oft auch grafisch positiv bemerkbar macht.

5.2.3.1 Die Vase

Für die Vase genügt eine einfache Noise Textur. Mit dem Autobutton wird der Name gleich entsprechend gewählt, in diesem Fall einfach „Noise“.

Unter *Map To* wird der Nor-Button (Bumpmap) zweimal gedrückt, um ihn auf negativ zu setzen, und der Csp-Button (Color: Specularity) einmal. Der No RGB-Button wird aktiviert, da die Textur keine Farbinformationen enthält, und so mit dem Farbfeld darunter eine Farbe festgelegt werden kann (0, 1, 0.2). Weiters wird mit dem Nor-Schieberegler die Stärke des Bumpmapeffekts auf 5 gestellt.

5.2.3.2 Die Insel

Die noch immer vorhandene und dem Material zugewiesene Cloud Textur wird auf den Typ Wood geändert, und die folgende Optionen unter *Wood* gesetzt: BandNoise, SoftNoise und Tri. Turbulence sollte auf 3.45 gesetzt werden. Auch hier wieder kann der Name der Textur mit dem Autobutton gesetzt werden.

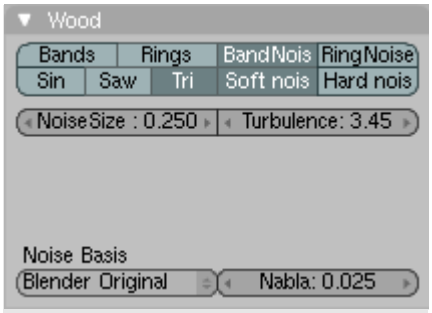

Abbildung 14: Wood-Textur

Sie wird nur als Bumpmap dienen, der Nor-Button muss aktiv sein und die Stärke auf 2 gestellt werden. Sie sorgt für eine leicht wellige Oberfläche.

Weiters verwendet die Insel die selbe Noise Textur wie die Vase, sie kann direkt in den Materialeinstellungen im Dropdownmenu ausgewählt werden.

Diesmal werden die Buttons Col und Nor aktiviert. Die Stärke von Nor wird auch dieses Mal wieder auf 5 gesetzt und auch der No RGB Button wird aktiviert. Die Farbe ist aber ein schmutziges Braun (0.5, 0.5, 0.1).

5.2.3.3 Das Meer

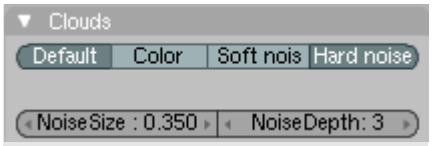

Abbildung 15: Cloud-Textur

Das Meer erhält eine Cloud-Textur mit Size 0.350, NoiseDepth 3 und Hard Noise aktiviert. Sie wird negativ auf den Alpha Kanal gemappt, No RGB sollte auch hier aktiviert sein. Der Name sollte diesmal „WasserAlpha“ oder ähnlich lauten.

Die zweite Textur kann grundsätzlich die selbe sein, für spätere Änderungen empfiehlt es sich aber, eine unabhängige Textur zu erzeugen. Dies kann aber jederzeit nachträglich mit einem Klick auf das Icon, welches die Anzahl der Benutzer der Textur zeigt, links neben dem Namen geschehen.

Diese Textur wird nun auf die Color-, negativen Normal- und Displacementkanäle (Disp) gemappt. Die Stärke beträgt 10 für den Normalkanal und 0.5 für den Displacementkanal. Auch hier gilt, wie bei den anderen Texturen: No RGB aktivieren und die Farbe festlegen (0.2, 0.4, 0.6).

5.2.3.4 Die Glaskugel

Die Glaskugel kommt ohne Textur aus, sie besteht aus schlichtem einfarbigem Glas.

5.2.3.5 Die Grashalme

Die Grashalme bedürfen auch in diesem Schritt wieder besonderer Aufmerksamkeit. Als erstes wird eine Blend Textur benötigt, welche mit einem Colorband versehen wird.

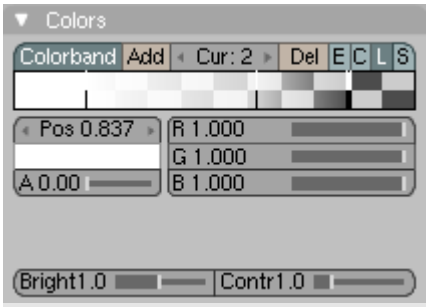

Abbildung 16: Colorband

Das Colorband enthält die Transparenzinformationen. Die Farben der einzelnen Werte sind hierbei egal. 3 Punkte eignen sich hier gut, zwei mit einem Alphawert von eins und einer mit einem Alphawert nahe 0. Rechts ist der Anfang der Grashalme, links befindet sich die Spitze. Der Verlauf sollte nun den eigenen Bedürfnissen angepasst werden.

Gemappt wird diesmal wieder auf den Alphakanal, und der Alphawert des gesamten Materials auf 0 gestellt. So sind die Strands vorne durchsichtiger als hinten, wenn zusätzlich noch im Tab *Map Input* der Button Strand aktiviert wird, er sorgt dafür, dass die Textur auf den Verlauf der Partikelpfade angewandt wird, und nicht auf das Emittermesh.

5.2.3.6 Der Himmel

Der Himmel bekommt eine Cloud-Textur, es kann auch die selbe wie die des Wassers sein. Die Buttons Blend und Hori im *Map To* Panel bestimmen, in welcher Weise die Himmelsfarbe von der Textur beeinflusst wird. No RGB aktivieren und die gewünschte Farbe (0.8, 0.8, 1) einstellen, schon sollte der Himmel in der Vorschau mit fülligen Wolken gezeigt werden.

Abbildung 17: Wolken

5.2.4 Lichtsetup

Drei Lampen werden die Szene in kurzer Zeit beleuchten, die erste befindet sich schon seit dem Beginn in der Szene. Sie wird nun etwas weiter außerhalb positioniert (4, -15, 6), erhält eine gelb-orange Lichtfarbe (1, 0.75, 0) und fungiert als Hauptlampe für die Szene.

Die zweite Lampe ist eine einfache Lamp, die sich genau in der Öffnung der Vase befindet. Sie hat eine Energy von 1.3, aber nur eine Reichweite (Dist) von 20, die Farbe ist auch wieder das selbe Gelb. Weiters soll sie keine Schatten werfen, der Button Ray Shadow schaltet Schatten ein und aus.

Für weiche und lange Schatten sorgt nun ein Spot, der die Szene relativ flach bestrahlt. Er befindet sich schräg hinter der Kamera (20, -10, 3) und zeigt auf die Vase und ist leicht nach unten geneigt (85, 0, 65). Um weich verlaufende Schatten zu erhalten, wird ein hoher SpotBi-Wert (1) benötigt, der Lichtkegel wird etwas spitzer gehalten, dies wird mittels SpotSi (20) in Grad gesteuert.

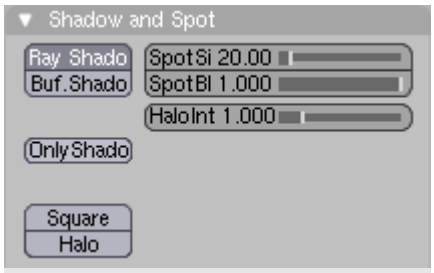

Abbildung 18: Spot

Schatten sollen per Raytracing berechnet werden, es muss der Button Ray Shadow anstatt Buf. Shadow aktiviert werden. Als Farbe wird hier ein kräftiges Orange (1, 0.4, 0) eingestellt.

5.2.5 Tweaking

Bei genauerem Betrachten der Szene fallen noch einige Ungereimtheiten und Fehler auf, diese müssen natürlich behoben werden.

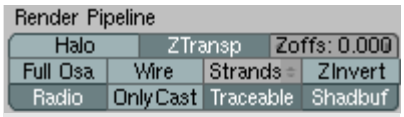

Abbildung 19: Z-Transparenz

Die Grashalme sind an den transparenten Stellen blau, wie der Himmel, und der Schatten der Glaskugel ist an allen Stellen gleich schwarz. Diese zwei Ungereimtheiten können aber schnell behoben werden, für die Grashalme wird in den Materialeinstellungen ZTransp im *Links & Pipelines* Panel aktiviert. Es sorgt dafür, dass die Transparenz tatsächlich mit den dahinter liegenden Objekten berechnet wird, und nicht einfach der Himmel als Referenzfarbe genommen wird. Diese Einstellung sollte bei allen Objekten gemacht werden, deren Transparenz nicht mit Raytracing berechnet wird, außer der Effekt ist gewünscht.

Abbildung 20: Transparente Schatten

Um transparente Schatten zu erhalten, muss in den Materialeinstellungen von den Objekten, auf die der Schatten geworfen wird, der Button TraShadow aktiviert werden. Es ist auch empfehlenswert, bei den transparenten Objekten selbst diese Einstellung zu aktivieren, denn oftmals werfen Objekte einen Schatten auch auf sich selbst. Eigentlich können alle Objekte in unserer Szene mit transparenten Schatten dargestellt werden, die längere Renderzeit macht sich kaum bemerkbar.

Ein kleines aber feines Detail kann mit Nebel simuliert werden. Zuerst sollte dazu in den Editing-Einstellungen der Kamera im *Camera*-Panel in der Kategorie *Show* der Button Mist eingeschaltet werden, so sieht man in der 3D-Vorschau, wo sich der Nebel befindet.

Abbildung 21: Nebel

In den Materialeinstellungen können dann die eigentlichen Nebeleinstellungen getroffen werden. Um Nebel zu erzeugen, muss der Button Mist aktiviert werden, mit den Werten darunter kann dann die Stärke und Reichweite eingestellt werden. Der Nebel sollte etwas hinter der Vase beginnen (Sta: 8) und über eine Distanz (Dist) von 3.5 an Stärke zunehmen. Nach oben soll er etwas leichter werden, mit einem Wert von 0.2 für Hi (High) ergibt das einen guten Effekt für unser Bild. Die Gesamtstärke des Nebeleffekts kann mit Misi (Mistintensity) eingestellt werden, es reicht 0.5. Der

Nebeleffekt wird dadurch erreicht, dass die Renderengine den Himmel durchscheinen lässt, je dichter der Nebel, umso mehr wird der Himmel sichtbar, er ist sozusagen eine Alphamaske, die über das Bild gelegt wird. Auf die selbe Weise können bei Bedarf auch Sterne hinzugefügt werden.

So ist aber noch immer sichtbar, wo die Meeresoberfläche aufhört. Wenn man diese um 45° rotiert, so dass die Kanten normal oder parallel zur Kamera stehen, verschwindet dieser kleine Fehler aber auch noch. Notfalls kann die Meeresfläche noch etwas weiter nach hinten verschoben werden, oder die Nebeldistanz verkürzt werden.

Ein letzter Schliff wird noch mit einem Rampshader auf der Insel hinzugefügt, mit Rampshadern kann man die Farbe in Abhängigkeit der auftreffenden Lichtenergie verändern und interessante Effekte kreieren.

Im Tab *Ramps* in den Materialeinstellungen kann dies schnell und einfach eingestellt werden. Dazu muss auf Colorband geklickt werden, um einen Farbverlauf hinzufügen. In diesem Farbverlauf wird der rechte Farbregler einfach auf Orange (1, 0.7, 0) gesetzt, und die Berechnung in Abhängigkeit der Normalvektoren (Normal) der auftreffenden Energie gesetzt.

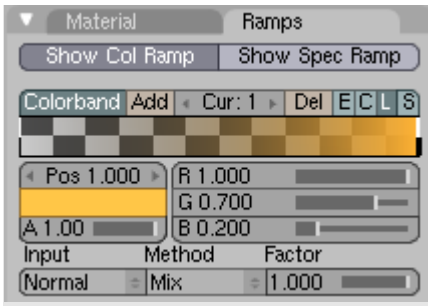

Abbildung 22: Rampshader

Die Szene ist noch an vielen Stellen verbesserungswürdig, dies alles hier aufzuzählen würde kein Ende nehmen. So kann zum Beispiel die kleine Lampe als Child der Vase gesetzt werden, um ein erneutes Ausrichten nach Veränderungen an der Vase überflüssig zu machen.

5.3 Rendern

Die Standardeinstellungen für den Rendervorgang, sind für diese Szene absolut ausreichend. Nach dem Drücken von F12 und einer kurzen Wartezeit, ist unser Bild fertig. Mit F3 kann das Renderergebnis für spätere Verwendung in einem Ordner gespeichert werden.

5.4 Resumé

Ich wollte die Arbeit mit Blender ein wenig darstellen. Es war natürlich nur ein kleines Beispiel, welches weit nicht alle Fähigkeiten von Blender aufzeigt, doch einige Grundzüge des Programms konnten hoffentlich dargelegt werden.

6 Nachwort

Im Nachhinein gibt es noch einiges von meiner Seite zu sagen. Während dem Schreiben dieser Arbeit steigerte sich mein Interesse für dieses Thema und damit verwandte Themengebiete. Ich las mich mehr und mehr in die Thematik der Versionierungssysteme CVS und SVN ein, mittlerweile sind diese FBA und einige andere Projekte auf meinem Computer unter Versionskontrolle, Änderungen können so einfach und ohne großen Aufwand rückverfolgt werden. Auch wenn mehrere Entwickler an einem gemeinsamen Projekt arbeiten, können diese gemeinsam an einer Datei arbeiten, ohne die Verbesserungen eines anderen Entwicklers zunichte zu machen, wie eben auch bei Blender.

Diese Systeme kommen bei vielen Entwicklungsprojekten zum Einsatz, wie zum Beispiel auf sourceforge.net, oder im Speziellen auf projects.blender.org. Bei Projekten und Teams ähnlich diesem Größenumfang ist es beinahe unmöglich, ohne diese Systeme effektiv und produktiv zu arbeiten.

Andererseits passiert es leicht, sich beim Recherchieren für eine Fachbereichsarbeit in den vorhandenen Informationen zu verlieren. Man sucht immer mehr und mehr, bis man schließlich nicht mehr beim eigentlichen Inhalt ist. So erging es auch mir, die FBA wurde immer wieder aufgeschoben und die Zeit wurde gern missachtet. So wurde es immer stressiger, je näher der endgültige Abgabetermin rückte, und es fiel mir immer wieder etwas ein, was noch geändert werden musste.

Aber jederzeit wieder würde ich mich zum Schreiben einer FBA entscheiden, so könnte ich mich wieder mit einem Thema meiner Wahl sehr intensiv beschäftigen und auch gleich für die Matura lernen.

7 Glossar

Ambient Occlusion: Ambient Occlusion (Umgebungsabdeckung) ist ein Beleuchtungsmodell, welches für realistischere Lichtverhältnisse bei relativ kurzer Renderzeit sorgt.
Area: Als Area wird ein Unterfenster in Blender bezeichnet
Armature: Eine Armature (Armatur) ist ein Skelett für ein animiertes Modell, häufig bei organischen Modellen verwendet. Beim Posing wird das assoziierte Mesh gemeinsam mit der Armature verformt, so erspart man sich umständliches Anpassen von Meshes.
Baking: Beim Baking werden die Ergebnisse von aufwendigen Simulationen berechnet und auf der Festplatte gespeichert, um später schnell aufgerufen zu werden. So müssen Fluidsimulationen, Softbodies und Partikel aber auch Ambient Occlusion Berechnungen nur einmal durchgeführt werden.
Blenderunit: Blenderunit ist der Name der Maßeinheit in Blender.
Blinn: Blinn ist ein Shader-Algorithmus, der für realistischere Glanzlicher bei beinahe gleich bleibenden Berechnungszeiten. Er wurde 1977 von James Blinn entwickelt.
Bone: Ein Bone (Knochen) ist ein Teil einer Armature.
Camera:Die Camera (Kamera) ist das Objekt, welches bestimmt, wie die Szene gerendert wird.
Compositing: Compositing (Mischen) ist eine Technik, die direkt während dem Rendervorgang eine digitale Fotonachbearbeitung erlaubt, so können zum Beispiel Schatten geschärft, oder Tiefenunschärfe zu Bildern hinzugefügt werden.
Diffuse: Diffus beschreibt einerseits einen Shadertyp und andererseits das Verhalten von Licht an Oberflächen.
Edge: Ein Edge (Kante) ist eine Verbindung zwischen zwei Vertices, und die Beschränkung von einem Face.
F-Gon: Ein F-Gon ist ein spezieller Typ von einem Face in Blender, er kann verwendet werden, wenn die interne Geometrie nicht wichtig ist, und es nur auf die äußere Form ankommt.
Face: Ein Face (Fläche) stellt die tatsächlich gerenderte Geometrie da, welche sich zwischen 3 oder mehr Vertices befinden kann. Der Name trifft noch keinerlei Bestimmung darüber, wie viele Eckpunkte vorhanden sind.
Fresnel: Mit den Fresnelschen Formeln lässt sich berechnen, bei welchem Blickwinkel die Oberfläche eines Gegenstands transparent beziehungsweise reflektierend ist. Mit dem Fresnel-Wert lässt sich diese Eigenschaft sehr gut beschreiben beziehungsweise einstellen. Entdeckt wurden die Formeln von Augustin Jean Fresnel.
GUI: Graphical User Interface (Grafische Benutzer-Schnittstelle) ist eine Bezeichnung für die Komponente einer Software, welche es dem Benutzer ermöglicht, mit Hilfe von Eingabegeräten wie Tastatur und Maus mit der Software in Interaktion zu treten.
Halo: Halos sind Effekte, die eigentlich nicht der Natur entsprechen, das menschliche Auge aber von Fotografien gewohnt ist. Sie helfen trotzdem oft, den Realitätsgrad eines Bildes zu erhöhen, können ihn aber auch komplett zerstören.
Hotkey: Hotkeys (Schnellzugriffstaste) ermöglichen das schnelle Ausführen von bestimmten Aktionen.
Lambert: Lambert ist ein Algorithmus für diffuse Lichtstreuung auf Flächen in Abhängigkeit des Winkels. Er wurde von Johann Heinrich Lambert formuliert.
Material: Mit Materialen lassen sich Farben, Texturen und das Lichtverhalten von Flächen steuern.
Mesh: Ein Mesh (Netz) beschreibt die eigentliche Geometrie eines Objektes.
Metaball: Ein Metaball ist die geometrische Beschreibung eines Algorithmus, welcher von James Blinn entwickelt wurde.
Modelling: Modelling (Modellieren) beschreibt den Vorgang 3D-Objekte und Szenen zu erstellen.

Modifier: Ein Modifier kann nachträglich einem Objekt zugewiesen werden und die Geometrie verändern, ohne das Originalmesh zu zerstören. Modifiers können beliebig ausgetauscht und hinzugefügt werden.
Object: Objects (Objekt) beinhalten Informationen, wie zum Beispiel welches Mesh verwendet wird, und können noch zusätzlich manipuliert werden.
Panel: Panels bezeichnen in Blender kleine Fenster, die eine Gruppe von Buttons und Einstellungen enthalten. Sie können beliebig verschoben werden, und mehrere können problemlos miteinander kombiniert werden. Sie erscheinen dann als Tabs, so wie es in modernen Browsern üblich ist.
Particle: Ein Particle (Partikel) ist ein Vertex, der durch Emittereinstellungen kontrolliert wird. Die Bewegung kann durch Manipulatoren noch nachträglich verändert werden. Partikel werden in Blender auch verwendet, um Haare darzustellen.
Phong: Phong ist die Bezeichnung für ein Beleuchtungsmodell für Glanzlichter. Er wurde nach dem gleichnamigen Erfinder Bui-Tuong Phong benannt.
Polygon: Ein Polygon ist eine Fläche in der 3D-Computergrafik. Im eigentlichen Sinn bedeutet sie nur Vieleck, wird meist jedoch als Synonym für Viereck verwendet.
Posing: Posing (Posieren) bezeichnet die Anordnung von Bones für einzelne Bewegungsabläufe und Animationsschritte.
Python: Python ist eine Skriptsprache, welche in Blender dazu verwendet werden kann, Add-Ins (Erweiterungen) zu programmieren, wie zum Beispiel Spiele oder Import/Exprotskripts
Quad: Ein Quad bezeichnet ein Polygon mit genau vier Ecken.
Radiosity: Radiosity ist ein globales Beleuchtungsverfahren, welches neben Raytracing am häufigsten eingesetzt wird.
Raytracing: Raytracing (Strahlenverfolgung) ist ein Algorithmus, der mit Hilfe von Strahlen die Sichtbarkeit und den Schattenwurf von Objekten berechnet. Aber auch Spiegelungen und Lichtbrechungen sind durch diesen Algorithmus möglich.
Rendering: Rendering (Übersetzen) bezeichnet den Arbeitsschritt, in dem die 3D-Daten in zweidimensionale Bilddaten übersetzt oder gezeichnet werden.
Scene: Die Scene (Szene) beinhaltet die Objekte, welche gerendert werden sollen.
Shader: Ein Shader (Töner) berechnet mit wählbaren Algorithmen die Farbverläufe auf Objekten.
Skinning: Skinning bezeichnet die Zuweisung einer Armature an ein Mesh oder Objekt.
Softbody: Softbodies (Weichkörper) sind eine Möglichkeit gummiartige Körper zu simulieren.
Space: Space (Raum) beschreibt unter normalen Umständen den Raum, in dem sich eine Szene befindet. In Blender beschreibt Space aber auch noch den Inhalt einer Area.
Specular: Specular (spekular) ist die Bezeichnung für Glanzlicher.
Texture: Eine Texture (Textur) kann noch zusätzliche Informationen für Materialen enthalten. Sie werden oft nicht nur für Farbinformationen verwendet, sondern für Geometrieinformationen, oder um bestimmte Eigenschaften wie den Anteil des Glanzlichtes an bestimmten Stellen zu steuern.
Toon: Toon (Zeichentrick) ist ein spezieller Stil, Bilder zu rendern. Bei ihm wird kein Wert auf Realität gelegt, sondern es wird versucht, das Bild künstlich aussehen zu lassen. Dies wird durch harte Kanten bei Schatten und Glanzlichtern erreicht, und durch zusätzliches Nachzeichnen von Kanten, was unter dem Namen Cel-Shading bekannt ist.
Triangle: Triangle (Dreieck) ist eine Bezeichnung für ein Polygon mit genau drei Ecken.
Vertex: Ein Vertex ist ein Kontrollpunkt im 3D-Koordinatensystem. Er ist die kleinste Einheit beim Arbeiten mit 3D-Modellen.
Wireframe: Wireframe (Drahtgitter) ist eine Darstellungsart, bei der nur die Kanten (Edges) von Objekten dargestellt werden, und nicht die gesamte Geometrie. Dieser Modus wird oft benötigt, um Elemente zu erreichen, die sonst verdeckt und nicht sichtbar wären.
World: Im Fall Blenders ist die World (Welt) das größte Objekt, das alle anderen Objekte enthält, sowie einige szenenspezifische Einstellungen.

8 Quellenverzeichnis

Alle Quellen wurden mit dem 25. Februar noch einmal auf ihre Aktualität kontrolliert:

http://blender.org/blenderorg/blender-foundation/history/

http://blender.org/development/

http://blender.org/development/architecture/notes-on-sdna/

http://blender.org/development/architecture/window-manager/

http://blender.org/development/current-projects/project-openings/

http://blender.org/forum/viewtopic.php?t=10540

http://blenderartists.org/forum/showthread.php?t=81725

http://blendernation.com/2006/06/06/blender-foundation-in-the-open-source-pavilion-at-siggraph/

http://blendernation.com/2007/01/03/blendernation-birthday-present-a-whole-school/

http://blendernation.com/2007/02/22/introducing-character-animation-with-blender-has-hit-the-shelves/

http://de.wikibooks.org/wiki/Blender_Dokumentation

http://de.wikipedia.org/wiki/3D-Computergrafik

http://de.wikipedia.org/wiki/3D-Grafik-Software

http://de.wikipedia.org/wiki/Brechzahl#Brechzahl_der_Luft_und_anderer_Stoffe

http://de.wikipedia.org/wiki/Grafikformat#Liste_von_Dateiformaten_f.C3.BCr_Rastergrafiken

http://features.cgsociety.org/story_custom.php?story_id=1647&page=

http://google-code-updates.blogspot.com/2007/02/speaking-of-summer.html

http://mediawiki.blender.org

http://orange.blender.org, http://elephantsdream.org

http://projects.blender.org/pipermail/bf-committers/2007-February/017423.html

http://projects.blender.org/pipermail/bf-committers/2007-February/017532.html

http://renderman.pixar.com/

http://wiki.cgsociety.org/index.php/Blender

http://www.answers.com/computer+graphics&r=67

http://www.cad.com.au

http://www.gnu.org/copyleft/gpl.html

http://www.irtc.org/pipermail/irtc-l/2002-July/011490.html

http://www.mopi.nl/blogo/p1.html

http://www.siggraph.org/publications/newsletter/v32n3/contributions/machover2.html

http://www.techfak.uni-bielefeld.de/ags/cg/2006_WS_LV/CG1/01_Einfuehrung.pdf

Abbildungsverzeichnis

9 Arbeitsprotokoll

Mai 2006: Erster Kontakt mit dem Programm "Blender"

Juni 2006: Erste Gespräche mit Professor Schürz über eine FBA

Juli – August: Intensives Einarbeiten in Blender

September 2007: Nochmals Gespräch mit Professor Schürz über FBA

21.09.06: Abgabe Themenstellung

23.10.06: Besprechung mit Professor Schürz

22.01.06: Besprechung mit Professor Schürz

19.02.06: Anpassen aller Links auf blender.org nach Website Update und neuem Release

25.02.06: Überprüfen aller Links

28.02.06: Drucken der Arbeit

10 CD

Fachbereichsarbeit als PDF-Datei

Blender 2.43 für Windows, Linux, OSX

Aktueller Quelltext (25. Februar 2007)

Vorkompilierte Versionen von graphicall.org und blenderbuilds.com

Beispieldateien der FBA

Beispieldateien für Blender 2.43

Deutsches Wikibook zu Blender