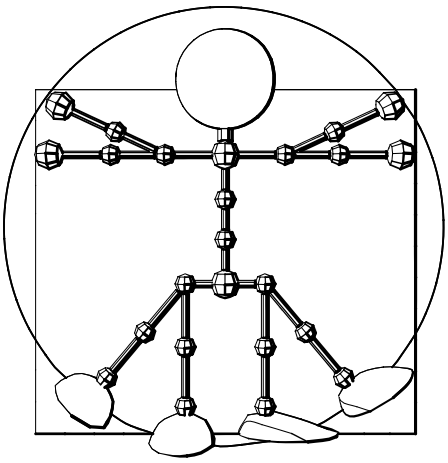
Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Institut für Simulation und Graphik

Visualisierungseffekte in liniengraphischen Animationen und Stillbildern

Diplomarbeit

Tobias Isenberg



Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Institut für Simulation und Graphik

Visualisierungseffekte in liniengraphischen Animationen und Stillbildern

Diplomarbeit

Verfasser: Tobias Isenberg
geboren am 14. Dezember 1974 in Gardelegen
Matrikel-Nr. 150 765

Prüfer: Prof. Dr. Thomas Strothotte
Prof. Dr. Michiel Smid

Betreuer: Dipl.-Inform. Maic Masuch

Beginn: 15. März 1999

Abgabe: 9. September 1999

Zusammenfassung

Illustrationen spielen eine große Rolle im täglichen Leben. Vor allem Liniengraphiken werden häufig als Illustrationen verwendet. Die computergestützte Erzeugung illustrativer Liniengraphiken war bisher jedoch nur ungenügend möglich. Die vorliegende Arbeit stellt Techniken zur computergestützten Generierung derartiger Graphiken aus dreidimensionalen Computermodellen vor.

Sogenannte Linienstileffekte werden mittels einer Linienstileffekt-Hierarchie und lokalem Keyframing kombiniert, um sowohl Einzelbilder als auch Liniengraphik-Animationen zu erstellen. Es werden exemplarisch vier konkrete Linienstileffekte entworfen, die besonders geeignet sind, illustrative Effekte in Liniengraphiken zu erzeugen. Die Implementierung des entwickelten Konzeptes sowie der vorgeschlagenen Linienstileffekte in einem Linienstileffekt-Animationssystem wird vorgestellt, und einige Beispiele von mit dem System erstellten Graphiken und Animationen werden präsentiert.

Abstract

Illustrations are very important in everyday life. In particular, line drawings are frequently used to illustrate things. Until now, the computer-assisted generation of line drawings for illustrative purposes did not produce satisfying results. This thesis presents techniques for computer-assisted generation of such graphics that make use of three-dimensional computer models.

So-called line style effects are combined using a line style effect hierarchy and local keyframing to produce line drawing stills as well as line drawing animations. Four line style effects are exemplarily designed which are particularly well qualified to produce illustrative effects in line drawings. The implementation of the developed concept as well as the suggested line style effects in a line style effect animation system is introduced, and some examples of stills and animations that were generated with the system are presented.

Danksagung

An dieser Stelle möchte ich allen herzlich danken, die mich beim Schreiben dieser Diplomarbeit und allgemein während des Studiums unterstützt haben. An erster Stelle steht dabei meine Familie. Vor allem danke ich meiner Mutter und meinem Vater, die mir durch ihre bei weitem nicht nur finanzielle Unterstützung das Studium erst ermöglicht haben. Dank gilt auch meiner Schwester für ihre Kompromißbereitschaft in bezug auf das „Studentenauto“.

Besonderen Dank schulde ich meinen Lehrern während meiner Schulzeit in Gardelegen und Kleinmachnow, von denen ich viel gelernt habe. Insbesondere möchte ich hiermit Ingrid Sanftenberg, Sabine Schwieger und Jürgen Adolf danken.

Bedanken möchte ich mich bei meinen Betreuern Thomas Strothotte und Maic Masuch, die mir durch Anregungen und Denkanstöße immer wieder weitergeholfen haben. Ebenfalls Dank an Stefan Schlechtweg und Bert Freudenberg für viele inspirierende Gespräche und Hinweise. Besonderer Dank gilt Stefan, der in einer entscheidenden Situation für mich da war und mir auch sonst u. a. bei LaTeX-Problemen behilflich war.

Vielen Dank an André, Anja, Claudia, Elke, Henry, Nicole und Volker für die Assimilierung, das dienstägliche Schwimmen und diverse Grillabende am Bungalow, die für die notwendige Abwechslung während der Diplomarbeit gesorgt haben.

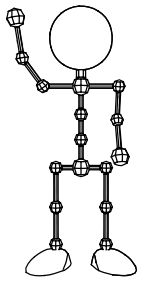
Dank gebührt Henry Handrich, Axel Hoppe, Bernd Knischewski, Felix Ritter und Nicole Schmidt für das Korrekturlesen der Arbeit und der D-216-Crew sowie Petra Janka, Petra Specht und Sylvia Zabel für die interessanten Diskussionen während der letzten Monate über Orthographie und Grammatik der deutschen Sprache alter Rechtschreibung und über andere schwierige Probleme.

Selbständigkeitserklärung

Hiermit versichere ich, Tobias Isenberg (Matrikel-Nr. 150 765), die vorliegende Arbeit allein und nur unter Verwendung der angegebenen Quellen angefertigt zu haben.

Magdeburg, 9. September 1999 Tobias Isenberg

Inhaltsverzeichnis



Kapitel 1

Einleitung

Vom Computer erzeugte photorealistische Bilder und besonders photorealistische Computeranimationen haben Einzug in weite Teile unseres täglichen Lebens gefunden. Man findet sie in Werbungen im Fernsehen oder in Zeitschriften. Viele der neu in die Kinos kommenden Filme sind ohne Computeranimationen überhaupt nicht mehr vorstellbar, wie der gerade in Deutschland angelaufene Film „Star Wars Episode I: Die dunkle Bedrohung“ hervorragend demonstriert.

Ein anderes großes Gebiet innerhalb der Computergraphik neben dem Photorealismus ist die Erzeugung nicht-photorealistischer Bilder. Diesem Forschungsgebiet wurde anfangs jedoch relativ wenig Aufmerksamkeit geschenkt. In den letzten Jahren ist das Interesse am Nicht-Photorealismus allerdings stark gestiegen, und die Forschung auf dem Gebiet wurde intensiviert.

Betrachtet man Illustrationen in Büchern, Zeitschriften oder wissenschaftlichen Arbeiten allgemein, so fällt auf, daß diese meist nicht-photorealistischer Natur sind. Oft finden Liniengraphiken Verwendung, da diese nicht nur technische Vorteile gegenüber Pixelbildern beim Druck haben, sondern auch aufgrund ihrer guten Abstraktionsmöglichkeiten besser als Illustrationen geeignet sind als andere nicht-photorealistische Graphiken.

Ein Ziel der Stilrichtung des Nicht-Photorealismus ist daher die computerunterstützte Erzeugung von wissenschaftlichen Illustrationen. Dabei bauen viele Verfahren auf die Verwendung dreidimensionaler Geometriemodelle auf. Das Hervorheben von Teilen des Modells ist bei existierenden Systemen jedoch stark von der Objektstruktur der Modelle abhängig. Mit derartigen Methoden erzeugte Graphiken erfordern somit aber entweder einen sehr hohen Aufwand bei der Modellerstellung bzw. Modellanpassung oder sind oft für illustrative Zwecke nur ungenügend geeignet.

Darüber hinaus ist der Informationsgehalt von Einzelbildern beschränkt. In Animationen wird die zusätzliche Dimension der Zeit genutzt, um weitere Informationen zu vermitteln. In der wissenschaftlichen Visualisierung werden bereits Animationen eingesetzt, um Sachverhalte zu illustrieren. Solche Animationen beschränken sich bisher jedoch meist auf Visualisierungen von wissenschaftlichen Daten, wie beispielsweise in animierten Diagrammen oder in Animationen bei bildgebenden Verfahren in der Medizin.

1.1 Ziele der Arbeit

Die vorliegende Arbeit verfolgt das Ziel, Methoden zur computergestützten Erzeugung illustrativer Liniengraphiken aus dreidimensionalen Geometriemodellen zu entwickeln, die mit zusätzlichen Informationen angereichert wurden. Es werden Techniken vorgestellt, mit denen es in Liniengraphiken nicht nur möglich ist, Linienstilattribute auch unabhängig von der im Geometriemodell vorhandenen Objektstruktur zu beeinflussen, sondern auch diese Beeinflussungen geeignet zu animieren. Die Techniken sollen es damit erlauben, illustrative Liniengraphiken bzw. Liniengraphik-Animationen zu generieren.

Die Beeinflussung der Linienstilattribute erfolgt auf der Grundlage von Linienstileffekten, die eine zeitlich beschränkte und voneinander unabhängige Manipulation der Linienstilattribute erlauben. Die vorgestellten Methoden sollen sowohl die Erzeugung von Einzelbildern als auch die Generierung von Animationen zulassen. Eine Implementierung soll es dem Benutzer ermöglichen, interaktiv Linienstileffekte auf eine bereits modellierte Szene anzuwenden, um damit eine der Visualisierungsaufgabe entsprechende Animation oder ein entsprechendes Einzelbild zu erzeugen.

1.2 Vorbilder und traditionelle Illustrationen

Vorbilder für die in dieser Arbeit zu entwickelnden Effekte bilden traditionelle Illustrationen, wie sie in wissenschaftlichen Veröffentlichungen, aber auch in anderen Medien gefunden werden können. Einige Beispiele werden im folgenden vorgestellt.

Abbildung 1.1 zeigt eine wissenschaftliche Illustration aus der Medizin. Hier wurde ein Teil des Daumens einer menschlichen Hand mit teilweise sichtbaren inneren Strukturen hervorgehoben, um das Interesse des Betrachters auf diese Stelle zu lenken. Die Hervorhebung erfolgte durch Verwendung besonders starker und dunkler Linien. Der Übergang geschieht jedoch nicht abrupt, sondern gleichförmig von weniger wichtigen hin zu wichtigeren Gebieten des Bildes.

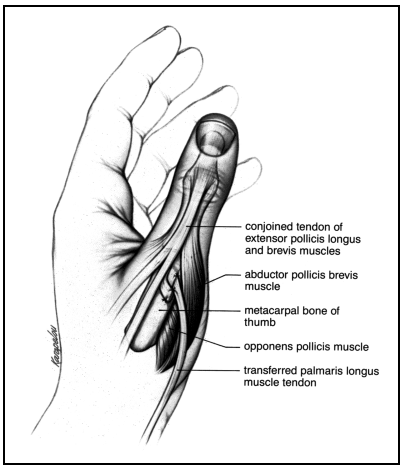


Abbildung 1.1: Beispiel aus der Medizin: Hervorheben eines Gebietes, um das Interesse darauf zu lenken (aus [Hod89, Seite 56]).

Ein Beispiel einer Comic-Zeichnung, die in einem eher skizzenhaften Stil gezeichnet ist, zeigt Abbildung 1.2. Für den Kopf der Figur wurden jedoch deutlich stärkere Linien verwendet als für den Rest des Körpers. Dies könnte darauf hindeuten, daß der Kopf im Verlaufe des Entwurfs der Figur schon besonders gut ausgearbeitet ist oder auch besondere Bedeutung für den darzustellenden Charakter hat.

Abbildung 1.2: Beispiel aus einer Comic-Zeichnung: Betonung des Kopfes (aus [TJ95, Seite 442]).

In Abbildung 1.3 ist eine technische Illustration dargestellt. Da es besonders auf die Seitenflächen der Körper ankommt, wurden sie durch einen starken Linienstil vom Rest der Darstellung abgehoben. Der Betrachter kann somit nicht nur die Gesamtheit der Körper erfassen, sondern auch die für sie charakteristischen Seitenflächen.

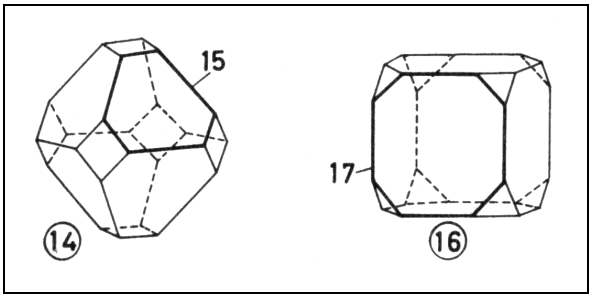


Abbildung 1.3: Hervorhebung einer Seitenfläche der Körper (aus [Dud77, Seite 609]).

Ein letztes Beispiel zeigt die Verwendung von Beleuchtungseffekten zur Verdeutlichung des räumlichen Charakters von Objekten (vgl. Abbildung 1.4). Die verschiedenen Helligkeiten wurden durch unterschiedliche Linienstärken dargestellt. Im konkreten Beispiel befindet sich die Lichtquelle links oberhalb der Objekte, und helle Stellen werden durch dünne Linien und dunkle Stellen durch starke Linien dargestellt.

Vorbilder für animierte Liniengraphik-Illustrationen konnten trotz intensiver Recherche nicht gefunden werden, was möglicherweise am bereits für einzelne Bilder hohen künstlerischen und technischen Aufwand liegt. Bei bestehenden liniengraphischen Animationen wurde bisher nicht versucht, Linienstilattribute gezielt zum Zwecke der Illustration zu verändern. Die vorliegende Arbeit betritt daher in diesem Punkt Neuland. Die Vorbilder für die zu erstellenden Graphiken reduzieren sich somit auf Einzelbilder.

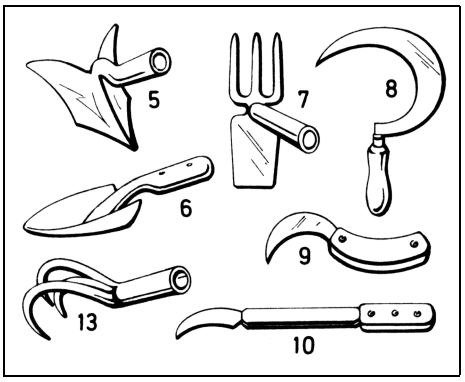


Abbildung 1.4: Erhöhung der räumlichen Wirkung durch die Verwendung von Beleuchtungseffekten (aus [Dud77, Seite 105]).

1.3 Gliederung der Arbeit

Nach dieser Einleitung gibt Kapitel 2 zunächst einen Überblick über die für die vorliegende Arbeit wichtigen Grundlagen. Dabei wird zuerst auf Illustrationen eingegangen, danach folgt eine Diskussion verschiedener Verfahren zur Generierung nicht-photorealistischer Graphiken. Es schließt sich ein kurzer Überblick über die Geschichte der Animation und über Verfahren zur Spezifizierung von photorealistischen Computeranimationen an. Das Kapitel schließt mit einer Diskussion von Techniken zur Animation nicht-photorealistischer Graphiken.

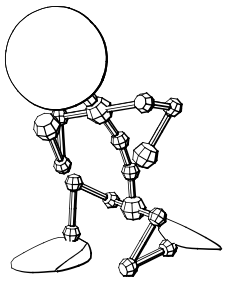
In Kapitel 3 wird ein Konzept zur Kombination von Linienstileffekten entwickelt. Das Kapitel geht zunächst auf die Notwendigkeit der Manipulation von Linienstilattributen unabhängig von Objektstrukturen ein und gibt eine Definition für den Begriff des Linienstileffektes. Anschließend werden verschiedene Methoden zur Kombination von Linienstileffekten vorgestellt, ihre Vor- und Nachteile abgewogen und daraus ein geeigneter Ansatz abgeleitet.

Für den so entwickelten Ansatz werden in Kapitel 4 neue Effekte entworfen, die eine geeignete Manipulation der Linienstilattribute erlauben. Die Beschreibung der einzelnen Linienstileffekte geht neben deren Definition auch auf mögliche Modifikationen und Erweiterungen ein.

Kapitel 5 stellt die im Rahmen dieser Arbeit angefertigte Implementierung eines Linienstileffekt-Animationssystems vor. Dabei wird neben Implementierungsdetails zum System und zu den einzelnen Effekten auch auf Interaktionsaspekte bei der Benutzung des Systems eingegangen.

Eine Auswahl von Beispielgraphiken und Beispielanimationen, die mit dem entwickelten System erzeugt wurden, ist in Kapitel 6 enthalten. Neben Erläuterungen zu den Beispielen gibt das Kapitel auch Hinweise zum Einsatz der Effekte, die sich aus den während der Arbeit mit dem Programm gesammelten Erfahrungen ergeben.

Kapitel 7 gibt abschließend eine Zusammenfassung der erreichten Ergebnisse und stellt einige Aspekte vor, die in weiterführenden Arbeiten behandelt werden könnten.



Kapitel 2

Grundlagen

Dieses Kapitel geht zunächst auf die computerunterstützte Generierung von Illustrationen und die Eignung nicht-photorealistischer Techniken auf diesem Gebiet ein. Anschließend werden einige Verfahren zur Erstellung nicht-photorealistischer Graphiken vorgestellt. Danach wird ein kurzer Überblick über die Geschichte und die Methoden der photorealistischen Animation gegeben, wobei ein besonderes Augenmerk auf die Technik des sogenannten Keyframings gelegt wird. Darauf aufbauend werden abschließend Verfahren zur Animation von nicht-photorealistischen Graphiken erläutert.

2.1 Illustrationen

Die Bedeutung des Wortes *Illustration* geht auf das lateinische Wort *illustrare* zurück, das soviel wie „erhellen“ bzw. „erleuchten“ bedeutet. Die moderne Bedeutung von *illustrieren* besteht in

1. „[durch Bilder] erläutern“,
2. „[ein Buch] mit Bildern schmücken; bebildern“ [Dud96].

Illustrationen spielen eine große Rolle im täglichen Leben. Sie unterstützen Lernende beim Verstehen des Lehrstoffes. Vor allem in naturwissenschaftlichen Veröffentlichungen werden Illustrationen eingesetzt, um Sachverhalte näher zu erläutern und Zusammenhänge zu erklären. Aufgrund von Einschränkungen im Druckprozeß[1] sind Illustrationen oft schwarzweiße Liniengraphiken. Neben diesen technischen Vorteilen gegenüber Fotos besitzen Liniengraphiken oft noch weitere entscheidende Vorzüge: In Liniengraphiken können Details weggelassen werden, Strukturen durch Änderung des Zeichenstiles akzentuiert und somit der Detaillierungsgrad dem Visualisierungsziel angepaßt werden. Trotz der Fortschritte der Technologie werden liniengraphische Illustrationen jedoch auch heute noch meistens von Hand realisiert, da die Erstellung guter Illustrationen hohe künstlerische Fähigkeiten und Erfahrung voraussetzt. Hodges schreibt in dem immer noch gültigen Buch [Hod89] beispielsweise (vgl. auch [Lei95]):

[1] Die Rasterung von Graustufenbildern ist im Gegensatz zur Rasterung von schwarzen Linien sehr viel gröber. Daher ist die Reproduktionsqualität von Liniengraphiken deutlich besser als die von Pixelbildern.

„... der Illustrator soll eine genaue Arbeit produzieren, die auch ästhetisch ansprechend ist, was die ausgewogene und künstlerische Darstellung des Objektes anbelangt.“[2]

2.1.1 Computergenerierte wissenschaftliche Illustrationen

Trotz des eben angesprochenen hohen künstlerischen Anspruchs bei traditionellen Illustrationen wird innerhalb der Computergraphik versucht, qualitativ hochwertige Illustrationen mit dem Computer zu generieren. Dabei werden nicht nur Verfahren untersucht, die es erlauben, statische Illustrationen zu erstellen. Es wird u. a. auch versucht, illustrative Animationen oder interaktive Illustrationen zu generieren. Die so erstellten Illustrationen verschiedener Art sollen es erlauben, den Prozeß der Erkenntnisgewinnung ähnlich wie oder sogar besser als traditionelle Illustrationen zu unterstützen.

Pionierarbeit auf dem Gebiet der interaktiven, computergestützten Illustration wurde von Höhne et al. geleistet, die das System VoxelMan entwickelt haben [HPP+96]. Aufbauend auf einem Voxelmodell des menschlichen Körpers mit dahinterliegendem semantischen Netz ermöglicht das System u. a. die interaktive Generierung beliebiger Schnitte durch das Modell, die Generierung virtueller Röntgenaufnahmen und die Simulation von Endoskopien. Es wird vor allem für die Ausbildung im Bereich der menschlichen Anatomie eingesetzt und verwendet Daten des Visible Human Projects.

Ein anderes Beispiel für ein interaktives Illustrationssystem ist der von Preim et al. vorgestellte Zoom Illustrator (vgl. [Str98, Kapitel 13]). Er zeigt exemplarisch anhand von medizinischen Illustrationen, wie in einem interaktiven System textuelle und bildliche Informationen kombiniert werden können. Durch die Möglichkeit zur Interaktion sowohl mit dem 3D-Modell als auch mit den textuellen Erläuterungen wird es dem Benutzer einfacher gemacht, komplexe Sachverhalte zu verstehen. Insbesondere werden Fisheye-Zoom-Techniken angewendet, um dem Nutzer die für ihn wichtige Information hervorzuheben und ihm mehr Details zu den von ihm gewünschten Themen zu präsentieren.

Ein weiteres Verfahren zur interaktiven Präsentation von Illustrationen wird in [SS99] von Schlechtweg und Strothotte vorgestellt. Das System TextIllustrator fokussiert mehr auf lange Texte, durch die ein Nutzer navigiert. Beim Verstehen des Textes wird der Nutzer durch eine Graphik unterstützt, die aus einem 3D-Modell gewonnen wird und mit der er interagieren kann. Ausgehend von der Interaktion mit dem Text werden angepaßte Ansichten aus dem 3D-Modell generiert – eine Technik, die *Illustrative Browsing* genannt wird. Umgekehrt kann aber auch der Text durch die Interaktion mit dem 3D-Modell bestimmt werden.

In allen genannten Arbeiten wird die vorteilhafte Wirkung der Interaktion auf das Verstehen komplexer Sachverhalte erläutert. Im Gegensatz zu solchen neuen Ansätzen mit interaktiven Methoden wird aber auch weiterhin versucht, Bilder ähnlich traditionellen

[2] Originalwortlaut: “... the illustrator should produce accurate work that is also pleasing to the eye in terms of balance and artistic handling of the subject.”

Illustrationen zu erzeugen. Dies erscheint sinnvoll, da herkömmliche Medien wie Bücher oder Zeitschriften, bei denen keine Interaktion möglich ist, immer noch eine große Rolle bei der Wissensvermittlung spielen.

2.1.2 Illustrationen mit nicht-photorealistischen Graphiken

Im Laufe der Zeit wurden die computerbasierten photorealistischen Verfahren immer besser und die so erzeugten Bilder immer realistischer. Derartige Bilder haben jedoch einen entscheidenden Nachteil. Sie enthalten zwar viele Informationen, sind aber nicht unbedingt *informativ* (vgl. [DC90b]). Photorealistische Bilder enthalten meist zu viel ablenkendes und unwichtiges Detail, um als Illustration genutzt werden zu können. Dooley und Cohen fordern ein System, das die Struktur in einem Modell visualisiert, ohne den Betrachter mit relativ insignifikanten Details zu belasten.

Die Autoren schlagen in [DC90a] vor, als Ausgangspunkt für die automatische Generierung von Illustrationen ein 3D-Modell zu verwenden. Weiterhin stellen sie folgende Ansprüche an ein Illustrationssystem:

- Den Objekten eines Modells müssen zusätzliche semantische Attribute zugeordnet werden,
- es müssen vielfältigere Zeichenprimitive vorhanden sein und
- Illustrationsregeln müssen die geometrischen Informationen und zusätzlichen Attribute in Zeichenprimitive umsetzen.

Durch die Verwendung von verschieden starken und auf unterschiedliche Art gestrichelten Linien ist es den Autoren möglich, Bilder zu erzeugen, die die Struktur von geometrischen Gebilden illustrieren (vgl. Abbildung 2.1). Die Linienattribute (Strichelung) werden durch den Verdeckungsgrad und durch benutzerdefinierte Attribute wie Wichtigkeit mittels der Illustrationsregeln bestimmt.

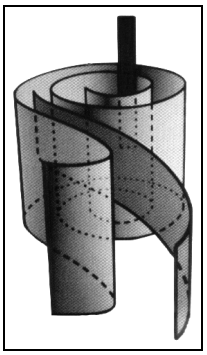


Abbildung 2.1: Beispiel für ein computergeneriertes Bild zur Visualisierung von Verdeckungen mit Hilfe von Transparenz und gestrichelten Linien zur Illustration der Struktur (aus [DC90b]).

Strothotte et al. führen in [SPRF94] aus, daß skizzenhafte Bilder einen Einfluß auf Betrachter ausüben können, ähnlich wie handgezeichnete Bilder. Derartige Bilder können über das in den Bildern tatsächlich vorhandene Wissen weitere Informationen vermitteln. So kann beispielsweise die Aufmerksamkeit des Betrachters auf bestimmte

Teile eines Bildes gelenkt werden, in denen mehr Details zu sehen sind als in anderen Teilen. Dies kann mit nicht-photorealistischen Graphiken realisiert werden. Die Nutzung von nicht-photorealistischen Techniken hat gegenüber photorealistischen Verfahren u. a. den Vorteil, daß durch die Wahl eines anderen nicht-photorealistischen Darstellungsstils einem Bild ein anderer Charakter gegeben werden kann, ohne daß das für die Erstellung verwendete Geometriemodell dafür geändert werden müßte.

Insbesondere für die Visualisierung von Unvollständigkeit und Unvollkommenheit von Darstellungen eignen sich nicht-photorealistische Darstellungsstile, konkret skizzenhafte Bilder. In [SSRL96] wird von J. Schumann et al. anhand einer Studie nachgewiesen, daß Betrachter für die Darstellung eines ersten Entwurfs eines Gebäudes eine skizzenhafte Darstellung gegenüber einer herkömmlichen CAD-Zeichnung bzw. einem mit einem photorealistischen Verfahren erzeugten Bild vorziehen würden.

Auch in [Ger98] wird von Gershon vorgeschlagen, nicht-photorealistische Gestaltungselemente für die Visualisierung von Unvollkommenheit oder Unsicherheit zu verwenden. So regt der Autor beispielsweise an, gestrichelte oder unscharfe Linien zu verwenden. In [SMI99] wird das System AncientVis vorgestellt, das ungenaue Linienstile verwendet, um Unsicherheiten in Rekonstruktionen zerstörter Bauwerke zu visualisieren.

Zusammenfassend kann gesagt werden, daß in vielen Situationen nicht-photorealistische Darstellungen den photorealistischen Bildern vorzuziehen sind. Dies sind u. a. Situationen, in denen es auf präzise Skizzen, erklärende Darstellungen und das Verstehen von Sachverhalten ankommt. Photorealismus ist auch insbesondere bei Generalisierungen schlecht geeignet, da photorealistische Abbildungen immer von konkreter Natur sind (vgl. [LS95]).

2.2 Techniken zur Generierung nicht-photorealistischer Graphiken

Im folgenden wird ein Überblick über verschiedene Techniken gegeben, mit denen nicht-photorealistische Computergraphiken erzeugt werden können. Der Fokus liegt dabei insbesondere auf Techniken, die *Liniengraphiken* erzeugen – im Gegensatz zu malereiähnlichen Bildern oder flächenhaften Graphiken (ähnlich Radierungen), die als Illustrationen i. d. R. ungeeignet sind. Unter Liniengraphiken sollen im folgenden die Graphiken verstanden werden, die mindestens eines der folgenden Stilelemente als wesentlichen Bestandteil enthalten:

- Konturlinien, die Objekte voneinander und vom Hintergrund abgrenzen und die Grobform der Objekte (möglicherweise inklusive einer Binnenstruktur) darstellen oder
- Schraffur- bzw. Texturlinien, die die Helligkeits-, Beleuchtungs- und Oberflächeninformationen darstellen.

Eine Übersicht über die verschiedenen Verfahren zur Erzeugung nicht-photorealistischer Graphiken wird von LANSDOWN und SCHOFIELD in [LS95] gegeben. Es ist zu bemerken, daß sich fast alle der vorgestellten Verfahren an vorhandenen künstlerischen Gestaltungsmethoden orientieren. Die zwei Hauptrichtungen sind die Nachahmung graphischer Gestaltungstechniken und die Imitation von Malereistilen. Die meisten der in [LS95] vorgestellten Verfahren werden von den Autoren sogenannten *Bildraum-Techniken* (*Image Space Effects*) zugeordnet. Derartige Techniken benutzen beliebige Eingabebilder und verfremden sie, um bestimmte Effekte zu erreichen. Eine andere Gruppe von Techniken werden *Perspektivraum-Techniken* (*Perspective Space Effects*) genannt, die mehr als ein einfaches Bild als Ausgangspunkt benötigen. So benutzen einige der aufgezählten Ansätze neben dem photorealistischen Ausgabebild eines herkömmlichen Renderers noch zusätzliche Tiefeninformationen oder Oberflächennormalen von Objekten der darzustellenden Szene.

Im Unterschied zur oben angeführten Unterteilung soll im folgenden zwischen Pixelverfahren und Vektorverfahren unterschieden werden. Zunächst werden einige Beispiele der ersten Gruppe besprochen. Anschließend wird näher auf vektororientierte (bzw. analytische) Techniken eingegangen und auch das in der vorliegenden Arbeit verwendete Linienmodell vorgestellt.

2.2.1 Pixelorientierte Techniken

Unter pixelorientierten Techniken werden die Verfahren verstanden, bei denen ausschließlich mit Pixelbildern gearbeitet wird. Grundlage dafür bilden die Arbeiten von SAITO und TAKAHASHI (vgl. [ST90]), die Methoden der Bildverarbeitung verwenden, um Pixelgraphiken mit zusätzlichen Informationen anzureichern. Diese zusätzlichen Informationen werden während des Renderingprozesses bzw. anschließend gewonnen und in sogenannten *G-Buffern* (geometrischen Zwischenspeichern) abgelegt. Dies sind Bilder, die die gleiche Größe des Zielbildes haben und in denen für jedes Pixel eine Eigenschaft an dieser Stelle abgespeichert ist. Beispielsweise können in einem G-Buffer Tiefenwerte abgespeichert werden (ein sogenannter *z-Buffer*). Mit Hilfe von derartigen zusätzlichen Informationen fügen die Autoren u. a. Konturkanten von Objekten, schraffurähnliche Strukturen und in Farbwerten kodierte Tiefeninformationen in die von einem Scanlinienrenderer erzeugten Bilder ein, um sie aussagekräftiger zu machen.

Das von SCHOFIELD entwickelte PIRANESI-System[3] benutzt vor allem den von einem photorealistischen Renderer generierten z-Buffer, um zusätzliche Informationen über den dreidimensionalen Aufbau der Szene zu erhalten. Davon abhängig werden Texturen auf das photorealistisch gerenderte Bild angewendet, wobei die Interaktion ähnlich einem normalen Bildbearbeitungsprogramm abläuft. Die Informationen aus dem z-Buffer werden u. a. dafür verwendet, um eine korrekte perspektivische Anordnung der Texturen zu erhalten. Auch werden Diskontinuitäten im z-Buffer für die Berechnung von Konturkanten benutzt (vgl. [Sch94] und [LS95]).

[3] Weitere Informationen unter `http://www.informatix.co.uk/piranesi.htm`

Ein weiteres Beispiel für pixelorientierte Techniken sind die von STRASSMAN in [Str86] vorgestellten *Hairy Brushes*. Sie simulieren das Verhalten von Farbpinseln und die von diesen auf das Medium übertragene Farbe, um so die japanische *Sumi-e*-Kunst nachzuahmen. Dazu wird ein Pinsel als aus einer Anzahl von Borsten bestehend definiert, die entlang einer Geraden angeordnet sind. Dieses Feld von Borsten wird beim „Malen“ entlang eines Pfades bewegt, wobei die Gerade mit den Borsten immer senkrecht zum Pfad ausgerichtet ist. Während der Bewegung des Pinsels über das Papier werden die Eigenschaften der Borsten verändert, u. a. ihre Position und die Menge an noch zur Verfügung stehender Farbe. Der Bewegungspfad selbst und die sich entlang des Pfades ändernden Parameter werden durch Splines[4] beschrieben. Nach einem Pinselstrich muß der Pinsel erneut „in die Farbe getaucht“ werden, um den Ausgangszustand wiederherzustellen. Abbildung 2.2 zeigt einige der erzielten Ergebnisse.

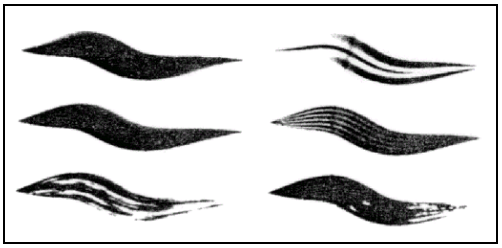


Abbildung 2.2: Pinselstriche mit *Hairy Brushes*-Technik (aus [Str86]).

2.2.2 Vektororientierte Techniken

Das Ausgabebild einer analytischen Technik ist im Gegensatz zu den pixelorientierten Verfahren kein Pixelbild, sondern eine auflösungsunabhängige Vektorgraphik. Im Verlaufe des Verfahrens können allerdings auch Pixelbilder zum Einsatz kommen, beispielsweise als Ausgangs- bzw. Referenzbild.

HSU, LEE und WISEMAN präsentieren in [HLW93] eine weitere sich an der *Pinselstrichmetapher* orientierende Methode zum Erzeugen nicht-photorealistischer Graphiken. Im Gegensatz zum oben vorgestellten Ansatz von STRASSMAN versuchen die Autoren aber nicht, die Funktion real existierender Malwerkzeuge zu simulieren. Statt dessen definieren sie einen *Stroke* als ein beliebiges Eingabebild mit einer Referenzlinie und einer Referenzbreite, das entlang eines Pfades (*Skeleton*, kubischer Spline bzw. polygonale Linie) deformiert wird. Des weiteren ist es möglich, verschiedene Teile des Strokes unterschiedlich zu deformieren (*Anchoring*). Die möglichen Deformationen beinhalten Verbiegen, Verdrehen sowie nicht-uniformes Stauchen und Dehnen der Ausgangsbilder. Auch können *Skeletal Strokes* rekursiv kombiniert werden. Einige Ergebnisse dieser Technik, die auch in ein kommerzielles Produkt[5] integriert wurde, sind in Abbildung 2.3 zu sehen.

Ein anderes Verfahren zur computerunterstützten Erzeugung von Liniengraphiken wird von WINKENBACH und SALESIN in [WS94] vorgestellt. Sie benutzen sogenannte *Prio-*

[4] Splines sind stückweise polynomiale Kurven, üblicherweise vom Grad 3.

[5] Expression von MetaCreations, weitere Informationen unter `http://www.metacreations.com/products/expression/`

Abbildung 2.3: Zwei Beispiele für die *Skeletal Strokes*-Technik (aus [HL94]).

ritized Stroke Textures, die sowohl die Struktur- als auch die Helligkeitsinformation einer liniengraphischen Textur darstellen. Eine Stroke Texture besteht aus einer Anzahl abgespeicherter bzw. prozedural erzeugter *Strokes*. Die Strokes sind in diesem Zusammenhang Geradensegmente, die durch eine Welligkeitsfunktion (*Waviness Function*) modifiziert wurden (vgl. [FS94]). Soll auf eine bestimmte Fläche eine Stroke Texture angewendet werden, dann werden so lange Strokes aus der Textur gezeichnet, bis der angestrebte Helligkeitswert erreicht ist. Eine Priorisierung der Texturelemente kann diesen Prozeß unterstützen, damit eine gleichmäßige Textur erreicht wird.

Der Vorteil dieses Verfahrens besteht in der Erzeugung auflösungsabhängiger Graphiken. Dies steht zunächst im Widerspruch zum Prinzip der Vektorgraphiken, da diese – technisch gesehen – auflösungsunabhängig sind. Betrachtet man jedoch die gleiche Vektorgraphik in zwei unterschiedlichen Vergrößerungsstufen, so fällt ein unterschiedlicher subjektiver Helligkeitseindruck auf. Diesem Phänomen wirkt das Prinzip der Stroke Textures entgegen. Da die Textur solange vervollständigt wird, bis die gewünschte Helligkeit erreicht ist, kann für jede gewünschte Auflösung eine entsprechende Graphik erzeugt werden. Eine Graphik mit geringerer Auflösung enthält entsprechend weniger Detailinformationen als eine vergleichbare Graphik in einer höheren Auflösung, um die beschriebene, gleiche subjektive Helligkeit zu erreichen (vgl. Abbildung 2.4).

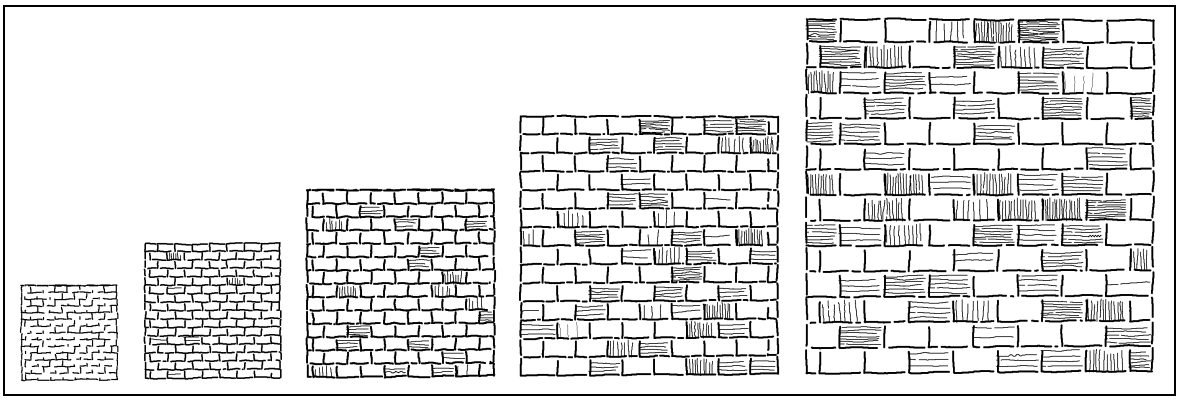


Abbildung 2.4: Die gleiche Stroke Texture in verschiedenen Auflösungen (aus [WS94]).

Das Prinzip der Stroke Textures wurde zum einen in einem 2D-System implementiert (vgl. [SABS94]). Es kann aber auch auf die Ausgabebilder eines photorealistischen 3D-

Renderingsystems angewendet werden, wobei die Blickrichtung und die Beleuchtung beachtet werden (siehe Abbildung 2.5). Es ist damit kein reines 3D-System, sondern eher ein $2\frac{1}{2}$D-System.

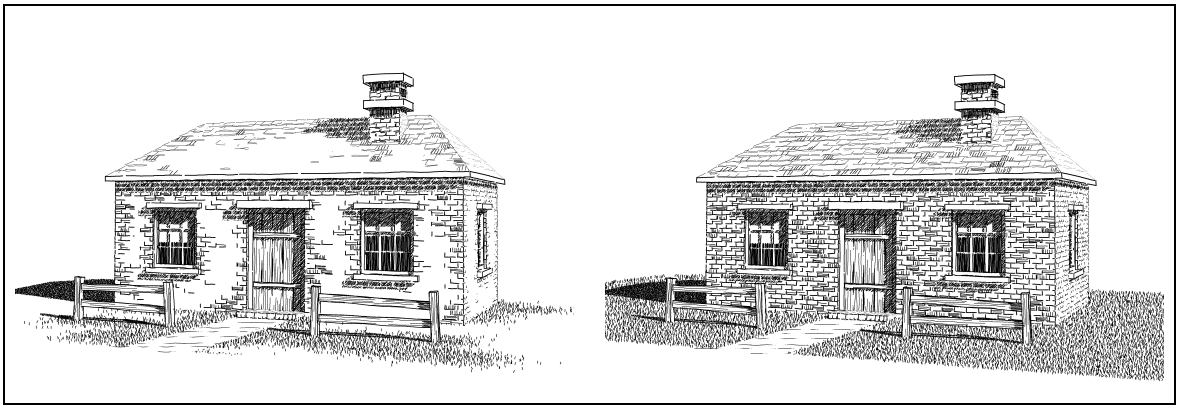
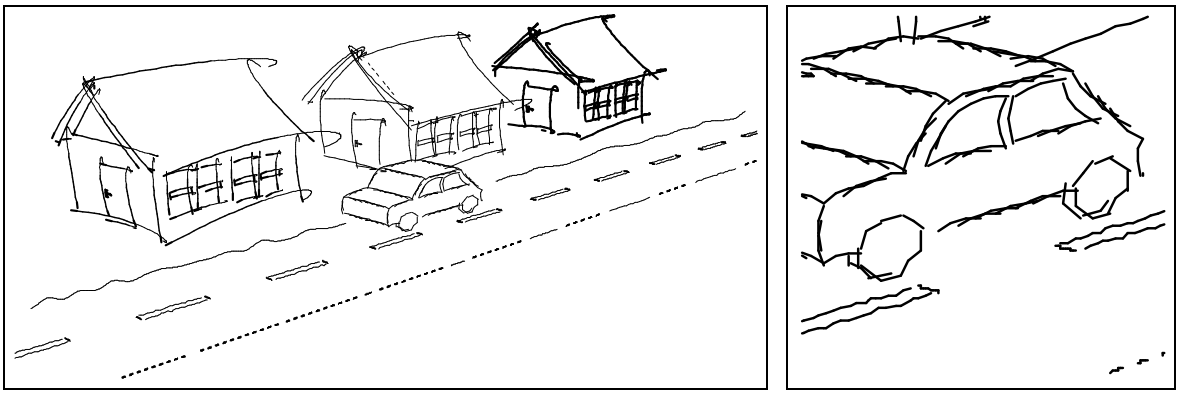


Abbildung 2.5: Ein Beispiel für die Verwendung von Stroke Textures, links mit und rechts ohne Einfluß auf den Detaillierungsgrad (aus [WS94]).

Die Grundlage für die vorliegende Arbeit bilden vor allem die Arbeiten von J. Schumann und L. Schumann. J. Schumann beschreibt in ihrer Diplomarbeit ein Modell, das die Erzeugung ungenauer Linien erlaubt (vgl. [Sch92]). Ziel ihrer Arbeit war es, computergenerierte Liniengraphiken „lebendiger“ zu machen, ähnlich handgezeichneten Illustrationen. Auch J. Schumann orientiert sich dazu an herkömmlichen Zeichenwerkzeugen und führt zu diesem Zweck den Begriff *Linienstil* ein. Ein Linienstil beschreibt, wie eine *Modellinie* in eine *Ziellinie* transformiert wird. Die Modellinie ist charakterisiert durch ihren Kurventyp, die Kontrollpunkte der Kurve, Farbsättigung, Linienstärke und Angaben über Strichelung. Weiterhin kann der Verlauf von Attributen angegeben werden (ansteigend, abfallend, gleichbleibend, etc.). Die Ziellinien, die den zu zeichnenden Pfad beschreiben, sind auf Geradensegmente beschränkt, die Ausgabe eines 2D-Graphiksystems oder eines analytischen Renderers sein können. Das beschriebene Linienmodell wurde im System SketchRenderer implementiert (vgl. [SPRF94]).

An einem Beispiel (vgl. Abbildung 2.6) läßt sich erkennen, daß die Beschränkung auf Geradensegmente zu Problemen führt, da so aus mehreren Segmenten bestehende Linienzüge nicht als eine Eingabelinie betrachtet werden können (siehe z. B. die Räder in Abbildung 2.6(b)). Auch ist die Einschränkung auf einen linearen Verlauf der Attribute ein wesentlicher Nachteil, da so reale Zeichenwerkzeuge nur ungenügend nachgeahmt werden können. Diese Probleme greift L. Schumann in seiner Diplomarbeit auf und schlägt ein erweitertes Linienmodell vor (vgl. [Sch97b]), das auch in der vorliegenden Arbeit Verwendung findet.

L. Schumann definiert als *Stil* die Abweichung von einem zu zeichnenden *Pfad* sowohl in geometrischer Hinsicht als auch in Hinblick auf bestimmte Attribute (Linienstärke und Liniensättigung). Die geometrische Störung ist durch die Differenz zwischen einer Störkurve und einer Referenzkurve, der *Stillinie* gegeben, wobei sich die Arbeit aus Gründen der Intuitivität auf Geradensegmente als Stillinien beschränkt. Die Stör-



(a) Gesamtansicht (b) Detailansicht

Abbildung 2.6: Vom SketchRenderer erzeugte Graphik (mit freundlicher Genehmigung von Andreas Raab).

kurven selbst sind durch stückweise polynomiale Kurven beschrieben. Ähnlich ist auch das Verhalten der Attributwerte durch polynomiale Kurven beschreibbar. Des weiteren wird auch der Pfad als Kurve aufgefaßt. Die Überlagerung von Pfad und Stil erfolgt durch eine angepaßte Differenzvektormethode, bei der der Differenzvektor zwischen Stillinie und geometrischer Störkurve parametrisch und tangential an den zu zeichnenden Pfad angetragen wird. Das Verfahren ist dabei nicht auf nur eine einzige Störkurve beschränkt, es ermöglicht auch die Definitionen von mehreren Kurven als Stil.

Ein wesentlicher Vorteil des Modells besteht in der Attributierbarkeit sowohl des Stiles als auch des Pfades. So ist es möglich, das Verhalten von Attributen der Ausgabelinie unabhängig vom benutzten Stil zu beeinflussen. Das Linienmodell kann nicht nur, wie in Abbildung 2.7 gezeigt, auf 2D-Graphiken angewendet werden, sondern auch auf analytisch gerenderte 3D-Szenen. Es ist dabei aufgrund der Attributierbarkeit des Pfades in der Lage, aus dem Modell gewonnene bzw. zusätzlich zum Modell vorhandene Daten in der Graphik in Attributverläufe umzusetzen.

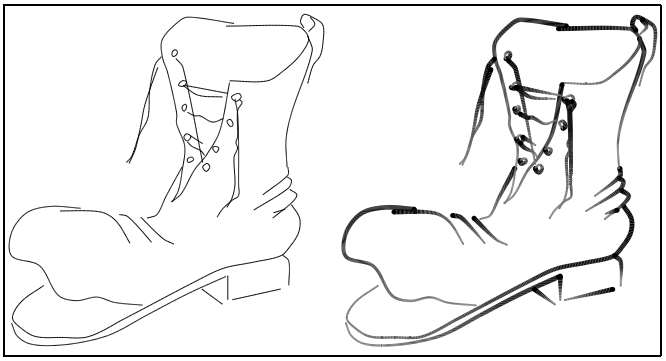


Abbildung 2.7: Beispiel für Graphik mit Linienmodell nach L. Schumann. Links sind die benutzten Pfade abgebildet, während rechts die fertige Graphik zu sehen ist (aus [Sch97b]).

Die Steuerung der Attribute durch den Pfad ist für die Darstellung von aus 3D-Modellen gewonnenen Liniengraphiken auch zwingend erforderlich. Würden die Attribute durch den Stil kontrolliert werden, so ist bei nichtkonstanten Attributen die Zuordnung der

Richtung des Verlaufs zu Kanten mehr oder weniger zufällig. So kann eine entlang eines Stiles abfallende Linienstärke z. B. an einer Kante $\overline{AB}$ von A nach B erfolgen, aber auch von B nach A.

Diesen Umstand berücksichtigt SCHÖNWÄLDER in seiner Diplomarbeit [Sch97a]. Er setzt das von L. SCHUMANN entworfene Linienmodell ein, um aus 3D-Modellen Graphiken mit charakteristischen Linienzügen zu erzeugen. Der Autor verwendet Informationen aus dem 3D-Modell, um die Konturkanten auszuwählen, zu Kurvenzügen zusammenzufassen und zu attributieren. So kann Beleuchtung mittels der Linienstärke visualisiert, Objekte können betont, Tiefeninformationen durch Linienattribute dargestellt und Verdeckungen durch Verkürzung von Linien hervorgehoben werden. Ein Beispiel für eine nach dieser Methode aus einem 3D-Modell erzeugte Graphik ist in Abbildung 2.8 zu sehen. Die Beeinflussung der Linienattribute aufgrund von Beleuchtung und Geometrieinformationen ist jedoch nur eingeschränkt möglich. So verwendet der Autor nur unendlich entfernte Lichtquellen und die Darstellung von räumlicher Information ist nur in eine Richtung, nämlich in Kamerarichtung möglich.

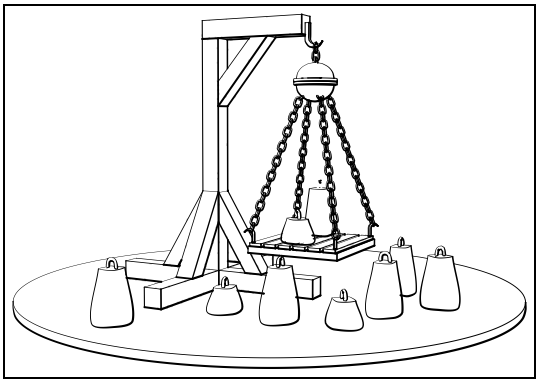


Abbildung 2.8: Beispiel für eine aus einem 3D-Modell erzeugte Graphik mit charakteristischen Linienzügen (aus [Sch97a]).

Nachdem Techniken des Nicht-Photorealismus vorgestellt und diskutiert worden sind, wird im folgenden zunächst kurz auf Computeranimationen eingegangen. Dabei wird insbesondere auf die Technik des Keyframings eingegangen, da diese für die Animation von nicht-photorealistischen Graphiken von besonderer Bedeutung ist.

2.3 Animation

Die wörtliche Bedeutung des Wortes *Animation* ist *Leben zu geben* [Web96]. Unter Animation wird üblicherweise sowohl die zeitliche Veränderung von Position, Form, Farbe, Transparenz, Struktur und Oberflächeneigenschaft von Objekten als auch die Veränderung von Lichtverhältnissen, der Kameraposition, -orientierung und -fokus sowie sogar die Änderung der Renderingtechnik verstanden [FvDFH90].

Mit der Entwicklung der elektronischen Rechentechnik und von graphischen Benutzungsschnittstellen wurden Techniken der traditionellen Animation auch auf den Computer übertragen. Im folgenden soll unter *Computeranimation* die kontinuierliche Berechnung von Zuständen in einem dynamischen System und die Darstellung sich in

Raum und Zeit verändernder Strukturen durch eine Reihe von Bildern (nach [GP99]) verstanden werden. Insbesondere beinhaltet diese Definition nicht nur die Darstellung von sich räumlich verändernden Objekten, sondern auch die sich mit der Zeit ändernde Gestalt und Darstellung der Objekte. Der Begriff *Animation* wird in dieser Arbeit synonym für *Computeranimation* verwendet.

2.3.1 Geschichte der Animation

Geiger und Paelke geben in [GP99] eine Zusammenfassung über die Geschichte der Animation. Sie begann um 1824 mit der Beobachtung von Roget, daß das menschliche Auge zu träge ist, einen Reiz sofort wahrzunehmen, und daß der Reiz auch über sein eigentliches Auftreten hinaus wahrgenommen wird. Wenige Jahre später wurden einfache Apparate entwickelt, die einfachste Animationen abspielen konnten. Um 1889 führte Edison den ersten Film mit einem auch von ihm entwickelten *Kinetoskop* vor. Die erste filmische Animation wurde 1899 von Melbourne-Cooper erstellt, 1911 folgte der Film „Gertie the Dinosaur“ von McCay, und 1937 kam der erste abendfüllende Zeichentrickfilm in die Kinos: „Snow White and the Seven Dwarfs“ von Disney.

Im Jahre 1949 wurde von Adams die erste Computeranimation erstellt. Es handelte sich um die Animation eines hüpfenden Balls auf dem am Massachusetts Institute of Technology (MIT) entwickelten Whirlwind-Computer. Mit der Einführung der graphischen Benutzungsschnittstellen wurde auch die Entwicklung der Computeranimation weiter vorangetrieben, so z. B. durch das Sketchpad von Sutherland, dem ersten graphischen System überhaupt (vgl. [Sut63]). Im Jahre 1974 gewann der computergenerierte Film „Hunger“ einen Preis bei den Filmfestspielen in Cannes. In den 70er Jahren wurde die dreidimensionale Computeranimation entwickelt, die 1982 in „Tron“ das erste Mal im Film eingesetzt wurde. Schließlich kam 1995 mit „Toy Story“ der erste abendfüllende, komplett computeranimierte Spielfilm in die Kinos.

2.3.2 Methoden der Computeranimation

Im Laufe der Zeit wurden verschiedene Methoden zur Modellierung von Computeranimationen entwickelt. Foley et al. nennen in [FvDFH90] die folgenden, die aber nicht unbedingt klar gegeneinander abgrenzbar sind:

Physikalisch basierte Animation: Werden die physikalischen Gesetze des Verhaltens bestimmter Systeme modelliert und auf die Animation von Objekten angewendet, spricht man von physikalisch basierter Animation. Insbesondere die Gesetze der Mechanik und Gravitation werden oft angewendet, um physikalische Systeme zu animieren.

Kinematik und Dynamik: Ein Spezialfall physikalisch basierter Animation ist die Verwendung von Kinematik und Dynamik, bei der die Position und Geschwindigkeit von Objekten bzw. deren Punkte unter Verwendung der physikalischen Gesetze modelliert werden.

Actors: Als *Actor* wird ein Computerprogramm bezeichnet, das die Eigenschaften eines Objektes im Verlaufe einer Animation bestimmt. Kommunizieren mehrere dieser Programme miteinander, können dadurch komplexe Animationen modelliert werden.

Motion Tracking: Benutzt man für die Animation von Menschen eine abstrakte Beschreibung ihrer Bewegungen, so wirken diese oft nicht sehr realistisch. Um derartige Animationen realistischer zu gestalten, verfolgt man die Bewegungen an realen Menschen und benutzt die so gewonnenen Daten für die Animation. Gleiches gilt auch für die Animation von Tieren.

Constraint-Beschreibung: Objekte in der realen Welt unterliegen oft vielfältigen Bedingungen. Diese Bedingungen können zur Beschreibung einer Animation benutzt werden. Das erste Beispiel für ein derartiges System war das bereits erwähnte SKETCHPAD (vgl. [Sut63]).

Prozedurale Beschreibung: Das Verhalten von Objekten kann durch andere Objekte der Szene beeinflußt werden. Werden solche Abhängigkeiten modelliert und zur Beschreibung einer Animation herangezogen, spricht man von einer prozeduralen Beschreibung. Unter anderem kommen hier Partikelsysteme zum Einsatz. Spezialfälle von prozeduraler Beschreibung sind die Verwendung von Actors und die physikalisch basierte Animation.

Volle, explizite Kontrolle: Hierbei werden alle veränderbaren Parameter der Animation explizit festgelegt. Die bei weitem populärste der hierbei angewendeten Techniken ist das *Keyframing*.

Von diesen Methoden ist besonders das im letzten Punkt genannte Keyframing für die vorliegende Arbeit von Bedeutung, da sie eine einfache Methode zur Spezifikation von Parameterveränderungen darstellt. Diese Technik steht daher im Mittelpunkt der Betrachtungen des nächsten Abschnitts.

2.3.3 Keyframing

Unter *Keyframes* wird die Definition von extremen bzw. charakteristischen Positionen der zu animierenden Objekte verstanden [FvDFH90]. Aus diesen Keyframes können dann durch Interpolation, auch *Inbetweening* genannt, die zwischen den Keyframes liegenden Einzelbilder generiert werden.

Die Technik des Keyframings wurde ursprünglich in den 20er Jahren von DISNEY entwickelt, um die Erstellung von Zeichentrickfilmen effizienter zu gestalten. Die Animation des ungefähr 90-minütigen Filmes „Snow White and the Seven Dwarfs“ erforderte beispielsweise mehr als 2 Millionen einzelne Zeichnungen. Da aber zum einen die talentierten Chefzeichner sehr teuer waren und zum anderen die schiere Menge der Zeichnungen ein großes Hindernis darstellte, mußte eine neue Technik für die Filmerstellung gefunden werden. Daher wurde der Produktionsprozeß hierarchisch organisiert. So

zeichneten die Chefzeichner nur die Keyframes und legten damit das generelle Aussehen der Animation fest. Die weniger erfahrenen oder talentierten *Inbetweeners* hatten diese anschließend zu einer Sequenz zu vervollständigen, die abschließend nur noch koloriert werden mußte [Wat93, WW92].

Diese Technik wurde in den 70er Jahren auch auf die dreidimensionale (photorealistische) Computeranimation übertragen. Auch in der Computeranimation definiert ein Benutzer über einen Keyframe den Zustand der zu animierenden Objekte zu einem bestimmten Zeitpunkt. Die Aufgabe des Inbetweening-Prozesses wird jedoch vom Computer übernommen, so daß der Mensch quasi die Rolle des Chefzeichners einnimmt.

Ein wesentlicher Unterschied zum Keyframing in Zeichentrickfilmen besteht darin, daß bei der Computeranimation ein Keyframe nicht mehr nur *einen* Gesamtzustand beschreibt. Vielmehr werden Keyframes unabhängig voneinander für die verschiedenen Parameter (Ort, Größe, Ausrichtung, Aussehen, etc.) definiert. Einige Autoren benutzen daher auch den Begriff des *Key Parameters* [WW92]. Jedem der animierbaren Parameter ist in modernen Animationssystemen ein eigener Zeitstrahl zugeordnet, auf dem unabhängig von den anderen Zeitstrahlen Keyframes angeordnet werden können. Diese Vorgehensweise ermöglicht die effizientere Spezifikation einer Animation, da so nur Parameter eingestellt werden müssen, die sich auch wirklich während der Animation ändern. Dies wurde u. a. notwendig, da in Computeranimationen wesentlich mehr Freiheitsgrade existieren als in der traditionellen Trickfilmtechnik.

Abbildung 2.9 zeigt das Fenster zum Einstellen der Keyframes in einem der gängigen Animationswerkzeuge, dem 3D Studio MAX R2.5. In der Abbildung sind verschiedene Zeitstrahlen zu erkennen, auf denen für die verschiedenen Parameter der Objekte (Position, Orientierung und Skalierung) unabhängig voneinander Keyframes definiert werden können.

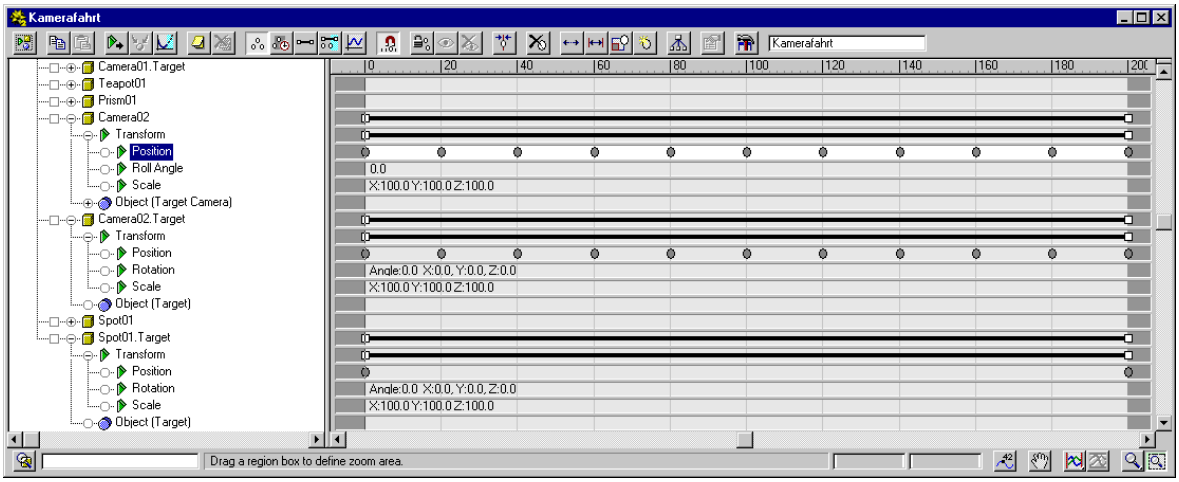


Abbildung 2.9: Trackview-Fenster von 3D Studio MAX R2.5. Für jeden animierbaren Parameter erscheint ein Keyframe-Track.

Für den Prozeß der Interpolation der Parameter zwischen den Keyframes werden verschiedene Methoden angewendet. Das einfachste Verfahren ist die lineare Interpolation

der Werte. Sie liefert jedoch meist nur ungenügende Ergebnisse, vor allem wenn es sich um die Beschreibung einer Bewegung handelt, da es an den Keyframes zu abrupten Änderungen kommt. Derartige plötzliche Änderungen sind meist unerwünscht, da sie nicht das Verhalten von Objekten in der realen Welt darstellen.

Daher werden zum Berechnen der Zwischenbilder meist Spline-Interpolationen verwendet, da sie allmähliche Übergänge auch an den Keyframes erlauben. Die Keyframes enthalten dazu die Stütz- bzw. Kontrollpunkte und die Art der zu verwendenden Spline-Interpolation. Dieses Verfahren wird nicht nur auf die Position eines Objektes, sondern auch auf andere Parameter angewendet. Für eine tiefergehende Diskussion von Keyframe-Interpolation wird auf [WW92] verwiesen.

2.4 Animation von nicht-photorealistischen Graphiken

Um Animationen mit nicht-photorealistischen Methoden zu erstellen, wird meist die eben beschriebene Technik des Keyframings benutzt. Dazu werden analog zu photorealistischen Computeranimationen die Parameter der nicht-photorealistischen Graphiken durch Keyframes festgelegt und durch eine geeignete Methode die zwischen den Keyframes liegenden Werte interpoliert.

Die im folgenden vorgestellten Systeme werden dabei in 2D- und 3D-Systeme unterschieden. In 2D-Systemen ist das unterliegende animierte Modell ein 2D-Modell, während bei 3D-Systemen dreidimensionale Modelle benutzt werden. Unter anderem wird im folgenden das dali-System vorgestellt, das neben dem bereits erwähnten Linienmodell nach L. Schumann einen Ausgangspunkt für die vorliegende Arbeit bildet.

2.4.1 2D-Systeme

Einige der bereits in Abschnitt 2.2 besprochenen nicht-photorealistischen Techniken eignen sich auch für die Erstellung von Animationen. So definiert Strassman beispielsweise durch Keyframes die Form und die Parameter des Bewegungspfades zu bestimmten Zeitpunkten und benutzt dabei den gleichen Pinsel und den gleichen Ausgangszustand (vgl. [Str86]). Zur Interpolation zwischen den Keyframes werden Splines verwendet. Auch Hsu und Lee animieren ihre Skeletal Strokes (vgl. [HL94]) mittels Keyframing. Ähnlich wie Strassman beschreiben auch sie alle Parameter (z. B. Form und Position des Skeletons) durch Keyframes. In ihrem System ist es sogar möglich, Zwischenzustände zu manipulieren und so neue Keyframes zu definieren.

Das System TicTacToon orientiert sich stark am traditionellen Prozeß des Herstellens von Trickfilmen (vgl. [FBC+95]). Auch hier wird eine zweidimensionale Keyframetechnik genutzt, um die Animation zu kontrollieren. Der Künstler benutzt ein Digitalisiertablett, um seine Figuren zu zeichnen, die anschließend vom System vektorisiert werden. Analog zur Trickfilmtechnik ist es möglich, mehrere verschiedene Schichten, sogenannte *Layers* zu verwenden, um eine dritte Dimension zu simulieren. Daher ist

das System kein reines 2D-, sondern eher ein sogenanntes $2\frac{1}{2}$D-System. Schwierigkeiten treten insbesondere beim Prozeß des Inbetweenings auf, da das System selbst die korrespondierenden Punkte in den Figuren identifizieren muß, was nicht für alle Fälle möglich ist. Daher ist das System nur in der Lage, einfache Animationen zu erstellen. Ein weiteres, sich am traditionellen Prozeß orientierendes System ist TOONZ von Softimage[6].

2.4.2 3D-Systeme

Ein System, das Partikelsysteme zur Animation nicht-photorealistischer Graphiken verwendet, wird von MEIER präsentiert (vgl. [Mei96]). Sie versieht die Oberfläche der Objekte der Szene mit Partikeln, die die Position einzelner Pinselstriche repräsentieren. Für die Pinselstriche werden Farbbilder mit einem Alphakanal benutzt. Üblicherweise werden jedoch nur Graustufenbilder verwendet, um die Farbe separat kontrollieren zu können. Weitere Attribute, wie Orientierung, Farbe und Größe der Pinselstriche, können sowohl in den Partikeln gespeichert als auch Referenzbildern entnommen werden, die mit herkömmlichen Renderern erzeugt werden. Anschließend wird ein *Painter's*-Algorithmus benutzt, um die Pinselstriche auf das Medium aufzutragen. Weiter hinten liegende Pinselstriche werden von weiter vorn liegenden „übermalt". Durch das Aufteilen des Bildes in verschiedene Schichten, eine auch in der traditionellen Malerei angewendete Technik, können die Ergebnisse noch verbessert werden (vgl. Abbildung 2.10). Ein nicht gelöstes Problem besteht in den konstanten Pinselstrichgrößen, die sich bei Skalierungen des Bildes nicht ändern, was dem intuitiv erwarteten Effekt bei Vergrößerungen oder Verkleinerungen widerspricht.

Abbildung 2.10: Einzelbild aus einer Painterly-Animation (aus [Mei96]).

Ein weiteres System zur Animation eines 3D-Modells, das ebenfalls Partikelsysteme einsetzt, wird von CURTIS in [Cur98] vorgestellt. Er berechnet mit einem herkömmlichen Renderer zunächst ein z-Buffer-Bild, aus dem er zum einen die Kanten im Modell (*Template Image*) und zum anderen ein sogenanntes Kraftfeldbild (*Force Field Image*) generiert. Diese beiden Bilder bilden den Ausgangspunkt für die Anwendung des Partikelsystems. Die Position eines Partikels wird pro Schritt durch den korrespondierenden Punkt des Kraftfeldbildes und weitere einstellbare (z. T. stochastischen) Parameter modifiziert. Trifft ein Partikel auf ein Gebiet, das keine Farbe benötigt (wo also keine

[6] Für ausführliche Informationen siehe `http://www.divideo.it/toonz/`

Kante vorhanden ist), stirbt er und ein neuer wird innerhalb des Bildes erzeugt. Damit wandern die Partikel über das Bild, und die so beschriebenen Liniensegmente bilden das Ausgabebild (vgl. Abbildung 2.11).

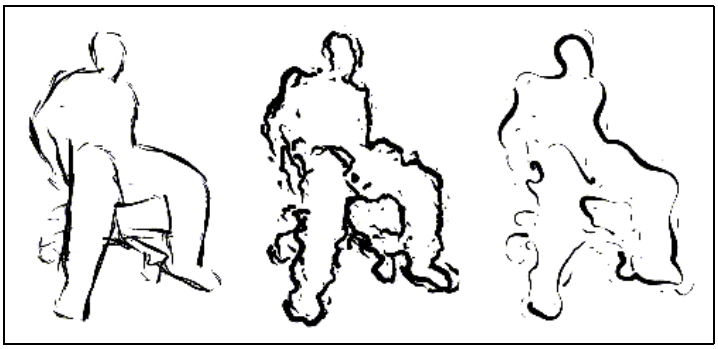


Abbildung 2.11: Unter Verwendung von *Loose and Sketchy Animation* erzeugte Bilder (aus [Cur98]).

Masuch, Schlechtweg und Schönwälder stellen in [MSS97] das daLi!-System vor, das zur Animation von Liniengraphiken benutzt wird und das auf dem in Abschnitt 2.2.2 vorgestellten Linienmodell nach L. Schumann aufbaut. Das daLi!-System benutzt zur Erzeugung der Ausgabelinien einen analytischen Renderer, um so auflösungsunabhängige Graphiken zu erzeugen. Die Animation erfolgt unter Verwendung von Keyframing auf zwei Ebenen. Die als Eingabe genutzten Geometriebeschreibungen und Animationsdaten von 3D Studio Release 4 werden zur Animation der Szenenobjekte verwendet. Zusätzlich wird aber auch die Animation der Linienstilparameter mittels Keyframes unterstützt. Abbildung 2.12 zeigt drei Einzelbilder einer mit daLi! erstellten Animation.

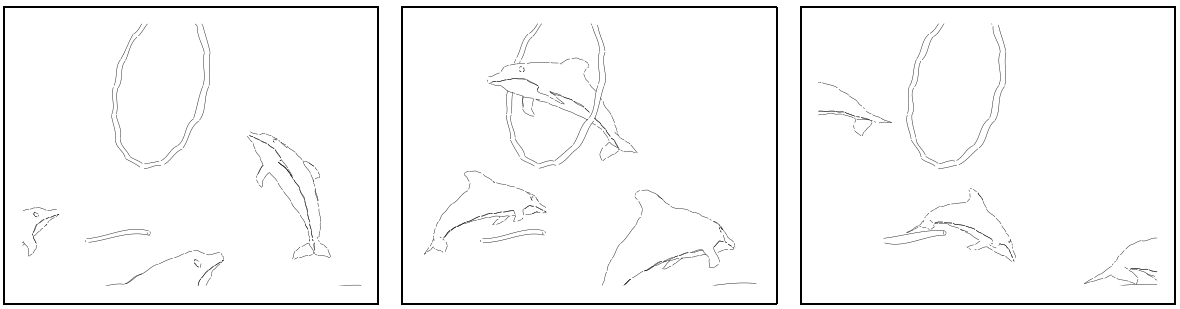


Abbildung 2.12: Drei Einzelbilder aus einer daLi!-Animation (aus [MSS97]).

In [MSS98] gehen Masuch, L. Schumann und Schlechtweg insbesondere auf die Animation der Linienstilattribute und des Linienstiles selbst ein, um die Animation zu illustrativen Zwecken einsetzen zu können (zur Anwendung des Systems vgl. beispielsweise [MS98]). Die Attribute Linienstärke und Liniensättigung können in daLi! linear zwischen zwei Werten interpoliert werden, wie in Abbildung 2.13 gezeigt ist. Ein besonderes Problem tritt beim Reduzieren der Attributwerte auf 0 auf, da die entsprechenden Objekte dadurch nicht mehr dargestellt werden. Andere Objekte werden aber immer noch durch das nicht mehr dargestellte Objekt verdeckt, da die sichtbaren Linien unter Beachtung aller Objekte berechnet wurden.

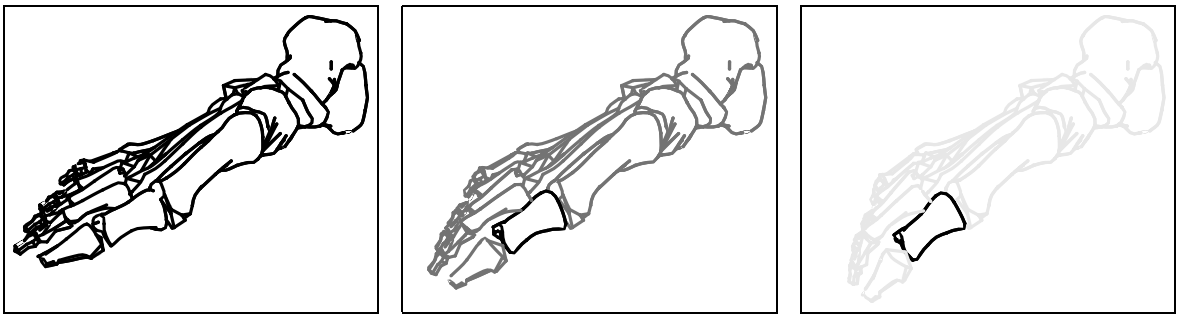


Abbildung 2.13: Gleichzeitige Animation von Position und Linienstil der Objekte mit dali! (nach [MSS97]).

2.4.3 Zeitliche Frame-Kohärenz

Eine wichtige Eigenschaft von ansprechenden Animationen ist die *zeitliche Frame-Kohärenz*. Dies bedeutet, daß zwei aufeinanderfolgende Bilder der Animation sich nur geringfügig unterscheiden dürfen, um eine ruhige Animation zu ermöglichen.

Viele nicht-photorealistische Techniken benutzen jedoch stochastische Methoden, um eine handerzeugte Wirkung zu simulieren. Dadurch kommt es aber bei der Animation der Graphiken zu Problemen. Durch die zufällige Veränderung von Parametern können sich zwei aufeinanderfolgende Bilder deutlich voneinander unterscheiden. Durch diese zeitliche Frame-Inkohärenz kommt es zu unruhigen Animationen. Dies ist meistens ungewollt, da die Bewegung vom eigentlichen Inhalt ablenkt.

Eine Möglichkeit, um zeitliche Frame-Kohärenz zu erreichen, ist daher das Verzichten auf stochastische Modifikationen. Dies liefert jedoch oft unbefriedigende Ergebnisse, da dadurch der Charakter der Graphiken verloren geht. Meier schlägt in [Mei96] vor, Startwerte des Zufallsgenerators (*Seeds*, engl. für „Samen“) abzuspeichern, um bei aufeinanderfolgenden Bildern die gleiche stochastische Modifikation der Elemente zu erreichen.

Durch die Verwendung von über ein Einzelbild hinaus existierenden Partikeln erreicht Curtis in seinem System zeitliche Frame-Kohärenz (vgl. [Cur98]). Die stochastischen Parameter haben nur einen Einfluß auf die zukünftige Bewegung der Partikel, nicht auf den bereits zurückgelegten Weg.

Masuch, L. Schumann und Schlechtweg betonen im Gegensatz dazu jedoch die Vorteile von zeitlicher Frame-Inkohärenz (vgl. [MSS98]). So beschreiben sie den Effekt, daß starke zufällige Veränderungen der Kontrollpunkte von Linien zwischen aufeinanderfolgenden Bildern einer Animation den Eindruck von Bewegung erwecken. Dies trifft vor allem auf runde Objekte und auf Szenen zu, die in keinem räumlichen Zusammenhang zu Fixpunkten stehen. Ein Beispiel für eine solche Szene (vgl. Abbildung 2.14) ist ein Auto auf einer Straße, wobei keine Häuser zu sehen und damit keine Fixpunkte vorhanden sind. Die Räder und der Straßenrand werden durch stark zufällig plazierte Linien dargestellt und erzeugen damit einen Bewegungseindruck.

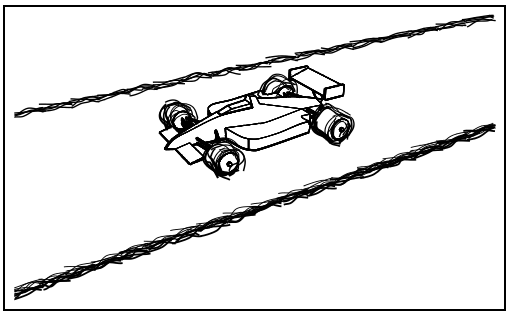


Abbildung 2.14: Bewegungseindruck durch zufällige Positionierung von Linien (mit freundlicher Genehmigung von Maic Masuch).

Als Stilmittel wird zeitliche Inkohärenz von liniengraphischen Animationen beispielsweise in der Fernsehwerbung der Red Bull GmbH eingesetzt.[7] In diesen Werbespots wird im Unterschied zum eben beschrieben Ansatz nicht versucht, konkret einen Eindruck von Bewegung zu vermitteln. Vielmehr wird durch die zeitliche Inkohärenz allgemein eine gewisse Dynamik und Lebendigkeit zum Ausdruck gebracht, die in einer kohärenten 2D-Animation nicht gegeben wäre und so subjektiv Eigenschaften des Produktes vermitteln soll.

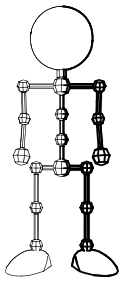
2.5 Zusammenfassung

In diesem Kapitel wurde ein Überblick über bestehende Techniken zur Generierung nicht-photorealistischer Graphiken und Animationen gegeben. Ebenfalls wurden bestehende Methoden zur Spezifikation solcher Animationen beschrieben. Des weiteren wurde die Eignung von Liniengraphiken als Illustrationen diskutiert.

Bei den Techniken zur Generierung nicht-photorealistischer Graphiken fällt auf, daß viele Verfahren eine 3D-Geometrie als Ausgangspunkt verwenden. Diese Methode wird auch in dieser Arbeit angewendet, da so Informationen über den Aufbau der Objekte vorhanden sind, die wichtig für die Erzeugung von Illustrationen sind. Diese Informationen werden beim traditionellen Erzeugen von Illustrationen durch das Wissen des Künstlers eingebracht. Für die maschinelle Erzeugung qualitativ hochwertiger Illustrationen stellt die Verwendung einer 3D-Geometrie also eine wichtige Voraussetzung dar.

Die vorgestellten nicht-photorealistischen Techniken wurden in pixelorientierte und vektororientierte unterschieden. Für diese Arbeit bietet sich ein vektororientiertes und damit analytisches Verfahren zur Erzeugung der Liniengraphiken an, da auflösungsunabhängige Graphiken beim Druck wesentliche Vorteile gegenüber Pixelbildern bieten, wie in Abschnitt 2.1 erwähnt wurde. Animationen können ebenfalls aus diesen Graphiken erzeugt werden, indem die Einzelbilder in der gewünschten Auflösung gerastert und anschließend aneinandergefügt werden. So können auch aus einer Renderingausgabe in Form von Vektorgraphiken Animationen in beliebiger Auflösung erzeugt werden, ohne erneut rendern zu müssen. Der umgekehrte Weg der Generierung von Vektorgraphiken aus Pixelbildern ist nicht immer einfach möglich.

[7] Zu Beispielen vgl. `http://www.redbull.com/de/cartoon-gallery.html`



Kapitel 3

Linienstileffekte

Für die Entwicklung eines liniengraphischen Illustrations- und Animationssystems ist eine geeignete Form der Spezifikation der Animation notwendig. In bezug auf die liniengraphische Komponente einer Animation treten dabei jedoch Besonderheiten im Vergleich zu herkömmlichen, photorealistischen Animationen auf. Insbesondere der Fakt, daß bei Liniengraphiken ein Attribut von mehreren Parametern beeinflußt werden kann, und die Verwendung mehrerer, zeitlich konkurrierender Einflüsse wirft Probleme auf. Aus diesem Grund muß nicht nur eine neuartige Beschreibung der Beeinflussung der Linienstilattribute gefunden werden, sondern auch eine daran angepaßte Möglichkeit der Kombination mehrerer dieser Beeinflussungen.

Zunächst wird daher der Begriff des *Linienstileffektes* eingeführt, der die Definition von objektabhängigen und objektunabhängigen Linienstilattribut-Beeinflussungen erlaubt. Anschließend werden verschiedene Ansätze zur Spezifikation und Kombination von Linienstileffekten für liniengraphische Animationen diskutiert. Sie werden auf ihre Eignung im vorliegenden Fall geprüft, und daraus wird ein Ansatz für eine mögliche Implementierung abgeleitet. Es wird dabei vor allem auf die Animation von Linienstileffekten eingegangen und Einzelbilder als ein Sonderfall von Animationen betrachtet. Abschließend wird die Einbettung von Linienstileffekten in ein bestehendes Renderingsystem für Liniengraphiken diskutiert.

3.1 Begriffsbestimmung

Wie in Abschnitt 2.2 beschrieben, orientieren sich die meisten Verfahren zur Erzeugung nicht-photorealistischer Graphiken an der Pinselstrichmetapher. Da diese eine Metapher im zweidimensionalen Raum ist, beschränken sich Visualisierungen in bisherigen Systemen (vgl. [MSS98] oder [MS98]) auf zweidimensionale Beeinflussungen des Erscheinungsbildes des Linienstiles wie die Anwendung eines anderen Linienstiles oder auch das Setzen eines objektglobalen Attributwertes zum Hervorheben von Objekten. Eine Ausnahme von dieser Beeinflussung der Linienstilattribute *auf Stilebene* oder *auf Objektebene* bildet nur einfaches *Depth Cueing*[1] oder eine einfache Beleuchtungsabhängigkeit wie durch SCHÖNWÄLDER implementiert (vgl. [Sch97a]).

[1] Darstellung der Tiefe von Objekten anhand bestimmter Attribute, z. B. anhand der Linienstärke

Eine derartige Strategie erfordert eine geeignete Unterteilung des Modells in Teilobjekte. Dies ist bei Modellen technischer bzw. mechanischer Natur meist gut möglich, da die entsprechenden Objekte durch ihre Herstellung aus Einzelteilen diese Unterteilung schon von vornherein besitzen. Insbesondere bei Modellen organischer Natur ist eine derartige geeignete Unterteilung aber nicht gegeben, da hier die Übergänge fließend sind. Des weiteren kommt es allgemein bei Modellen oft vor, daß logische Unterteilungen vorgenommen werden müssen, die nicht mit den gegebenen physischen Unterteilungen übereinstimmen. Einige Beispiele sollen die eben genannten Punkte illustrieren:

Beispiel 3.1 (Modelle technischer Natur)
Ein Fahrrad (vgl. Abbildung 3.1) ist aus Objekten wie Rahmen, Schrauben, Muttern, Speichen, Reifen etc. zusammengesetzt, die die einzelnen mechanischen Bauteile des Fahrrades darstellen. Eine Unterteilung in Einzelobjekte ist hier leicht möglich.

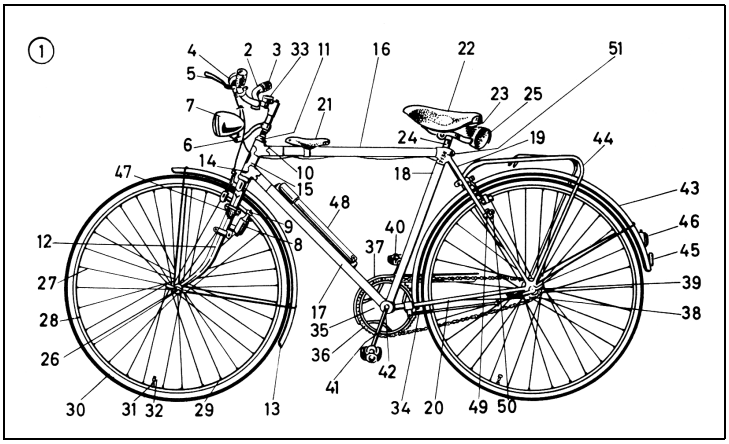
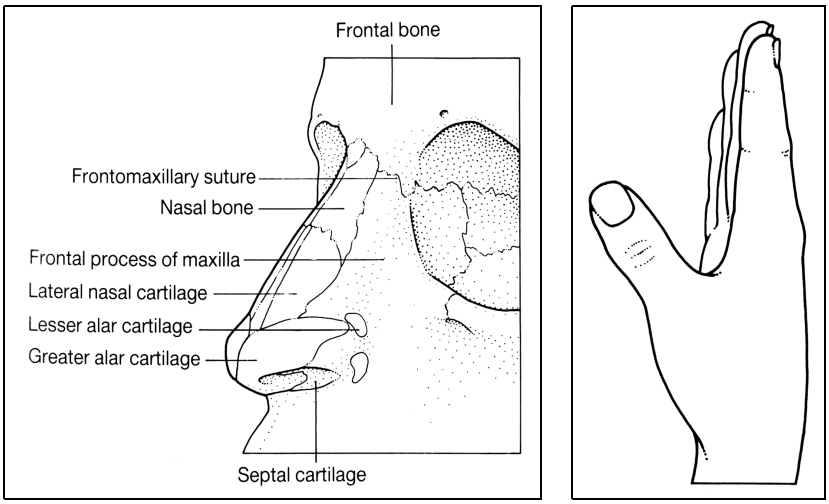


Abbildung 3.1: Zusammensetzung eines Fahrrades aus vielen Bauteilen (aus [Dud77, Seite 187]).

Beispiel 3.2 (Modelle organischer Natur)
Ein Modell eines menschlichen Kopfes läßt sich nicht einfach in Einzelobjekte unterteilen. Es ist u. a. nicht klar definierbar, wo die Nase anfängt und wo sie aufhört. Üblicherweise ist ein Schädel (ohne Unterkiefer) als ein Objekt modelliert, obwohl noch andere Strukturen identifizierbar sind, wie etwa der Nasenknochen bzw. die Knochen, die den Nasenansatz bilden (vgl. Abbildung 3.2(a)). Auch bei einer Hand (Modell mit Haut) ist nicht exakt spezifizierbar, wo ein Finger aufhört und wo die Handfläche beginnt (vgl. Abbildung 3.2(b)).

Beispiel 3.3 (Modelle mit logischer Unterteilung)
Eine übliche logische Unterteilung ist die in links und rechts, oben und unten, hinten und vorn. Ein Auto hat einen vorderen und einen hinteren Teil, genau wie eine linke und eine rechte Seite. Ein 3D-Modell eines Autos würde trotzdem keinen separaten rechten und linken Teil der Windschutzscheibe oder einen vorderen und einen hinteren Teil des Unterbodens als



(a) Die Knorpel und Knochen des externen Teils der Nase.

(b) Hand.

Abbildung 3.2: Beispiele für Modelle aus der Anatomie: es ist nicht klar spezifizierbar, wo bestimmte Strukturen beginnen oder enden (aus [Rog92, Seiten 509 bzw. 247]).

> Einzelobjekt haben, sondern die Windschutzscheibe als ein Objekt und den Unterboden als ein weiteres.

Je nach Visualisierungsaufgabe ist also eine andere geeignete Unterteilung eines Modells in Objekte notwendig. Nicht nur, daß eine geeignete Unterteilung nicht immer existieren muß, sie ist auch sehr aufwendig. Besser ist ein Verfahren, das unabhängig von der Aufteilung des Modells in Einzelobjekte die Linienstilattribute beeinflußt, also *unterhalb der Objektebene* arbeitet. Wie bereits in Abschnitt 2.2.2 angesprochen, ist eine Anwendung eines vordefinierten Linienstiles dazu nicht ausreichend, da in diesem Falle die Beeinflussung der Attribute zu einem Zeitpunkt stattfindet, zu dem noch keine Informationen über das Modell verfügbar sind.

Damit kann eine Definition des Begriffs Linienstileffekt gegeben werden, die sich nicht nur für Einzelbilder, sondern auch für Animationen eignet:

Definition 3.1

> Ein Linienstileffekt ist die Beeinflussung eines Linienstilattributes unter Verwendung von zusätzlichen Informationen, mit denen das 3D-Modell angereichert wurde. Er kann das Attribut objektspezifisch als auch objektunspezifisch beeinflussen.

Diese Definition wird im folgenden verwendet, um eine geeignete Technik der Kombination von Linienstileffekten zu finden. Dazu wird im folgenden zunächst das am häufigsten angewendete Animationsverfahren auf seine Eignung für die Animation von Liniengraphiken mittels Linienstileffekten untersucht.

3.2 Keyframing

Betrachtet man herkömmliche, photorealistische Animationen, so wird dort zur Spezifikation der Bewegungen der Objekte meist die Technik des Keyframings benutzt, wie sie in Abschnitt 2.3.3 vorgestellt wurde. Dieses Verfahren wird auf die Animation von Linienstilen mittels Linienstileffekten übertragen und die Vor- und Nachteile dieser Methode abgewogen.

Im einfachsten Fall bedeutet die Übertragung der Technik des Keyframings auf die Animation von Linienstileffekten, daß die Ausprägung der Attribute eines Stiles zu verschiedenen Zeitpunkten festlegbar ist, wie in [MSS97] und [MSS98] beschrieben. Dies betrifft Linienstärke, Liniensättigung, die Zufälligkeit der Linienführung und den Linienstil selbst. Weiterhin kann der Wirkungsbereich eines Keyframes auf einen bestimmten Knoten der Objekthierarchie[2] und den dadurch spezifizierten Teilbaum eingeschränkt werden, um so Objekte oder Objektgruppen gesondert zu behandeln (vgl. schematische Darstellung der Spezifikation von Keyframes für einige Objekte einer Objekthierarchie in Abbildung 3.3). Der folgende Pseudocode illustriert das Verfahren zur Bestimmung der Attributbeeinflussung zu einem bestimmten Zeitpunkt innerhalb der Animation:

```
PROZEDUR Bestimme_Attributbeeinflussung

FÜR jedes Objekt FÜHRE AUS: {
    BESTIMME die Keyframes für das aktuelle oder das in der Hierarchie
        nächsthöhere Objekt mit Keyframes;
    INTERPOLIERE die durch die Keyframes bestimmten Parameter;
    GENERIERE einen passenden Linienstil;
    WEISE den generierten Stil dem Objekt ZU;
};
```

Die durch objektgebundenes Verändern von Linienstilparametern erreichbaren Effekte sind für die Erzeugung von Illustrationen aus 3D-Modellen nicht ausreichend, wie bereits in Abschnitt 3.1 erläutert wurde. Es ist notwendig, die Linienstilattribute auch unterhalb der Objektebene zu beeinflussen. Daher muß eine objektunabhängige Form der Spezifikation der Beeinflussung von Attributen gefunden werden. Die kann durch eine Spezifizierung der Beeinflussung realisiert werden, anhand derer für bestimmte Kontrollpunkte die entsprechenden Attributwerte berechnet werden können. Die Art dieser Kontrollpunkte werden in Abschnitt 3.5 ausführlicher behandelt. Der folgende Pseudocode illustriert eine entsprechend erweiterte Form des Keyframings für die Animation von Linienstilattributen:

[2] Die Objekthierarchie muß nicht mit der Hierarchie übereinstimmen, die zur Animation der Objekte im herkömmlichen Sinne benutzt wurde.

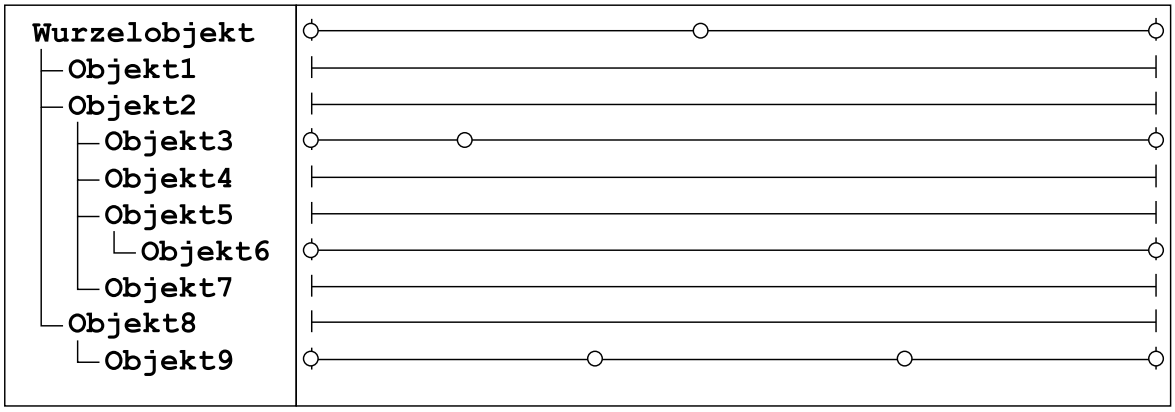


Abbildung 3.3: Spezifikation von Keyframes für Objekte in einer Hierarchie zur Bestimmung der Attributbeeinflussung. Jedem Objekt der Hierarchie ist ein Zeitstrahl zugeordnet, auf dem Keyframes angeordnet sein können.

```
PROZEDUR Bestimme_Attributbeeinflussung

FÜR jeden Kontrollpunkt FÜHRE AUS: {
    BESTIMME Position VOM Kontrollpunkt;
    BESTIMME das zugehörige Objekt;
    BESTIMME die Keyframes für das aktuelle oder das in der Hierarchie
        nächsthöhere Objekt mit Keyframes;
    INTERPOLIERE die durch die Keyframes bestimmten Parameter;
    BESTIMME die Attributwerte FÜR Position, Zeit;
    WEISE die bestimmten Attributwerte dem Kontrollpunkt ZU;
};
```

Beim Einsatz von Keyframing in der photorealistischen Animation wird ein Attribut nur von einem Parameter bestimmt. So ist die Position eines Objektes zu einem bestimmten Zeitpunkt durch die Angabe von x-, y- und z-Wert charakterisiert, also durch nur einen Parameter je Achse. Es soll jedoch möglich sein, verschiedenste Effekte zu kombinieren. Damit wird bei der Animation von Linienstilattributen ein Attribut von mehreren Parametern (die von den verschiedenen Effekten stammen) beeinflußt. Somit wächst die Anzahl der einstellbaren Parameter für die Beeinflussung eines Linienstilattributes mit steigender Anzahl der Effekte sehr schnell. Sollen beliebig viele Effekte für eine Szene zugelassen werden, ist der Einsatz von Keyframing in dieser Form für die Animation der Attribute nicht mehr praktikabel, da alle Parameter in einem Keyframe eingestellt und berücksichtigt werden müssen.

Des weiteren ist ein grundlegendes Problem zu nennen: Bei photorealistischen Animationen werden die Keyframes als Stützpunkte für eine andauernde Veränderung von Parametern definiert. Dies ist einsichtig, da sich die Parameter meist über den Verlauf der gesamten Animation verändern. Im Gegensatz dazu will man mit Linienstilen oft zeitlich begrenzte Effekte erreichen. Nutzt man jedoch Keyframes zur Spezifikation der Parameter, muß man sowohl über die ganze Animation Keyframes definieren als auch insbesondere die Start- und Endbedingungen durch mehrere Keyframes spezifizieren.

Es sind also relativ viele Einstellungen notwendig, um wenige Effekte zu erreichen. Dies trägt zur Unübersichtlichkeit und Unhandlichkeit des Ansatzes bei.

Als ein letzter Punkt sollen konkurrierende Ereignisse genannt werden. Man stelle sich einen Effekt vor, der Teile von Objekten um ein zu spezifizierendes Zentrum auf geeignete Weise hervorhebt, z. B. durch einen stärkeren Linienstil. Dieser Effekt wird sowohl zeitlich als auch räumlich konkurrierend mit einem weiteren derartigen Effekt angewendet, d. h. zu einem Zeitpunkt haben beide Effekte Einfluß auf den gleichen Kontrollpunkt. Um eine derartige Situation durch einen Keyframe beschreiben zu können, müssen mehrere Effekte in einem Keyframe spezifizierbar sein. Will man nur Parameter eines Effektes durch einen Keyframe verändern, müssen gleichzeitig die Parameter für alle weiteren für das betrachtete Objekt spezifizierten Keyframes so eingestellt werden, daß sie diese Keyframes nicht beeinflussen. Wird der neue Keyframe anschließend verschoben, ändert sich das Verhalten aller Keyframes, was in den meisten Fällen unerwünscht ist.

Aus den genannten Gründen erscheint der Ansatz des Keyframings zur Kombination der Linienstileffekte nur begrenzt geeignet, da das System zu komplex werden würde und dies dem Ziel einer einfachen Spezifikation der Effekte zuwiderläuft. Daher soll im folgenden ein weiterer Ansatz vorgestellt und diskutiert werden, der eine einfachere und übersichtlichere Spezifikation der Linienstileffekte erlaubt.

3.3 Effekthierarchie

Betrachtet man bestehende Systeme, die ähnlich wie im vorliegenden Fall Effekte in eine Animation einbinden können, so sind dies u. a. Videoschnittsysteme, wie z. B. der Alias|*wavefront* Composer oder Adobe Premiere. Diese benutzen einen hierarchischen Ansatz, um das schrittweise Hinzufügen unterschiedlicher Effekte zu bereits vorhandenen Animationen zu erreichen. Jeder Hierarchieknoten kann als eigenständiger Effekt aufgefaßt werden, womit das rekursive Zusammensetzen von Effekten ermöglicht wird.

Abbildung 3.4 zeigt einen Screenshot des Alias|*wavefront* Composer, in dem man die hierarchische Anordnung der Effekte sehen kann. Sie zeigt auch, daß Effekte einen eingeschränkten zeitlichen Wirkungsbereich haben und daß in der Hierarchie höher angesiedelte Effekte immer die in der Hierarchie unter ihnen liegenden Effekte zeitlich einschließen.

Ähnlich wie bei derartigen Videoschnittwerkzeugen kann zur Definition der Linienstileffekte auch eine Effekthierarchie aufgebaut werden (vgl. schematische Darstellung der Spezifikation einer Effekthierarchie in Abbildung 3.5). Ein Effektknoten definiert dazu, wie er das Attribut an einer beliebigen Stelle im Modell beeinflußt. Außerdem ist in jedem Knoten abgespeichert, wie er die Attributbeeinflussungen seiner Kinder mit seiner eigenen Beeinflussung zusammenfaßt. Dies kann durch verschiedene Funktionen geschehen, wie z. B. Maximumfunktion, Minimumfunktion, Multiplikation der Werte, Addition der Werte oder auch Durchschnittsfunktion.

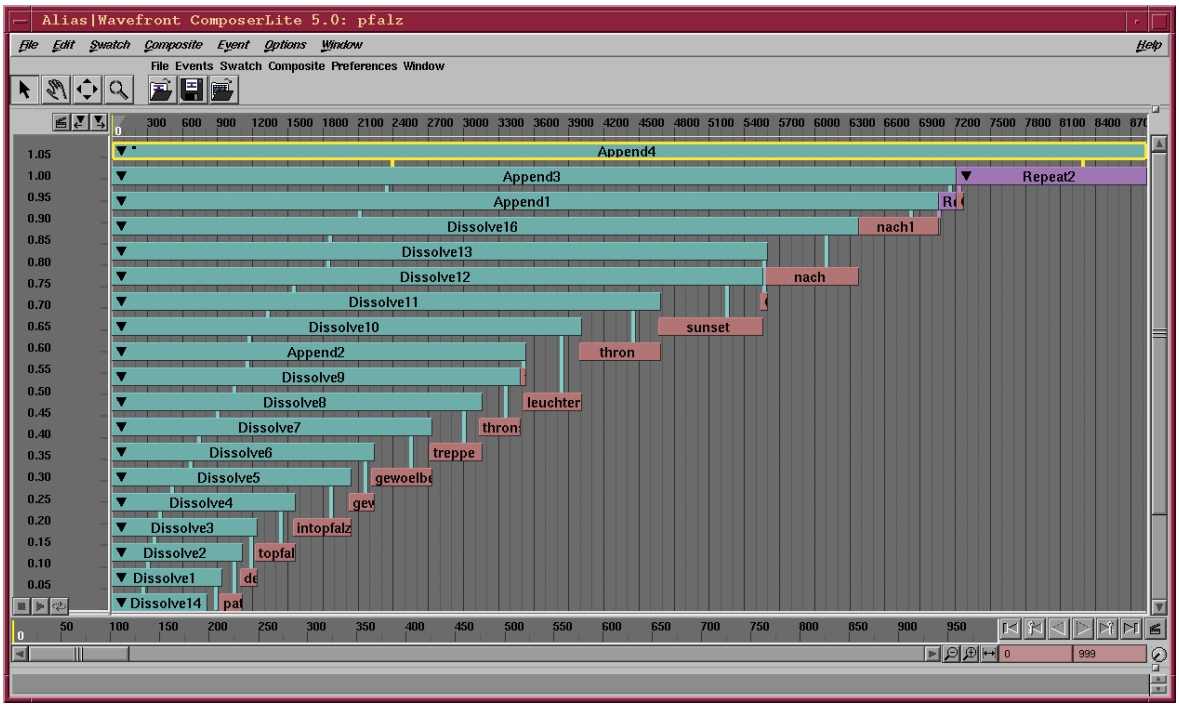


Abbildung 3.4: Alias|*wavefront* Composer. Erkennbar ist die hierarchische Anordnung der einzelnen Effekte und daß in der Hierarchie höher angesiedelte Effekte die in deren Unterbaum und damit tiefer liegenden zeitlich einschließen.

Der Wurzelknoten definiert zusätzlich noch einen Grundzustand, der von den verschiedenen Effekten abgewandelt werden kann. Wird ein bestimmter Punkt im Modell nicht von einem Effekt beeinflußt, so gelten für ihn die im Wurzelknoten spezifizierten globalen Linienstilattribute.

Wie bereits erwähnt, schließen bei Videoschnittwerkzeugen in der Hierarchie höher angeordnete Knoten die in ihrem Teilbaum befindlichen tiefer angeordneten Knoten zeitlich ein. Dies ist durch die Trennung von Effekt und Kombinationsfunktion bei den Videoschnittprogrammen bedingt, wodurch nur die Blätter des Hierarchiebaumes Effekte im Sinne von Definition 3.1 sind, alle anderen Knoten sind Kombinationsfunktionen. Letztere Knoten sind implizit über den gesamten Zeitraum der Animation wirksam, haben jedoch nur während des zeitlichen Einflußbereichs ihrer Kindknoten einen Effekt. Gleiches gilt für den hier vorgestellten Ansatz, nur daß hierbei Effekt und Kombinationsfunktion eine Einheit bilden und der zeitliche Wirkungsbereich des eigentlichen Effektes im Vordergrund steht. Im Unterschied zu den Hierarchien von Videoschnittwerkzeugen erscheinen somit tiefer angeordnete Knoten meist nicht zeitlich innerhalb der nächsthöheren Knoten. Der Vorteil dieses Ansatzes besteht in flacheren Hierarchien mit weniger Knoten.

Des weiteren ist bei den Videoschnittwerkzeugen der Einflußbereich der Effekte immer die zweidimensionale Bildebene. Dieser kann zwar maskiert werden, um die Beeinflussung einzuschränken, der Einfluß wirkt aber immer auf ein Objekt, nämlich das Bild. Im Gegensatz dazu wird der Einflußbereich bei der Linienstileffekthierarchie durch ei-

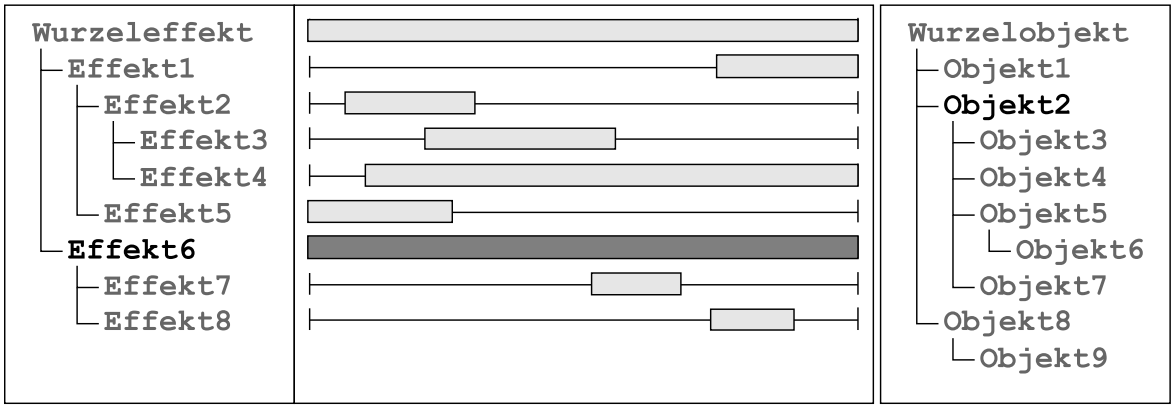


Abbildung 3.5: Spezifikation der Attributbeeinflussung mit Hilfe einer Effekthierarchie. Jeder Effektknoten ist visualisiert durch einen Block auf einem Zeitstrahl. Außerdem ist jedem Effekt ein Knoten der Objekthierarchie zugeordnet, auf der rechten Seite des Schemas dargestellt. Dieser charakterisiert dessen Einfluß- bzw. Geltungsbereich (exemplarisch an `Effekt6` und `Objekt2` gezeigt).

ne zweite Hierarchie bestimmt, um spezifisch bestimmte Objekte bzw. Objektgruppen beeinflussen zu können.

Damit ergibt sich zur Bestimmung des Einflusses der Linienstileffekte ein Algorithmus zur Bestimmung der Attributbeeinflussung zu einem bestimmten Zeitpunkt, der durch den folgenden Pseudocode illustriert wird.

```
PROZEDUR Bestimme_Attributbeeinflussung

FÜR jeden Kontrollpunkt FÜHRE AUS: {
    BESTIMME Position VON Kontrollpunkt;
    FÜR alle Attribute FÜHRE AUS: {
        Bestimme_Attributwert (Wurzelknoten, Attribut, Position,
            Zeit);
    };
    WEISE die bestimmten Attributwerte dem Kontrollpunkt ZU;
};
```

```
FUNKTION Bestimme_Attributwert (Knoten, Attribut, Position, Zeit)

FÜR alle Kinderknoten FÜHRE AUS: {
    Bestimme_Attributwert (Kinderknoten, Attribut, Position, Zeit);
};
BESTIMME eigenen Attributwert FÜR Position, Zeit;
FASSE Attributwerte mit Kombinationsfunktion ZUSAMMEN;
GIB zusammengefaßten Attributwert ZURÜCK;
```

Im Gegensatz zum Keyframing ist es mit dem Effekthierarchieansatz möglich, sowohl zeitliche als auch räumliche Konkurrenzsituationen geeigneter und intuitiver behandeln zu können. Die Effekte können einfacher spezifiziert werden, da es nicht mehr

notwendig ist, viele meist ähnlich lautende Einstellungen für einen einzigen Effekt zu tätigen.

Jedoch auch dieser Ansatz hat durch das Benutzen einer reinen Effektansicht Nachteile. So ist es z. B. nur unter großem Aufwand möglich, die Parameter der Effekte zeitlich zu variieren. Man kann nur relativ statische Ein- und Ausblendeffekte erreichen. Die Möglichkeit der zeitlichen Variation der Parameter ist aber eine wichtige Option, da die Effekte auch in Animationen eingesetzt werden sollen.

Da beide bisher besprochenen Ansätze Nachteile mit sich bringen, wird im folgenden ein dritter Ansatz vorgestellt, der sowohl die Nachteile beseitigt als auch die Vorteile von Keyframing und Effekthierarchie in sich vereinigt.

3.4 Kombination von Keyframing und Effekthierarchie

Der eben beschriebene Nachteil der Statik der Effekte bei der Anwendung der Effekthierarchie ist beim Keyframing nicht vorhanden. Um also die Statik zu überwinden, muß ein Keyframing der Attribute durchgeführt werden. Im Gegensatz zum bereits vorgestellten Ansatz wird dies aber nicht *global*, sondern nur *lokal* pro Hierarchieknoten bzw. Effekt angewendet (vgl. schematische Darstellung in Abbildung 3.6).

Der Algorithmus zum Bestimmen des Attributwertes eines Effektes muß daher gegenüber der oben beschriebenen Prozedur wie folgt abgeändert werden, um *lokales Keyframing* zu ermöglichen:

```
FUNKTION Bestimme_Attributwert (Knoten, Attribut, Position, Zeit)

FÜR alle Kinderknoten FÜHRE AUS: {
    Bestimme_Attributwert (Kinderknoten, Attribut, Position, Zeit);
};
BESTIMME die aktuellen Keyframes FÜR Zeit;
INTERPOLIERE die durch die Keyframes bestimmten Parameter;
BESTIMME den Attributwert FÜR Position, Zeit;
FASSE Attributwerte mit Kombinationsfunktion ZUSAMMEN;
GIB zusammengefaßten Attributwert ZURÜCK;
```

Dieses lokale Linienstil-Keyframing hat den Vorteil, daß die Anzahl der das Attribut beeinflussenden Parameter nur vom jeweiligen Effekt abhängt. Somit bleibt die Anzahl der möglichen Einstellungen pro Effekt übersichtlich. Auch müssen im Gegensatz zum reinen Keyframing nicht mehr „unnötige“ Keyframes außerhalb des eigentlichen Effektes definiert werden, die die Anfangs- und Endsituation festlegen. Der Effekt als solcher wird unabhängig von der Existenz anderer Effekte und kann unabhängig davon beeinflußt werden.

Dieses Verfahren wurde für die Implementierung des in Kapitel 5 beschriebenen Systems verwendet. Die Verwendung einer Hierarchie mit lokalem Keyframing für die

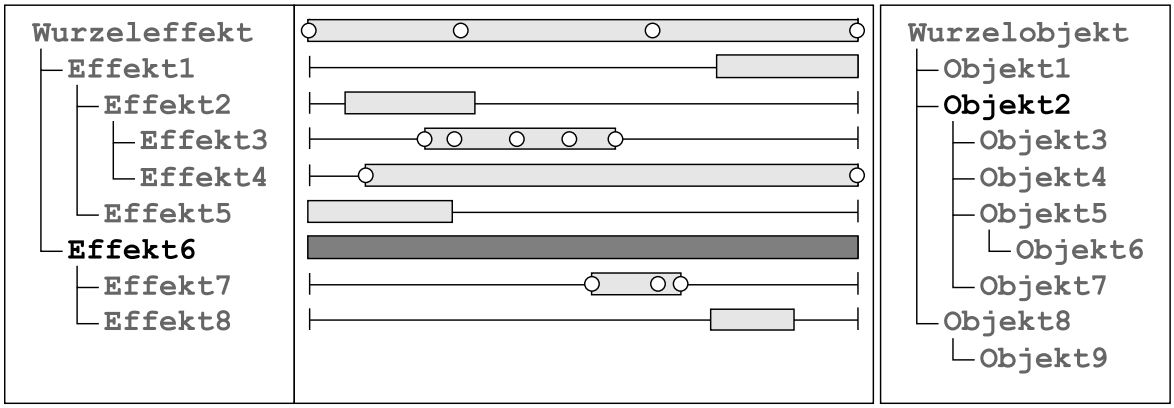


Abbildung 3.6: Spezifikation der Attributbeeinflussung mit Hilfe einer Effekthierarchie und lokalem Keyframing. Jeder Effektknoten ist visualisiert durch einen Block auf einem Zeitstrahl, auf dem bei Anwendung von lokalem Keyframing die Keyframes angeordnet sind. Außerdem ist jedem Effekt ein Knoten der Objekthierarchie zugeordnet, auf der rechten Seite des Schemas dargestellt. Dieser charakterisiert dessen Einfluß- bzw. Geltungsbereich (exemplarisch an `Effekt6` und `Objekt2` gezeigt).

Spezifikation von Linienstileffekten macht allerdings eine Anpassung eines üblichen Liniengraphik-Renderingsystems notwendig, die im folgenden beschrieben wird.

3.5 Systementwurf

Eine typische Rendering-Pipeline eines analytischen Linienrenderers wird in [Str98, Kapitel 4] von Schlechtweg und Raab beschrieben (vgl. Abbildung 3.7). Die darin enthaltenen Schritte sehen jedoch keine Beeinflussung von Linienstilattributen unterhalb der Objektebene vor. Daher muß die Rendering-Pipeline erweitert werden, um den im vorigen Abschnitt vorgestellten Ansatz zur Spezifikation von Linienstileffekten mittels einer Effekthierarchie mit lokalem Keyframing realisieren zu können.

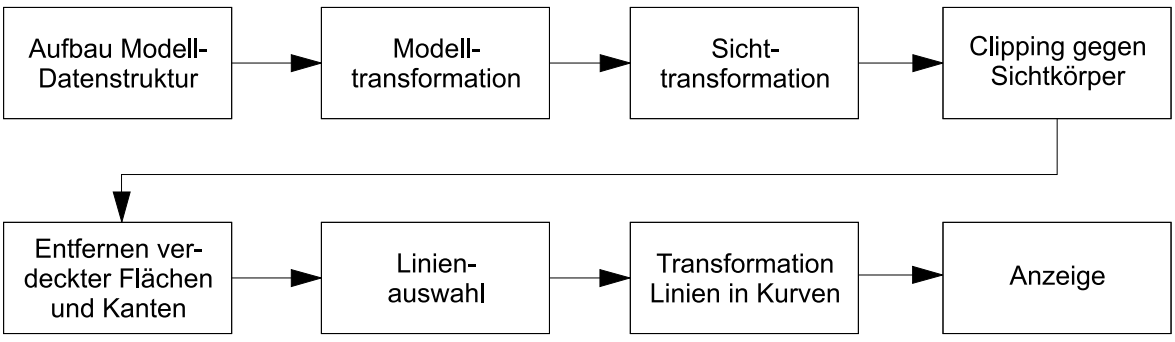


Abbildung 3.7: Rendering-Pipeline eines analytischen Linienrenderers (nach [Str98, Kapitel 4]).

Der letzte Schritt der genannten Rendering-Pipeline, die Anzeige der Liniengraphik, teilt sich bei der Verwendung des Linienmodells nach L. Schumann in folgende Schritte auf (vgl. [Sch97b]):

1. Berechnung eines polygonalen Ausgabelinienzuges (bestimmt durch sogenannte *Kurvenpunkte*) durch parametrische Überlagerung von Pfad- und Stilpositionen, Schrittweite je nach gewünschtem Detaillierungsgrad
2. Bestimmung der Attributwerte an den Kurvenpunkten des Polygonzuges durch parametrische Überlagerung der Werte von Pfad und Stil
3. Umsetzung der Werte für das Attribut Linienstärke in eine Anzahl von Polygonen
4. Füllen dieser Polygone anhand der Werte für das Attribut Liniensättigung
5. Ausgabe der Polygone

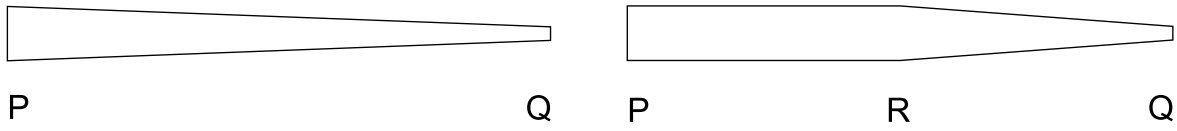
Damit ergeben sich für die Einflußnahme auf die Werte der Attribute prinzipiell 3 Möglichkeiten: die Beeinflussung der Attributwerte an

- den Kontrollpunkten des Stiles,
- den Kontrollpunkten des Pfades und
- den Kurvenpunkten des Polygonzuges.

Daß die zuerst genannte Variante nicht die gewünschten Ergebnisse liefert, wurde bereits in Abschnitt 3.1 erläutert. Das Linienmodell nach L. Schumann erlaubt die als zweite Möglichkeit genannte Attributierung des Pfades und damit die Beeinflussung von Linienstilattributen in Abhängigkeit von zusätzlichen Informationen, mit denen das Modell angereichert wurde. Bei Verwendung dieser einfachen Möglichkeit zur Beeinflussung von Attributwerten treten jedoch bei grob triangulierten Objekten Probleme mit der Genauigkeit des Einflusses auf, die im folgenden an einem Beispiel erläutert werden.

Gegeben sei ein grob trianguliertes Objekt, wie etwa ein Würfel. Eine Kante des Objektes sei durch die Punkte P und Q beschrieben (vgl. Abbildung 3.8). Ein bestimmter Linienstileffekt legt die Linienstärke von Punkt P auf 1, die von Punkt Q auf 0,25 fest. Eine lineare Interpolation der Werte angenommen, wird die Linie als gleichmäßig dünner werdende Linie ausgegeben. Im zweiten Fall sei der gleiche Würfel doppelt so fein trianguliert. D. h., auf der Hälfte der Strecke zwischen P und Q existiert in diesem Fall ein dritter Punkt R. Der im ersten Falle verwendete Linienstileffekt sei aber so definiert, daß er R in der gleichen Weise wie P beeinflußt. Dies würde im ersten Fall zu keinen Änderungen führen. Wird nun eben dieser Effekt auch im zweiten Falle angewendet, so ist die Ausgabe eine Linie, die zuerst konstant stark ist und ab der Hälfte rasch dünner wird. Bei gleichem Linienstileffekt kann es also bei unterschiedlich feiner Triangulierung von Objekten zu einem unterschiedlichen Ergebnis kommen.

Diese Nachteile hat die Variante der Beeinflussung der Attributwerte an den Kurvenpunkten des Polygonzuges nicht, da diese unabhängig von der Triangulierung der Modellobjekte ist. Außerdem kann die Schrittweite zur Erzeugung des Polygonzuges der verwendeten Auflösung des Ausgabemediums angepaßt werden. Aufgrund dieser Vorteile sollte die Attributbeeinflussung nicht nur an allen Pfadkontrollpunkten der Ausgabelinien bestimmt werden, sondern an allen Kurvenpunkten der Ausgabelinien.



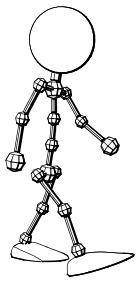
(a) grobe Triangulierung (b) feine Triangulierung

Abbildung 3.8: Unterschiedlicher Verlauf der Linienstärke einer Kante eines Objektes bei gleichem Linienstileffekt aber unterschiedlicher Triangulierung des Objektes.

Die Kurvenpunkte existierten bisher jedoch ausschließlich als zweidimensionale Punkte im Bildkoordinatensystem. Um die Beeinflussung der Linienstileffekte an den Kurvenpunkten berechnen zu können, müssen diese jedoch auch als dreidimensionale Punkte vorliegen. Daher werden Kurvenpunkte in der vorliegenden Arbeit weiter gefaßt. Ein Kurvenpunkt verfügt neben seinen zweidimensionalen Bildkoordinaten auch über die dreidimensionalen Koordinaten seiner Entsprechung im Kamera- und Weltkoordinatensystem (vgl. auch Abschnitt 5.2).

Mit dieser Erweiterung kann die Schrittfolge der Ausgabe der Liniengraphik angepaßt werden, indem anstelle von Punkt 2 der Punkt „Bestimmung der Attributbeeinflussung der Linienstileffekte an den Kurvenpunkten“ eingefügt wird. Ein Linienrenderer mit einer derartig veränderte Rendering-Pipeline ist in der Lage, die Linienstileffekte mittels einer Effekthierarchie mit lokalem Keyframing auf die Renderingausgabe anzuwenden.

Mit dem Verfahren der Effekthierarchie mit lokalem Keyframing steht somit nun eine Methode zur Verfügung, mit der Linienstileffekte auf einfache Weise miteinander verknüpft werden können, ohne die Unabhängigkeit der einzelnen Effekte zu beeinträchtigen. Darüber hinaus wurde gezeigt, daß das Verfahren mit wenigen Modifikationen in bestehende Liniengraphik-Renderingsysteme integrierbar ist.



Kapitel 4

Entwurf von Linienstileffekten

Im vorangegangenen Kapitel wurde der Begriff Linienstileffekt eingeführt und ein Konzept zur Verknüpfung von Linienstileffekten entworfen und diskutiert. Im folgenden werden exemplarisch einige konkrete Effekte beschrieben, die über das triviale Hervorheben ganzer Objekte hinausgehen und mit dem beschriebenen Konzept zu verwirklichen sind. Vorher wird jedoch zuerst auf einige Gemeinsamkeiten der entwickelten Effekte eingegangen.

4.1 Gemeinsamkeiten der Linienstileffekte

Definition 3.1 erlaubt die Spezifizierung verschiedenster Effekte. Wie in Abschnitt 2.5 aber bereits erläutert wurde, sind Informationen über den dreidimensionalen Aufbau der Szene für die Generierung von qualitativ hochwertigen computergenerierten Illustrationen von entscheidender Bedeutung. Aus diesem Grund ist eine zweidimensionale Beeinflussung der Linienstilattribute durch Linienstileffekte nicht ausreichend. Die im folgenden vorgestellten Effekte haben dementsprechend gemein, daß sie die Linienstilattribute im dreidimensionalen Raum beeinflussen und somit die zusätzlich vorhandenen Informationen ausnutzen.

Darüber hinaus wird zu jedem Effekt eine Funktion $f(t)$ mit $t \in [0;1]$ definiert. Sie steuert die Stärke des Effektes entlang einer Richtung im dreidimensionalen Raum, wobei sowohl diese Richtung als auch die Grenzen des Definitionsbereiches der Funktion ($t = 0$ bzw. $t = 1$) abhängig von den Parametern des Effektes sowie dem betrachten Punkt sind. Über diese Funktion ist es dem Nutzer möglich, die Ausprägung des Effektes zu beeinflussen. Die Beschränkung auf genau eine Funktion erscheint sinnvoll, da durch mehrere Funktionen die Wirkung unübersichtlich würde.

Der Algorithmus zur Bestimmung der Attributwerte ist bei allen Linienstileffekten grundsätzlich in zwei Phasen unterteilt. Die erste Phase dient der Vorbereitung der eigentlichen Berechnung der Werte, welche anschließend in der zweiten Phase vollzogen wird. Die Vorbereitung der Bestimmung der Attributwerte erfolgt dabei auf Basis der n Punkte der im Modell definierten Objekte. Im Gegensatz dazu wird die so vorbereitete Berechnung mit allen während des Zeichenprozesses bestimmten m Kurvenpunkten durchgeführt (vgl. auch Abschnitt 5.3.1).

4.2 Plane-Sweep-Effekt

Bisherige Systeme, wie beispielsweise durch SCHÖNWÄLDER in [Sch97a] vorgestellt, bieten zur objektunabhängigen Beeinflussung der Linienstilattribute nur wenige Möglichkeiten. Eine dieser Möglichkeiten ist die Verwendung von Depth Cueing, das eine eindimensionale Beeinflussung in Blickrichtung der Kamera darstellt. Diese Einschränkung auf eine Richtung sowie auf das Kamerakoordinatensystem erscheint jedoch unnötig. Der im folgenden vorgestellte Linienstileffekt stellt einen allgemeineren Ansatz dar, mit dem sich neben Depth Cueing auch andere Beeinflussungen der Linienstilattribute erreichen lassen.

4.2.1 Definition des Effektes

Der Plane-Sweep-Effekt ähnelt in seiner Wirkungsweise *Plane Sweep*[1] genannten Verfahren. Analog zu diesen Verfahren wird eine Richtung in der Szene definiert, charakterisiert durch den sogenannten *Effektstrahl* (zur Veranschaulichung siehe Abbildung 4.1). Er verläuft durch den Koordinatenursprung und wird durch eine Transformation eines auf der x-Achse liegenden Strahls in seine Position in der Szene gebracht. Um alle möglichen Richtungen eines solchen Effektstrahls beschreiben zu können, sind zwei Werte notwendig: die Rotation um die z-Achse (Deklination, im Bereich von $-90°$ bis $90°$) und die Rotation um die y-Achse (Azimut, im Bereich von $0°$ bis $360°$). Aus diesen Parametern kann die Strahlprojektionsmatrix bestimmt werden, die später für die Berechnung benötigt wird. Eine senkrecht zum Effektstrahl liegende und entlang dieses Strahles von negativ unendlich bis positiv unendlich durch den Raum verschobene Ebene definiert durch ihre erste und ihre letzte Berührung mit einem umschließenden Körper der zu betrachtenden Objekte ein *aktives Segment* auf dem Effektstrahl. Die senkrechten Projektionen der Berührungspunkte auf den Effektstrahl (P_0 und P_1) seien durch die Vektoren $\overrightarrow{p_0}$ und $\overrightarrow{p_1}$ beschrieben.

Über diesem aktiven Segment sei eine Funktion $f(t)$ definiert, die das Verhalten der Linienstilattribute an den zugehörigen Kurvenpunkten beschreibt. Für einen beliebigen Punkt P sei P' der auf den Effektstrahl projizierte Punkt. Für den zu P' gehörenden Vektor $\overrightarrow{p}$ sei t als

$$t = \frac{\|\overrightarrow{p} - \overrightarrow{p_0}\|}{\|\overrightarrow{p_1} - \overrightarrow{p_0}\|} \tag{4.1}$$

definiert. Damit hat t am Beginn des aktiven Segmentes den Wert 0, und steigt linear entlang der Richtung des Effektstrahles auf den Wert 1 am Ende des aktiven Segmentes an.

[1] Der Begriff Plane Sweep kommt aus der algorithmischen Geometrie und bezeichnet Algorithmen, bei denen ein Problem mittels einer $(d-1)$-dimensionalen Ebene gelöst wird, die eine Menge von Objekten im $\mathbb{R}^d$ überstreicht.

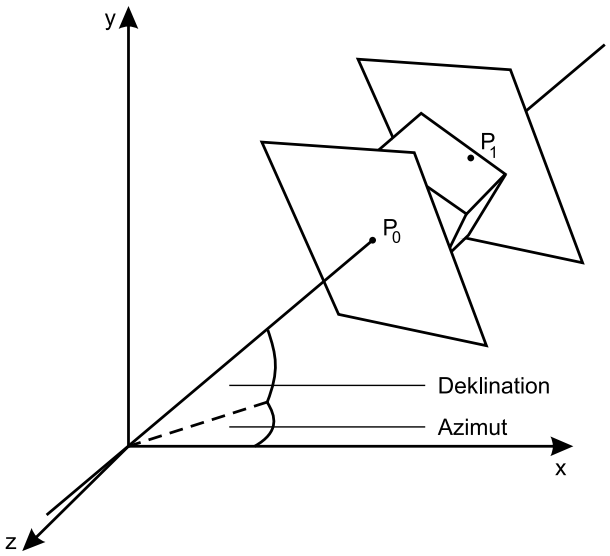


Abbildung 4.1: Skizze des Plane-Sweep-Effektes. Der Effektstrahl ist durch die Werte Azimut und Deklination beschrieben. Durch die Punkte P_0 und P_1, die durch die erst- und letztmalige Berührung einer entlang des Strahles verschobenen und zu ihm senkrechten Ebene mit dem Objekt gewonnen wurden, wird das aktive Segment definiert.

4.2.2 Umschließende Körper

Als umschließender Körper ist zunächst die Verwendung eines *umschließenden Quaders* (*Bounding Box*) naheliegend. Ein großer Nachteil eines umschließenden Quaders besteht jedoch darin, daß er (bei den meisten Objekten) nicht in alle Richtungen eng am Objekt anliegt. Zum anderen ist seine Größe und damit die Wirkung des Effektes stark von der Lage des Objektes in bezug auf das Welt- bzw. Objektkoordinatensystems abhängig. Insbesondere achsenparallele umschließende Quader erzeugen so eine schlechte Approximation der betrachteten Objekte. Mit dem Objekt transformierte umschließende Quader erzeugen bessere Ergebnisse, sind aber nur in seltenen Fällen (wie beispielsweise bei einem Quader) optimal für den Algorithmus geeignet, da auch sie von der Lage des Objektes zum Koordinatensystem zum Referenzzeitpunkt abhängig sind.

Besser erscheint daher die Nutzung einer *konvexen Hülle*. Ihre Form und Größe ist unabhängig von der Lage der Objekte zu den Koordinatenachsen. Außerdem liegt sie in allen Richtungen eng am Objekt an. Daher eignet sie sich gut als umschließendes Objekt.

Eine weitere interessante Option ist die Nutzung einer *umschließenden Kugel* (*Bounding Sphere*). Dies hat gegenüber der Verwendung einer konvexen Hülle den Vorteil, daß die Länge des aktiven Segmentes unabhängig von der Lage der betrachteten Objekte im Raum immer konstant bleibt. Dieser Umstand ist bei sich drehenden Objekten von Bedeutung, da dadurch ein konstanter Bezug der Funktion $f(t)$ gewährleistet ist. So muß die Funktion nicht an die sich ändernde Länge des aktiven Segmentes angepaßt werden.

Beide zuletzt genannten umschließenden Körper ermöglichen die Spezifikation unterschiedlicher Effekte, es ist nicht einer besser oder schlechter geeignet. Daher sollten beide Optionen in eine mögliche Implementierung einbezogen werden.

4.2.3 Bestimmung der Attributwerte

Zur Vorbereitung der Bestimmung der Attributwerte eines Plane Sweep-Effektes wird zunächst eine Transformationsmatrix bestimmt, so daß sich die Position eines Punktes innerhalb des aktiven Segmentes leichter bestimmen läßt. Dazu werden alle Punkte der die relevanten Objekte beschreibenden polygonalen Gitter ggf. zuerst in Kamerakoordinaten transformiert. Anschließend werden die Punkte mittels der inversen Strahlprojektionsmatrix projiziert. Diese beiden Transformationen werden in der Transformationsmatrix M abgespeichert. Nach der Projektion werden von allen Punkten die Extrempunkte in x-, y- und z-Richtung bestimmt, wobei die Extrempunkte in x-Richtung die Grenzen des transformierten aktiven Segmentes (nun auf der x-Achse) charakterisieren. Bei Nutzung einer umschließenden Kugel wird diese vor der Anwendung der Projektion bestimmt, nur der Mittelpunkt der Kugel transformiert und anschließend unter Nutzung des Radius die Extrempunkte bestimmt. Die Extremwerte in y- und z-Richtung sind für die Visualisierung wichtig, um einen Mittelpunkt des Objektes berechnen zu können (vgl. Abschnitt 5.4.1).

Zum eigentlichen Bestimmen des Attributwertes eines Kurvenpunktes P wird zuerst dessen Position in Welt- bzw. Kamerakoordinaten bestimmt. Anschließend wird der Punkt mittels der bei der Vorbereitung ermittelten Transformationsmatrix M projiziert. Der Parameter t_P wird nun durch Vergleich des x-Anteils des projizierten Punktes mit den zuvor bestimmten transformierten Grenzen des aktiven Segmentes mittels Gleichung 4.1 bestimmt. Durch Einsetzen des ermittelten Parameters t in die zuvor definierte Funktion $f(t)$ wird schließlich der Attributwert für den Kurvenpunkt berechnet.

4.2.4 Komplexitätsbetrachtungen

Die Praktikabilität eines Algorithmus und damit dessen Eignung für die Lösung eines Problems hängt wesentlich von dessen Laufzeitkomplexität ab. Im folgenden soll daher gezeigt werden, daß die zeitliche Komplexität der Berechnung des aktiven Segmentes nur linear von der Anzahl der Objektpunkte abhängt und die eigentliche Bestimmung des Attributwertes an einem Punkt P in konstanter Zeit realisierbar ist.

Die Berechnung einer konvexen Hülle einer Punktmenge im dreidimensionalen Raum kann für n Punkte mit einer Komplexität von $O(n \log n)$ (sowohl obere Schranke als auch zu erwartende Komplexität) durchgeführt werden. Da die Anzahl der Punkte der Hülle im Vergleich zur ursprünglichen Punktmenge immer noch in der Größenordnung von n sein kann, hat die Projektion der Punkte eine Komplexität von $O(n)$.

Nun werden aber nicht alle Informationen der konvexen Hülle benötigt, sondern nur die Extrempunkte in bezug auf das Koordinatensystem nach der Projektion. Diese Punkte können aber auch in linearer Zeit aus den projizierten Punkten gewonnen werden. Die konvexe Hülle muß also gar nicht berechnet werden. Statt dessen werden die Informationen eines umschließenden Quaders nach der Projektion benötigt. Weil die

Länge des aktiven Segmentes jedoch die gleiche ist wie die bei der Berechnung mit einer konvexen Hülle und die Berechnung des aktiven Segmentes auf diese Art einfacher vorstellbar ist, wird weiterhin vom Modus „Konvexe Hülle“ gesprochen.

Die Projektion der Gitternetzpunkte (Matrixmultiplikation in $\mathbb{R}^4$) ist in linearer Zeit in Abhängigkeit von der Anzahl der Punkte durchführbar. Damit ergibt sich als Gesamtkomplexität für die Bestimmung des aktiven Segmentes im Modus „Konvexe Hülle“ $O(n)$.

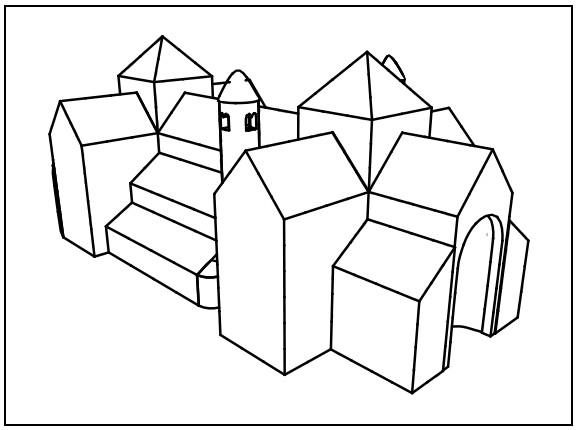
Die Berechnung einer umschließenden Kugel einer beliebigen Punktmenge mit n Punkten mit dem Algorithmus nach Welzl hat im $\mathbb{R}^d$ eine erwartete Komplexität von $O(d(d+1)!\,n)$ [Wel91]. Der Algorithmus beruht auf dem inkrementellen Einfügen von Punkten in eine Datenstruktur in einer zufälligen Reihenfolge. Die angegebene Komplexität wird über alle möglichen Reihenfolgen gemittelt und stellt keine *Worst-Case*-Schranke (Schranke im schlechtesten Fall) dar, sondern die Komplexität im durchschnittlichen Fall. Dies ist aber im vorliegenden Fall durchaus ausreichend, da die Berechnung nicht nur einmal durchgeführt werden muß und somit im Durchschnitt die genannte Komplexität zu erwarten ist. Im dreidimensionalen Raum ergibt sich durch die konstante Dimension eine erwartete Komplexität von $O(n)$. Bei der anschließenden Projektion der Kugel muß nur der Mittelpunkt betrachtet werden, was in konstanter Zeit passiert. Auch die folgende Berechnung der Extrempunkte ist in $O(1)$ realisierbar, da es sich um eine konstante Anzahl von betrachteten Werten handelt. Für die Vorbereitungsphase beträgt die Gesamtkomplexität in diesem Fall also auch $O(n)$.

Die Berechnung des Parameters t für einen gegebenen Kurvenpunkt ist in $O(1)$ durchführbar, da nur eine Matrixmultiplikation in $\mathbb{R}^4$ und eine anschließende Auswertung der Gleichung 4.1 mit den bei der Transformation ermittelten Werten nötig ist. Für die Berechnung des Attributwertes an allen m Kurvenpunkten beträgt die Komplexität also $O(m)$.

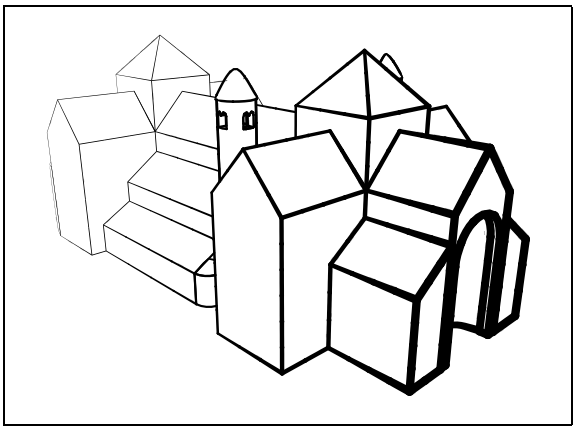
4.2.5 Modifikationen

Nachdem die Einzelheiten des Plane-Sweep-Effektes vorgestellt wurden, sollen nun noch einige Modifikationen bzw. weitere Optionen des Algorithmus besprochen werden. Daß die Rotation des Effektstrahls durch die Szene und damit die Orientierung des aktiven Segmentes sowohl in bezug auf das Weltkoordinatensystem als auch in bezug auf das Kamerakoordinatensystem betrachtet werden kann, wurde oben bereits erläutert. Damit lassen sich zum einen Effekte in Abhängigkeit von Objekten erreichen. Dazu ist es aber meist notwendig, daß bei sich ändernder Orientierung der Objekte im Raum auch die Orientierung des aktiven Segmentes angepaßt wird. Zum anderen lassen sich aber auch kameraorientierte Effekte wie beispielsweise Depth Cueing erzeugen. Siehe dazu auch Abbildung 4.2, in der die verschiedenen Varianten an einem Beispiel dargestellt sind. Abbildung 4.2(a) zeigt die Graphik ohne Anwendung eines Effektes, in Abbildung 4.2(b) wurde der Plane-Sweep-Effekt entlang der Objektachse angewendet

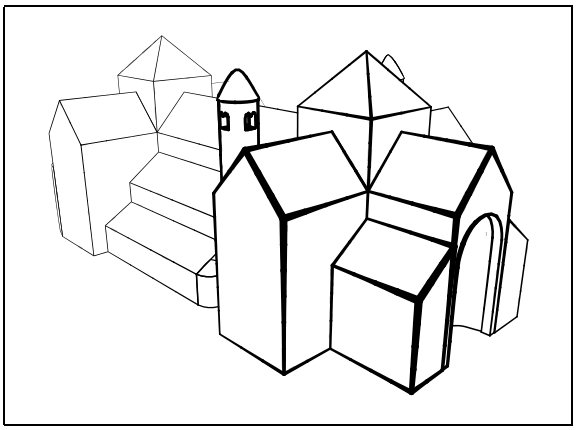
und in Abbildung 4.2(c) wurde ein Depth Cueing dargestellt, der Plane-Sweep-Effekt also in Kamerarichtung angewendet.



(a) Normale Liniengraphik.



(b) Veränderung der Linienstärke entlang einer Achse des Weltkoordinatensystems.



(c) Depth Cueing in bezug auf das Kamerakoordinatensystem.

Abbildung 4.2: Beispiele zum Bezug auf verschiedene Koordinatensysteme. Die Effekte wurden absichtlich übertrieben angewendet, um ihre Wirkungsweise deutlich zu illustrieren.

Wie oben beschrieben, können als eine Variante bei der Berechnung des aktiven Segmentes nur die Punkte der Objekte des aktuellen Hierarchieknotens herangezogen werden (lokale Berechnung). Alternativ können aber auch alle Punkte der Szene benutzt werden (globale Berechnung), um so einfach einen konstanten Bereich spezifizieren zu können, durch den sich ein Objekt hindurchbewegt. Dies ist aber auch nützlich, wenn bei zwei Objektgruppen von jeweils unterschiedlichen Effekten zu einem für beide Objektgruppen einheitlichen Effekt übergeblendet werden soll und man durch die globale Berechnung den gleichen Bezugspunkt hat.

4.2.6 Erweiterungen

Der vorgestellte Plane-Sweep-Linienstileffekt ermöglicht es, ein Linienstilattribut in einer Richtung (und damit eindimensional) zu beeinflussen. Erweiterungsmöglichkeiten des Effektes bestehen in der Implementierung einer höherdimensionalen Beeinflussung des Effektes. Bei einer derartigen höherdimensionalen Beeinflussung ist jedoch zu beachten, daß bei einer d-dimensionalen Beeinflussung die zu definierende Funktion immer die Dimension $(d + 1)$ hat. So kann durch zwei linear unabhängige Funktionen eine zweidimensionale Beeinflussung erreicht werden. Die resultierende Funktion, die durch die zwei linear unabhängigen zweidimensionalen Funktionen definiert ist, ist jedoch dreidimensional. Für eine effiziente Spezifikation der beeinflussenden Funktion ist es notwendig, daß sich der Nutzer diese vorstellt, indem er sie aus den zwei zweidimensionalen kombiniert. Dies erscheint schwierig. Eine dreidimensionale Beeinflussung durch drei linear unabhängige Funktionen scheint aber nicht mehr praktikabel zu sein, da sich ein Nutzer hierfür eine vierdimensionale Funktion vorstellen müßte.

Einfacher wäre die Nutzung einer 3D-Textur, um eine entsprechende Beeinflussung der Attribute zu erreichen. Eine Schwierigkeit ergibt sich allerdings an sich auch bei einer 3D-Textur: durch die zeitliche Komponente (lokales Keyframing) wird auch hier eine weitere Dimension in den Effekt eingebracht, so daß sich ein Nutzer auch in diesem Falle eine vierdimensionale Funktion vorstellen müßte. Dies ist demzufolge ebenfalls nicht praktikabel.

4.3 Volumeneffekt

Wie eben angedeutet, ist die Beeinflussung der Linienstilattribute in nur einer Dimension eine starke Einschränkung. Es ist z. B. vorstellbar, daß man in der zu erstellenden Graphik einen begrenzten Teil eines Objektes oder einer Objektgruppe hervorheben möchte, um die Aufmerksamkeit darauf zu lenken. Um eine solche lokale Wirkung zu erzielen, ist eine andere Herangehensweise als die im vorigen Abschnitt vorgeschlagene erforderlich. Es ist ein Effekt notwendig, dessen Wirkungsbereich in allen drei Raumdimensionen begrenzt ist. Daher wird im folgenden ein Effekt vorgestellt, dessen Einflußbereich durch eine Kugel beschrieben wird.

4.3.1 Definition des Effektes

Die Kugel ist spezifiziert durch die Position ihres Mittelpunktes M (repräsentiert durch den Vektor $\overrightarrow{m}$) im Raum und ihren Radius r (zur Illustration siehe Abbildung 4.3). Des weiteren wird wieder eine Funktion $f(t)$ definiert. Diese ist jedoch in diesem Falle entlang des Radius der Kugel wirksam. Um den Attributwert eines Kurvenpunktes P

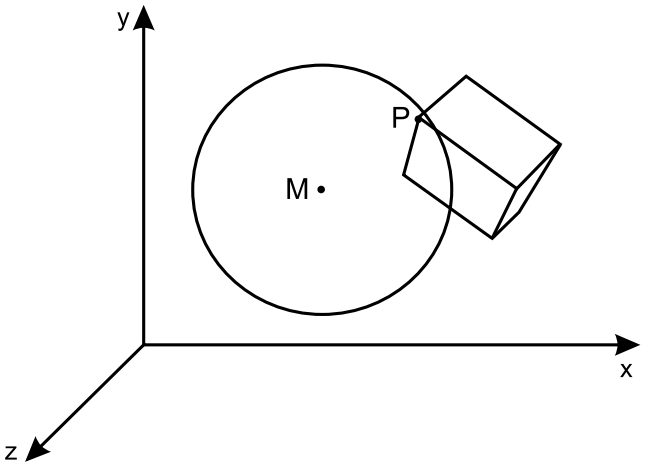


Abbildung 4.3: Skizze des Volumen-Effektes. Eine Kugel mit dem Mittelpunkt M definiert den Einflußbereich des Volumeneffektes. Ein Punkt P wird in Abhängigkeit von seinem Abstand zu M beeinflußt.

zu bestimmen, wird der Abstand des Punktes vom Mittelpunkt der Kugel bestimmt und in Relation zum Radius gesetzt:

$$t = \frac{\| \overrightarrow{p} - \overrightarrow{m} \|}{r} \tag{4.2}$$

Die Berechnung des Attributwertes eines Kurvenpunktes beim Volumeneffekt ist in konstanter Zeit realisierbar, da die Berechnung der Entfernung von zwei Punkten nur eine Komplexität von $O(1)$ hat. Die Berechnung der Attributwerte eines Objektes bzw. einer Objektgruppe hat daher eine Komplexität von $O(m)$. Eine Vorbereitung der Berechnung ist in diesem Fall nicht notwendig, da keine Transformationsmatrix für die eigentliche Attributwertberechnung bestimmt werden muß.

4.3.2 Dynamische Position

Diese Spezifikation ermöglicht die Definition eines relativ statischen Effektes zur Veränderung des Attributwertes eines Linienstilattributes in einem kugelförmigen Gebiet. Die Position der Kugel kann durch das in Abschnitt 3.4 beschriebene lokale Keyframing animiert werden. Um damit gute Ergebnisse erzielen zu können, muß jedoch nicht nur eine nicht-lineare Form der Interpolation zwischen den Keyframes implementiert werden. Es müssen auch viele Keyframes definiert werden, die mit den eigentlichen Parametern für die Beeinflussung der Linienstilattribute wenig zu tun haben.

Besser für die Bestimmung der Position der Kugel erscheint daher die Nutzung von im Modell bereits vorhandenen Pfaden, wie sie beispielsweise durch Kamerafahrten oder animierte Lichtquellen vorhanden sind. Die beiden genannten Modellelemente eignen sich neben im Modell definierten Hilfsobjekten (Dummy-Objekten) besonders gut, da sie nicht-visuelle Objekte sind und damit im Ausgabebild nicht erscheinen.

4.3.3 Erweiterungen

Eine mögliche Erweiterung des Effektes ist die Implementierung weiterer Körper, innerhalb derer der Effekt wirksam ist. Die Kugel stellt die einfachste der Möglichkeiten dar. Denkbar sind aber auch z. B. Ellipsoide und Quader als Effektkörper. Um alle Freiheiten bei der Definition der Effekte zu erhalten, könnte man verschiedene Funktionen entlang der Achsen der Bezugskörper definieren. Dies wird jedoch schnell unübersichtlich. Will man das vermeiden und nur eine Funktion zur Beeinflussung des Effektes definieren, gibt es verschiedene Varianten, entlang welcher Richtung bzw. Achse diese definiert sein soll. Im folgenden sollen exemplarisch zwei Wirkungsweisen diskutiert werden.

Eine Möglichkeit ist der Bezug auf die durch den Mittelpunkt M des Effektkörpers und den betrachteten Punkt P definierte Gerade. Die Entfernung zwischen dem Mittelpunkt M und dem Schnittpunkt der Geraden mit dem Effektkörper sei als *aktueller Radius* r_P bezeichnet. Die Berechnung von t erfolgt dann wie folgt:

$$t = \frac{\|\overrightarrow{p} - \overrightarrow{m}\|}{r_P} \tag{4.3}$$

Eine weitere Möglichkeit besteht in der Berechnung des Parameters t als aktueller minimaler Abstand des Punktes P zur Bezugskörperoberfläche d_P im Verhältnis zum maximal möglichen minimalen Abstand d_{max}:

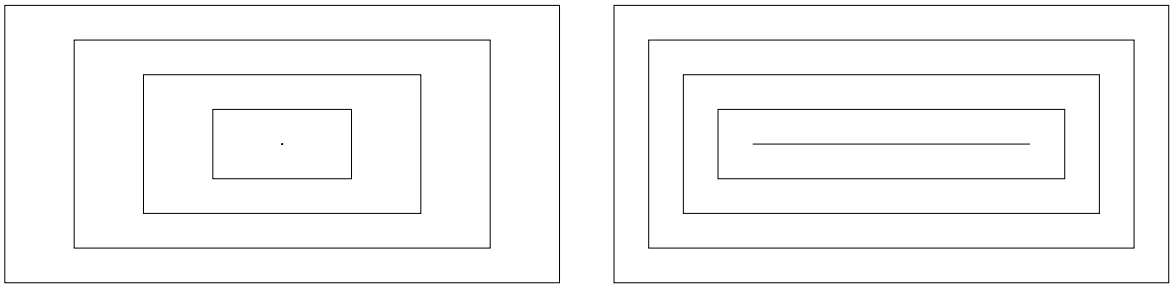
$$t = \frac{d_P}{d_{max}} \tag{4.4}$$

Der Unterschied der Wirkungsweisen beider Funktionen ist in Abbildung 4.4 schematisch veranschaulicht. Die Graphik zeigt einen Schnitt durch einen Quader, in dem jeweils für fünf Werte von t die dadurch beschriebenen Schnittrechtecke dargestellt sind. Abbildung 4.4(a) zeigt die Wirkungsweise der Funktion $f(t)$ nach Gleichung 4.3, während Abbildung 4.4(b) der Wirkungsweise der Funktion nach Gleichung 4.4 entspricht.

Neben diesen angesprochenen bestehen noch weitere Möglichkeiten, die Wirkungsweise der Funktion $f(t)$ festzulegen. Zusätzliche Varianten aufzuzählen würde aber den Rahmen dieser Arbeit sprengen.

4.4 Kamera-Effekt

Eine Möglichkeit der Beeinflussung von Attributwerten ist die abhängig von einer im Modell definierten Kamera. Alle die Punkte, die innerhalb der Sichtpyramide der Kamera liegen, werden vom Effekt beeinflußt.



(a) Bezug auf den aktuellen Radius

(b) Bezug auf den maximal möglichen minimalen Abstand

Abbildung 4.4: Mögliche Wirkungsweisen der Funktion $f(t)$, schematisch dargestellt am Beispiel eines Schnittes durch einen Quader. Die von außen nach innen angeordneten Rechtecke kennzeichnen gleiche Werte für t.

4.4.1 Definition des Effektes

Zu jeder Kamera kann ein sogenannter *Sichtstrahl* angegeben werden, der durch die Kameraposition (S) und den Kamerazielpunkt (T) definiert ist (vgl. Abbildung 4.5). Ähnlich wie der beim Plane-Sweep-Effekt in Abschnitt 4.2.1 definierte Effektstrahl wird der Sichtstrahl der Kamera dazu verwendet, um eine Transformation zu berechnen, die den Strahl geeignet auf die x-Achse projiziert. Diese Transformation setzt sich aus drei Einzeltransformationen zusammen:

1. Translation der Kameraposition in den Koordinatenursprung,
2. Skalierung, um eine Normierung des Abstandes zwischen Kameraposition und Kamerazielpunkt auf 1 zu erreichen und
3. die Rotation des Kamerazielpunktes in $(1, 0, 0)$, wobei die „Up“-Richtung der Kamera in die z-Achse rotiert wird.

Mittels des Öffnungswinkels der Kamera und eines einstellbaren Seitenverhältnisses kann das Sichtfenster der Kamera an der Stelle $x = 1$ bestimmt werden, wobei die y- bzw. z-Richtung im Weltkoordinatensystem der x- bzw. y-Richtung im Bildkoordinatensystem entspricht.

Ein beliebiger Punkt im Weltkoordinatensystem kann auf seine Lage innerhalb des Kamerabildes durch Anwendung der oben beschriebenen Transformation und anschließender Normierung auf $x = 1$ getestet werden.

Eine auch bei diesem Effekt zu definierende Funktion $f(t)$ kann entsprechend der in Abschnitt 4.3.3 beschriebenen Gleichungen 4.3 oder 4.4 angewendet werden. Es erscheint plausibel, t als minimalen Abstand vom Bildrand in Relation zum maximal möglichen minimalen Abstand zu berechnen (Gleichung 4.4), da dies eine gleiche Beeinflussung des Attributes bei gleichem minimalen Abstand zum Bildrand bedeutet. Diese Art der

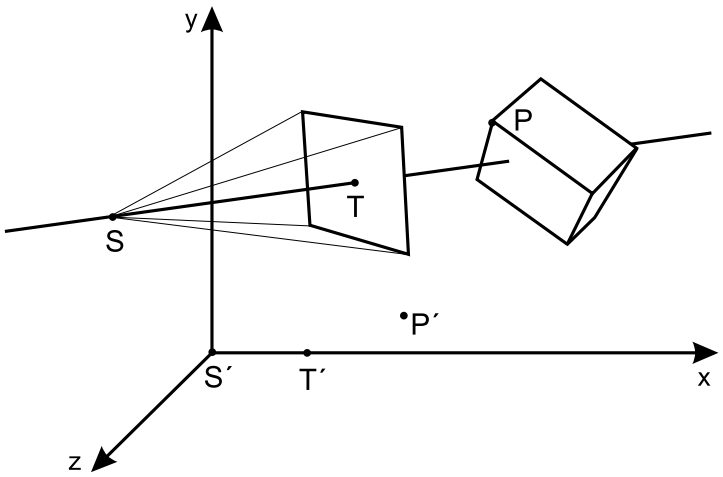


Abbildung 4.5: Skizze des Kamera-Effektes. Ein durch den Augenpunkt S, den Zielpunkt T, einen Öffnungswinkel und ein Seitenverhältnis beschriebene Kamera bildet die Grundlage des Effektes. Ein Punkt P wird entsprechend seiner relativen Lage im Sichtfeld der Kamera beeinflußt.

Anwendung der Funktion ähnelt Verfahren in der Bildbearbeitung, bei denen Ränder um Bilder gelegt werden.

Die Vorbereitung der Berechnung in Form der Bestimmung der Transformationsmatrix wird in konstanter Zeit erledigt. Die eigentliche Berechnung erfolgt analog zum Plane-Sweep-Effekt durch Matrixmultiplikation, die für einen einzelnen Kurvenpunkt ebenfalls in konstanter Zeit erfolgt. Die Berechnung der Attributbeeinflussung für alle m Kurvenpunkte hat damit die Komplexität $O(m)$.

4.4.2 Erweiterungen

Eine mit erheblich höherem Rechenaufwand realisierbare Erweiterung des Kamera-Effektes ist die Beachtung von Verdeckungen. In der oben beschriebenen Variante werden alle Punkte beeinflußt, die in der Sichtbarkeitspyramide der Kamera liegen, unabhängig davon ob sie von ihr aus sichtbar sind oder nicht. Denkbar ist, dies auf die von der Kamera aus sichtbaren Punkte zu beschränken. Diese Berechnung wäre allerdings nicht mehr in $O(1)$ pro betrachtetem Punkt machbar und wird daher auch nicht weiter betrachtet.

4.5 Beleuchtungseffekt

Ähnlich dem eben beschriebenen Kamera-Effekt ist auch ein Beleuchtungseffekt vorstellbar. Dieser soll die Teile des Objektes beeinflussen, die von einer Lichtquelle beschienen werden. Die Definition stützt sich dabei auf das Beleuchtungsmodell nach Phong und insbesondere auf den Anteil der diffusen Reflexion. Des weiteren orientiert sich die Definition des Effektes an der Definition von Lichtquellen in 3D Studio MAX R2.5.

4.5.1 Definition des Effektes

Im Beleuchtungsmodell nach PHONG wird die Intensität des diffus reflektierten Lichtes als

$$I_d = I_i k_d (\overrightarrow{l} \cdot \overrightarrow{n}) \tag{4.5}$$

definiert [Wat93], wobei I_d die Intensität des reflektierten Lichtes, I_i die Intensität der Lichtquelle, k_d eine Reflexionskonstante der reflektierenden Oberfläche, $\overrightarrow{l}$ der Beleuchtungsvektor vom betrachteten Punkt zur Lichtquelle und $\overrightarrow{n}$ der Normalenvektor an diesem Punkt ist (vgl. Abbildung 4.6). Der Term $\overrightarrow{l} \cdot \overrightarrow{n}$ entspricht dabei dem Cosinus des Winkels zwischen den beiden Vektoren und ist im Beleuchtungsmodell nach PHONG nur im Bereich von 0° bis 90° definiert. Da im vorliegenden Fall aber auch das Verhalten der Attribute an vom Licht abgewandten Kanten von Objekten beschrieben werden soll, wird hier der Bereich von 0° bis 180° betrachtet. Außerdem soll die Beeinflussung der Attribute nicht wie beim photorealistischen Rendering durch den Lichtquellen zugeordnete Faktoren, sondern ausschließlich durch den Effekt selbst geschehen, so daß I_i und k_d als konstant 1 angenommen werden.

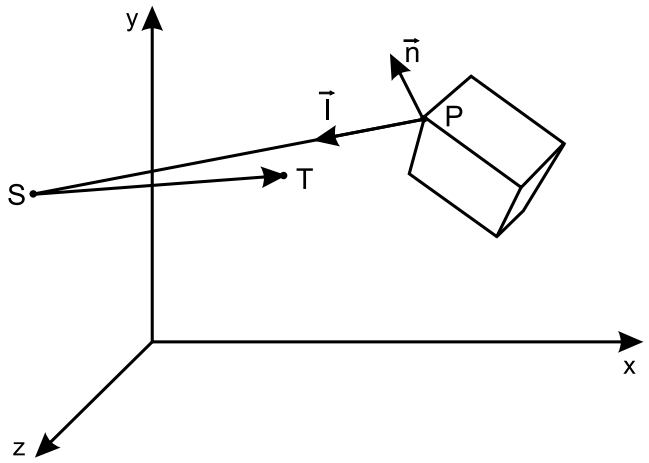


Abbildung 4.6: Skizze des Beleuchtungseffektes. Die Lichtquelle ist durch die Punkte S und T sowie einen Öffnungswinkel gegeben. Für Punkt P wird entsprechend dem Beleuchtungsmodell nach PHONG unter Einbeziehung des Beleuchtungsvektors und der Normalen die Intensität der diffusen Reflexion bestimmt, anhand derer die Beeinflussung der Attributwerte erfolgt.

Analog zu den bisher vorgestellten Effekten kann auch bei diesem Effekt das Verhalten durch eine Funktion $f(t)$ beeinflußt werden, die über I_d definiert ist. Damit gilt (nach Normierung auf $[0;1]$, wobei bei $t = 0$ der betrachtete Punkt der Lichtquelle zugewandt ist)

$$t = \frac{(\overrightarrow{l} \cdot \overrightarrow{n}) + 1}{2}. \tag{4.6}$$

4.5.2 Komplexitätsbetrachtungen

Der Vektor $\overrightarrow{l}$ kann in konstanter Zeit aus den Positionen der Lichtquelle und des betrachteten Punktes bestimmt werden. Der zweite Vektor $\overrightarrow{n}$ muß aus den Normalen der Endpunkte der Kanten interpoliert werden, zu der der betrachtete Punkt gehört. Dabei ist darauf zu achten, daß ein Eckpunkt unterschiedliche Normalen haben kann, je nachdem zu welcher Kante gehörig er gerade betrachtet wird. Bestimmt er das

Ende eines Kurvenzuges und gleichzeitig den Beginn eines weiteren Kurvenzuges, so hat er in den meisten Fällen unterschiedliche Normalen. Dies ist notwendig, um scharfe Übergänge zwischen Kurvenzügen modellieren zu können, wie dies beispielsweise bei Quadern der Fall ist.

Die Berechnung der Beleuchtung eines Punktes hat somit eine konstante Komplexität, da das Skalarprodukt von zwei Vektoren in konstanter Zeit berechnet werden kann. Also ist die gesamte Berechnung auch nur linear von der Anzahl der betrachteten Punkte abhängig.

4.5.3 Erweiterungen

Alternativ zur Spezifikation einer Funktion über der Intensität kann die Funktion auch über dem Winkel zwischen den Vektoren $\overrightarrow{l}$ und $\overrightarrow{n}$ definiert sein, so daß sich eine Berechnung von t wie folgt ergibt:

$$t = 1 - \frac{\arccos(\overrightarrow{l} \cdot \overrightarrow{n})}{\pi}. \tag{4.7}$$

Neben der Veränderung des Bezuges der Funktion $f(t)$ kann auch der Öffnungswinkel von gerichteten Lichtquellen in die Berechnung mit einbezogen werden. Analog zur Spezifikation von gerichteten Lichtquellen in 3D Studio MAX R2.5 wird die Intensität einer Lichtquelle innerhalb ihres durch den Winkel α angegebenen Hauptstrahlbereichs als 1 angenommen, während sie in der Randzone (durch einen zweiten Winkel β, $\beta > \alpha$ spezifiziert) linear auf 0 abfällt. Damit können auch gerichtete Lichtquellen mit eingeschränktem Öffnungswinkel für Lichteffekte benutzt werden. So kann ein einem Spotlight auf einer Bühne ähnelnder Effekt erreicht werden.

4.6 Zusammenfassung

Nachdem eine Reihe verschiedener Effekte vorgestellt wurde, werden diese im folgenden noch einmal zusammengefaßt. Dabei wird vor allem auf die Laufzeitkomplexität der Gesamtberechnung und auf die zu jedem Effekt definierte Funktion eingegangen.

4.6.1 Komplexität

Bei den einzelnen Effekten wurde deren Laufzeitkomplexität für eine Berechnung angegeben (vgl. Zusammenfassung in Tabelle 4.1). Wichtig ist jedoch auch das Verhalten im Kontext der gesamten Berechnung und inwieweit deren Komplexität erhalten bleibt.

Bis auf den Plane-Sweep-Effekt ist bei allen Linienstileffekten die Komplexität der Vorbereitungsphase bei n Objektpunkten $O(1)$, beim Plane-Sweep-Effekt beträgt sie $O(n)$. Bei e Effekten liegt die Komplexität der Vorbereitungsphase der Effekte also

Effekt	Komplexität der Vorbereitung	Komplexität der Berechnung
Plane-Sweep-Effekt	$O(n)$	$O(m)$
Volumeneffekt	$O(1)$	$O(m)$
Kamera-Effekt	$O(1)$	$O(m)$
Beleuchtungseffekt	$O(1)$	$O(m)$

Tabelle 4.1: Komplexität der Effekte

bei $O(e \cdot n)$. Im Normalfall kann jedoch von einer beschränkten und damit konstanten Anzahl von Effekten ausgegangen werden, da zu viele angewendete Effekte im allgemeinen dem Ziel einer klaren Visualisierung entgegen wirken. Da bei der Darstellung der Effekte die Komplexität des Verfahrens bereits bei $O(n)$ (für die Bestimmung der Ausgabepolygone muß auf jeden Kontrollpunkt zugegriffen werden) liegt, wird die Gesamtkomplexität des Algorithmus im Normalfall durch den Prozeß der Vorbereitung der Berechnung aller Effekte somit nicht erhöht.

Die Komplexität zur Berechnung der Attributbeeinflussungen an allen m Kurvenpunkten liegt bei allen Effekten bei $O(m)$. Bei a Attributen und e Effekten liegt die Gesamtkomplexität somit bei $O(a \cdot e \cdot m)$. Nun kann von einer konstanten Anzahl von Attributen ausgegangen werden, da im vorliegenden Fall nur zwei betrachtet werden, nämlich die Attribute Linienstärke und Liniensättigung. Wird die Anzahl der Effekte, wie oben angesprochen, wieder als konstant angenommen, ist die Gesamtkomplexität der Attributwertbestimmung an den Kurvenpunkten $O(m)$. Somit wird die Gesamtkomplexität des Verfahrens nicht erhöht, da bereits auf jeden Kurvenpunkt bei dessen Erzeugung einmal zugegriffen wird.

4.6.2 Effektfunktionen

Wie bereits in Abschnitt 4.1 erwähnt, ist allen hier vorgestellten Effekten je eine Funktion zugeordnet, die dem Nutzer die Steuerung der Attributbeeinflussung der Effekte erlaubt. Die unterschiedlichen Richtungen, entlang derer die Funktionen bei den verschiedenen Effekten wirken, sind in Tabelle 4.2 zusammengefaßt.

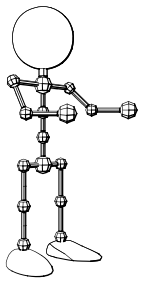
Die Berechnung der von den Effekten verursachten Attributbeeinflussung erfolgt jeweils im dreidimensionalen Raum. Die Richtungen, entlang derer die jeweilige Funktion angewendet wird, beschreiben jedoch einen Unterraum, der nicht notwendigerweise dreidimensional ist. Die Dimension dieses Unterraumes ist ebenfalls in Tabelle 4.2 enthalten. So wirkt die Funktion beim Plane-Sweep-Effekt nur eindimensional, nämlich entlang der Richtung des Effektstrahles. Im Gegensatz dazu wird die Funktion beim Volumeneffekt dreidimensional angewendet, da sie über dem Radius einer Kugel definiert ist. Kamera-Effekt und Beleuchtungseffekt erscheinen zunächst sehr ähnlich, abgesehen vom Unterschied in der Form des Volumens, innerhalb dessen die Beeinflussung erfolgt (pyramiden- bzw. kegelförmig). Da die Funktion beim Kamera-Effekt aber nur parallel zur Bildebene der betrachteten Kamera wirksam ist, hat sie eine zweidimensionale

Effekt	Wirkungsrichtung der Funktion $f(t)$	Dimension
Plane-Sweep-Effekt	• über dem aktiven Segment, d. h. nutzerdefinierte Richtung in der Szene	1
Volumeneffekt	• über dem Radius der Kugel	3
Kamera-Effekt	• über einer nutzerdefinierten Richtung parallel zur Bildebene der benutzen Kamera	2
Beleuchtungseffekt	• über dem Cosinus des Beleuchtungswinkels • über dem Beleuchtungswinkel	3

Tabelle 4.2: Wirkungsweise der Funktion bei den einzelnen Effekten

Beeinflussung. Die Funktion des Beleuchtungseffektes beeinflußt die Attribute dagegen wiederum dreidimensional, da jeweils die Richtung zur Lichtquelle und die Normale in die Berechnung einbezogen werden.

Zusammenfassend kann gesagt werden, daß die hier vorgestellten Linienstileffekte exemplarisch die Möglichkeiten der Attributbeeinflussung unterhalb der Objektebene illustrieren. Es sind weitere Effekte denkbar, die ebenfalls im Rahmen der Effekthierarchie mit lokalem Keyframing realisierbar sind. Die hier vorgestellten Effekte wurden ausgewählt, da sie besonders naheliegende und einsichtige Wirkungen erzielen.



Kapitel 5

Ein Linienstileffekt-Animationssystem

In diesem Kapitel wird die Implementierung der in Kapitel 3 entwickelten Effekthierarchie mit lokalem Keyframing und der in Kapitel 4 vorgestellten Linienstileffekte beschrieben. Bei der Implementierung kam die objektorientierte Programmiersprache Smalltalk (VisualWorks 2.5) zum Einsatz, da sie sich besonders gut für die prototypische Implementierung von Algorithmen eignet. Des weiteren ist ein am Institut für Simulation und Graphik bereits vorhandener und in dieser Arbeit verwendeter analytischer Linienrenderer in dieser Umgebung implementiert.

Bevor jedoch näher auf Implementierungsdetails eingegangen wird, werden zunächst einige Anforderungen an ein Linienstileffekt-Animationssystem näher spezifiziert. Darauf folgt eine Beschreibung der Implementierung der Rahmenapplikation und danach der Linienstileffekte. Anschließend werden einige Überlegungen zur Interaktivität des Programmes angeführt. Nach Betrachtungen zum Generieren von Bildserien werden zum Abschluß des Kapitels einige spezielle Anforderungen an Modelle erläutert.

5.1 Anforderungen an das System

Im folgenden werden einige Anforderungen an das zu schaffende Linienstileffekt-Animationssystem formuliert, die sich aus dem entwickelten Konzept sowie der zu erwartenden Art des Einsatzes des Programms ergeben.

In Abschnitt 2.5 wurde erläutert, daß ein dreidimensionales Geometriemodell eine wichtige Voraussetzung für die computerunterstützte Erzeugung von Illustrationen ist. Daher sollte das zu schaffende System zunächst in der Lage sein, diese Geometriemodelle in einem üblichen Dateiformat einzulesen. Die Implementierung eines zusätzlichen Modelliermoduls ist nicht notwendig, da bereits sehr gute Modellierer existieren, die alle benötigten Daten bereitstellen und außerdem den Zugriff auf bestehende Modellbibliotheken erlauben.

Für die Erzeugung von Vektorgraphiken muß ein analytisches Rendering durchgeführt werden (vgl. ebenfalls Abschnitt 2.5), das rechentechnisch sehr aufwendig ist. Daher ist keine direkt-manipulative Interaktion mit der Liniengraphik-Animation möglich. Aus diesem Grund sollte das System eine in einem photorealistischen Stil gerenderte Vorschau bereitstellen, die den photorealistischen Anteil der Animation zeigt. Mit

dieser ist es auch möglich, die Sicht auf die Szene benutzerdefiniert zu verändern und so den Blick auf die Szene dem Visualisierungsziel anzupassen.

Die zu erstellenden Liniengraphik-Animationen sollen sowohl innerhalb des Systems betrachtbar als auch in einem gängigen Animationsformat exportierbar sein, um auch extern als Illustration verwendet werden zu können. Neben dem Abspeichern als Animation sollen Einzelbilder auch getrennt abspeicherbar sein, sowohl in einem Pixelformat als auch als Vektorgraphik. Letzteres ist für die Reproduktion der Graphik in Printmedien von Bedeutung (vgl. Abschnitt 2.1).

Das zu schaffende System soll das in Kapitel 3 entwickelte Konzept einer Effekthierarchie mit lokalem Keyframing unterstützen, um so die computerunterstützte Erstellung von Illustrationen mittels der in Kapitel 4 besprochenen Linienstileffekte zu ermöglichen.

Die Objekte der im Modell gespeicherten Szene sollen mit dem System zu einer Hierarchie angeordnet werden können. Es soll möglich sein, weitere Informationen wie eine Effekthierarchie mit lokalem Keyframing anzulegen. Das System soll die Erstellung dieser Informationen mit einer geeigneten Navigations- bzw. Interaktionsschnittstelle unterstützen. Außerdem sollen diese Informationen abspeicherbar sein, um für weitere Veränderungen zur Verfügung zu stehen.

Aufgrund der hohen Laufzeit von insbesondere analytischem, aber auch konventionellem (Scanlinien-) Rendering soll das System Verfahren zur Beschleunigung der Abläufe implementieren, wie beispielsweise Caching[1].

Basierend auf diesen Anforderungen wurde ein Linienstileffekt-Animationssystem implementiert, das im folgenden beschrieben wird. Dabei wird zunächst auf die Rahmenapplikation eingegangen und anschließend die Implementierung der einzelnen Effekte näher betrachtet.

5.2 Rahmenapplikation

Die Rahmenapplikation implementiert die sowohl für das Keyframing als auch für die Animation mittels Effekthierarchie notwendigen Hilfsmittel. Diese beinhalten

- den Aufbau der Modelldatenstrukturen aus der abgespeicherten Szenenbeschreibung,
- die Anzeige der Szene im OpenGL-Fenster sowie die Animation der OpenGL-Ansicht,
- die notwendigen Steuerungen zur Interaktion mit der Animation und dem Linienrendering,

[1] Zwischenspeichern von Informationen, die wiederverwendet werden können

- das Speichern von fertigen Liniengraphiken und Linienanimationen sowie der Zusatzinformationen (Objekthierarchie sowie Keyframe- bzw. Effekthierarchiedaten) und
- einige weitere Elemente.

Als Format für die Szenenbeschreibungsdateien wurde auf das 3DS-Format[2] zurückgegriffen, da dies ein von den üblichen Modellierwerkzeugen exportierbares Format ist. Außerdem steht ein Import-Modul für 3DS-Dateien bereits in VisualWorks zur Verfügung, das auch die Modelldatenstrukturen für den in dieser Arbeit verwendeten Linienrenderer erzeugt. Damit ist sowohl die Kompatibilität mit gängigen Modellierwerkzeugen als auch die einfache Integration in die vorhandene Software gesichert.

Zur photorealistischen Visualisierung der Animation wurde auf eine bereits vorhandene OpenGL-Implementierung für VisualWorks zurückgegriffen, die für die Animation von Objekten angepaßt werden mußte. Um die Animation möglichst schnell ablaufen lassen zu können, wurden die Objekte und ihre Positionen nicht für jedes Einzelbild neu berechnet, sondern nur die entsprechenden Transformationsmatrizen ausgehend von der Anfangssituation bestimmt. Die eigentliche Transformation wird dann von OpenGL soweit möglich hardwarebeschleunigt durchgeführt, so daß eine flüssige Animation zustande kommt. Bei der Berechnung der Transformationsmatrizen mußte darauf geachtet werden, daß das Koordinatensystem der 3DS-Dateien um $-90°$ um die x-Achse gegenüber dem rechtshändigen Standardkoordinatensystem gedreht ist. Die OpenGL-Ansicht bietet die aus dem OpenInventor bekannten Interaktionsmöglichkeiten, die an das veränderte Koordinatensystem angepaßt wurden. Darüber hinaus wurden Kontrollelemente zur Interaktion mit der photorealistischen Animation implementiert (vgl. Screenshot in Abbildung 5.1).

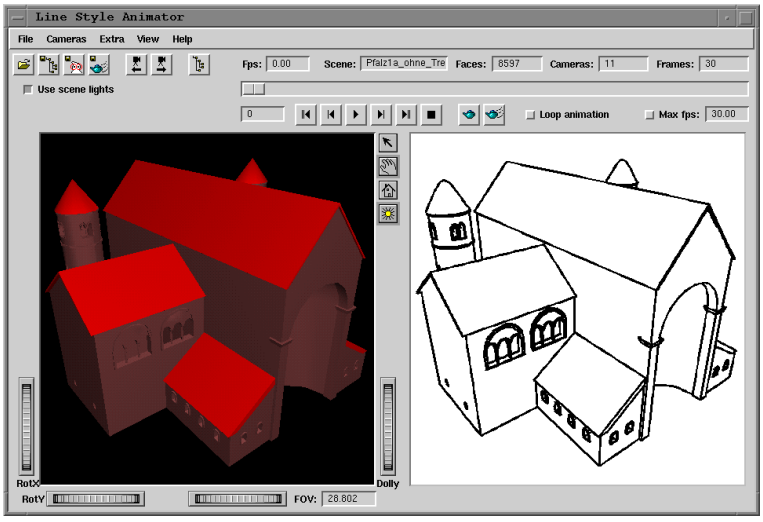


Abbildung 5.1: Rahmenapplikation. Auf der linken Seite ist die OpenGL-Darstellung der Szene und rechts die liniengraphische Ansicht erkennbar. Im oberen Teil befinden sich die Kontrollelemente der Applikation.

Um die interaktive Erstellung einer Objekt- und Linienstileffekthierarchie zu ermöglichen, wurde ein Hierarchie-Explorer entwickelt, der sich an die aus dem Betriebssystem Windows95 bekannten Explorer-Metapher anlehnt. Dieser Hierarchie-Explorer

[2] Geometriemodell-Format von 3D Studio Release 4

kann nach einer Anpassung an die jeweiligen Hierarchieklassen genutzt werden, um die Hierarchie zu bearbeiten, Knoten zu erstellen, zu löschen, in der Hierarchie zu versetzen und deren Eigenschaften zu editieren. Benutzt wurde der Hierarchie-Explorer im vorliegenden Fall für die Realisierung der Objekthierarchie-Ansicht und der Effekthierarchie-Ansicht im Effekthierarchie-Browser (vgl. Screenshot in Abbildung 5.2).

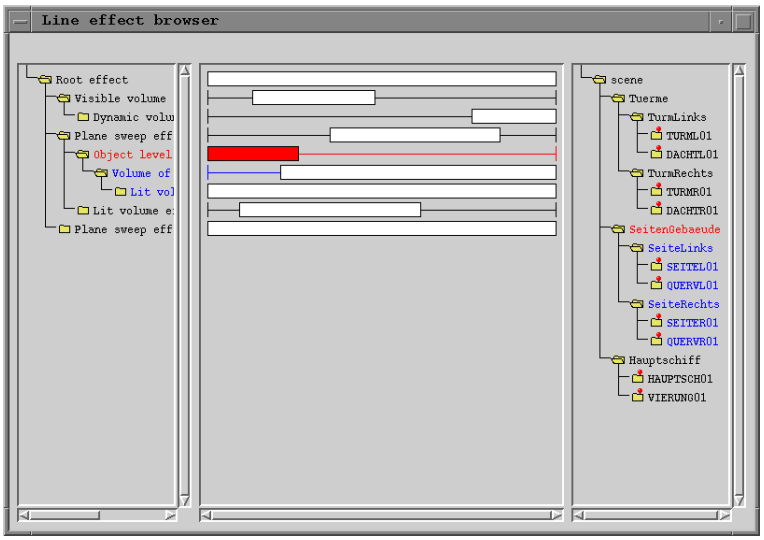


Abbildung 5.2: Effekt-Browser. Links ist die Effekthierarchie dargestellt, in der Mitte sind die zeitlichen Wirkungsbereiche der einzelnen Effekte auf deren zugehörigen Zeitstrahlen visualisiert und auf der rechten Seite ist die Objekthierarchie der Szene angeordnet.

Die Ergebnisse des Renderings können sowohl als Einzelbilder in einem Pixelformat (GIF) oder einem Vektorformat (Encapsulated PostScript) als auch als komplette Animation im AnimatedGIF-Format abgespeichert werden. Für die Speicherung in einem Pixelformat wurde ein sogenanntes Oversampling[3] implementiert, um eine weichgezeichnete Darstellung der Graphik zu ermöglichen. Allerdings ist der Rechenaufwand des Oversamplings aufgrund der Implementierung in der Sprache Smalltalk sehr hoch. Daher erscheint eine nachträgliche Generierung der Pixelgraphiken aus den Vektorgraphiken (beispielsweise mit den PPM-Tools[4]) effektiver.

Die während der Arbeit mit dem Programm getätigten Einstellungen, wie Linienstileffekt-Einstellungen und die Objekthierarchie, können in einer Hierarchiedatei zusätzlich zur Modelldatei abgespeichert werden, damit sie für Änderungen zur Verfügung stehen. Die Trennung von Modell- und Hierarchiedatei ist sinnvoll, da so die Modelldatei unabhängig vom vorgestellten Programm auch noch in weiteren Applikationen genutzt werden kann. Das Format der Hierarchiedatei entspricht zusätzlich noch dem am Institut für Simulation und Graphik verwendeten Hierarchieformat, so daß mit dem Linienstileffekt-Animationssystem erstellte Hierarchiedateien ohne Probleme übernommen werden können, wobei zumindest die Objekthierarchie weiterverwendet werden kann.

Die Linienzeichenroutine des vorhandenen Systems mußte so erweitert werden, daß 3D-Koordinaten bis nach der Transformation in Bildschirmkoordinaten erhalten bleiben,

[3] Ausgabe der Graphik auf einem n-fach größeren Bild und anschließendes Verkleinern auf $\frac{1}{n}$, um so einen Weichzeichnungseffekt zu erreichen.

[4] Gruppe von Programmen zur Bildbearbeitung und Konvertierung zwischen verschiedenen Dateiformaten

damit Rückschlüsse auf die Geometrie möglich sind. Dazu werden die Bildschirmkoordinaten wieder in Kamerakoordinaten und anschließend in Weltkoordinaten umgewandelt, um die Berechnungen der Effekte in Welt- bzw. Kamerakoordinaten durchführen zu können. Ein einfaches Speichern der Koordinaten ist nicht ausreichend, da die Kurvenpunkte erst in Bildschirmkoordinaten erzeugt werden, also vorher nicht in Kamera- oder Weltkoordinaten vorhanden sind. Neben der Interpolation der x- und y-Koordinaten wird auch die z-Koordinate interpoliert, damit eine inverse Transformation möglich ist. Neben der inversen Transformation ist auch noch die Interpolation von Normalen an den Kurvenpunkten notwendig, um die Berechnungen des Beleuchtungseffektes zu ermöglichen.

5.3 Effekthierarchie und Effekte

Für die Realisierung der Berechnung der in der Effekthierarchie spezifizierten Linienstileffekte ist es notwendig, zur Laufzeit bestimmte Programmabschnitte kombinieren zu können. Die Implementierung baut daher im wesentlichen auf dem Konstrukt des *Blocks* in Smalltalk auf, der dieses Konzept unterstützt. Ein Block ist ein Objekt, das neben einem Stück Programmcode auch einen Kontext enthält. Der Kontext speichert dabei u. a. die Belegung von Variablen und andere Statusinformationen ab. Ein Block kann durch das Senden der Nachricht `value` ausgewertet werden und liefert ein Ergebnis zurück. Neben dieser einfachen Methode gibt es auch Nachrichten, die eventuell für die Auswertung notwendige, zusätzliche Parameter übergeben.

5.3.1 Effekthierarchie

Die für die Anwendung der Linienstileffekte notwendigen Berechnungen werden als Block an die Zeichenroutine übergeben. Dem Block werden als Parameter der zu beeinflussende Kurvenpunkt, die zugehörige z-Koordinate und die für den Beleuchtungseffekt notwendige Normale an dem Punkt übergeben. In einem ersten Schritt werden die Koordinaten in Welt- bzw. Kamerakoordinaten umgewandelt und anschließend die Berechnungen der Effekthierarchie durchgeführt (vgl. Abbildung 5.3).

Der Programmcode für diese Berechnungen liegt auch in Form von Blöcken vor und wurde vor dem Beginn der Berechnungen aus einzelnen Blöcken entsprechend der Hierarchie zusammengestellt. Jeder dieser einzelnen, den Hierarchieknoten zugeordneten Blöcke berechnet den von den Einstellungen des Effektes spezifizierten Attributwert, was der zweiten Phase der Attributwertbestimmung entspricht (vgl. Abschnitt 4.2.3). Wird der betrachtete Kurvenpunkt nicht beeinflußt, wird der Wert `nil` als Ergebnis zurückgegeben.

Außerdem muß darauf geachtet werden, daß Zwischenergebnisse soweit wie möglich bereits außerhalb des Blockes berechnet werden, damit die Berechnung nicht bei jedem Aufruf des Blockes erneut durchgeführt werden muß. Daher erfolgt die erste Phase,

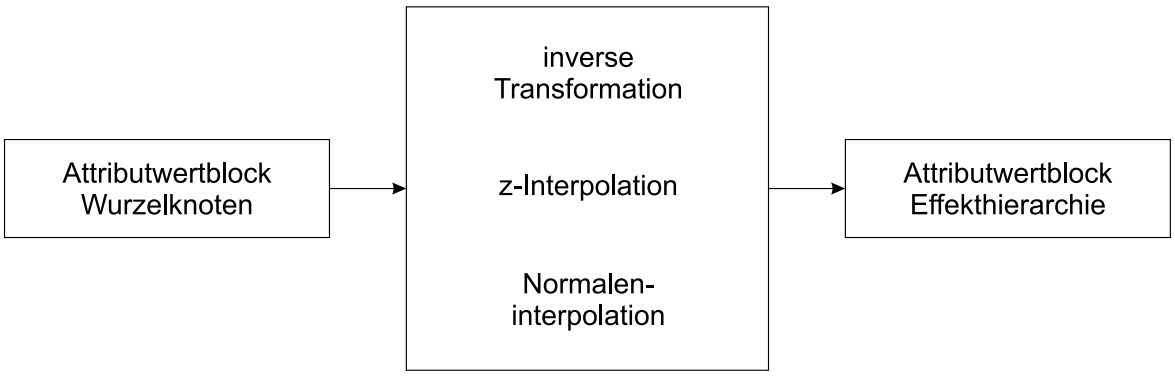


Abbildung 5.3: Generierung des an die Zeichenroutine zu übergebenden Blockes zur Beeinflussung der Attributwerte.

in der die Berechnung vorbereitet wird, bereits bei der Zusammenstellung der Blöcke. Die in dieser Phase ermittelten Ergebnisse, wie beispielsweise Transformationsmatrizen, sind im Kontext des Blockes gespeichert, der die Attributwertberechnung durchführt. Sie stehen somit bei jeder Auswertung des Blockes zur Verfügung und müssen nicht erneut berechnet werden.

Das Zusammenfassen der Ergebnisse der einzelnen Linienstileffekte ist durch verschiedene Funktionen möglich. Die in den Kindknoten eines Effektes ermittelten Attributwerte werden mittels einer solchen zu spezifizierenden Funktion mit dem Attributwert des betrachteten Effektes kombiniert (vgl. Abbildung 5.4). Dabei werden Ergebnisse mit dem Wert `nil` ignoriert, so daß nur Punkte mit einer tatsächlichen Beeinflussung durch einen Effekt Einfluß auf das Gesamtergebnis haben. Implementiert wurden die Funktionen *Maximum*, *Minimum*, *Addition*, *Multiplikation* und *Durchschnitt*. Diese Liste kann aber auch einfach um weitere Funktionen ergänzt werden.

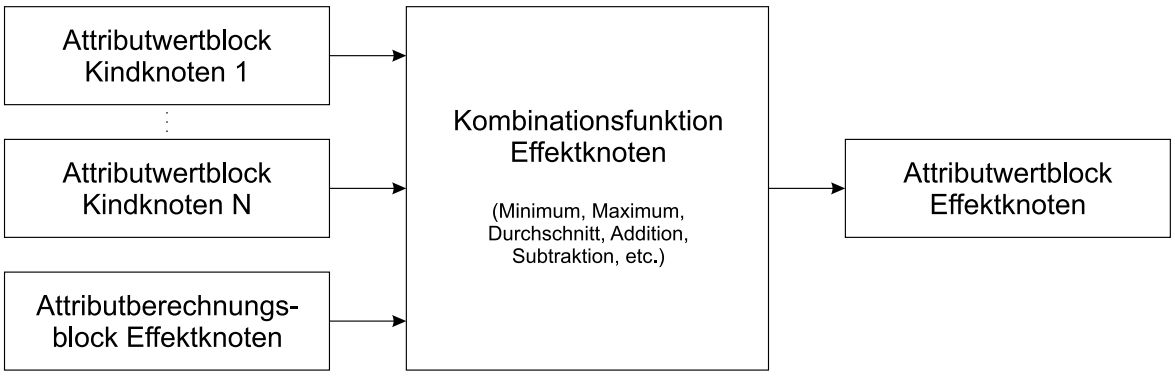


Abbildung 5.4: Zusammenfassen der Attributwerte der Kindknoten mit dem eigenen Attributwert eines Linienstileffektes im Rahmen der Effekthierarchie. Der Attributberechnungsblock berechnet nur den Einfluß des eigentlichen Effektes, wohingegen der Attributwertblock die Beeinflussung der Teilhierarchie eines Effektes bestimmt.

Bei den meisten implementierten Effekten ist die Beeinflussung eines Parameters durch eine beliebig gestaltbare Funktion möglich. Um dem Benutzer die Eingabe dieser Funktion zu erleichtern, wurde eine Klasse `InterpolationObject` implementiert, die

die graphische Spezifikation dieser Funktion mittels Kontrollpunkten ermöglicht (vgl. beispielsweise Abbildung 5.5 auf Seite 58). Dazu wird eine eindimensionale Spline-Interpolation genutzt. Diese hat die Eigenschaft, daß für einen beliebigen x-Wert innerhalb des Definitionsbereiches nur genau ein y-Wert existiert. Damit kann sie zur Spezifikation einer Funktion genutzt werden.

Diese so angegebene Funktion kann mit in das Keyframing einbezogen werden, d. h., an verschiedenen Keyframes können unterschiedliche Funktionen definiert werden. Bei der Bestimmung des Funktionswertes zu einem Zeitpunkt werden zunächst die Werte zu den Zeitpunkten der beiden benachbarten Keyframes bestimmt und anschließend linear zwischen den beiden Werten interpoliert. Dies hat den Vorteil, daß die Anzahl der Stützpunkte der Kurven von verschiedenen Keyframes nicht gleich sein muß, da nicht die Kurve selbst interpoliert wird, sondern nur der Funktionswert.

5.3.2 Gemeinsame Parameter

Alle Effekte haben einige Einstellmöglichkeiten gemein (vgl. Abbildungen 5.5 bis 5.9). Dazu zählen die Grenzen ihres zeitlichen Wirkungsbereiches, angegeben durch Start- und Endframe. Weiterhin ist das zu beeinflussende Attribut wählbar, wobei mit einem zusätzlichen Faktor der Wert auch auf Werte oberhalb von 1 skalierbar ist, was insbesondere beim Attribut „Linienstärke“ von Bedeutung ist. Die Funktion zum Zusammenfassen des ermittelten Attributwertes mit den Attributwerten der Kindknoten des Effektes kann aus einer Liste gewählt werden. Diese Liste wird jeweils aus den aktuell implementierten Funktionen zusammengestellt. Damit ist eine einfache Erweiterbarkeit gewährleistet. Eine weitere Möglichkeit der Beeinflussung des Effektverhalten ist die Angabe einer zeitlichen Einflußfunktion. Diese bewirkt eine Skalierung des Attributwertes in Abhängigkeit von der innerhalb des Effektes verstrichenen Zeit. Mit dieser Funktion sind beispielsweise Ein- und Ausblendeffekte realisierbar.

In allen Fällen ist auch die Steuerung des lokalen Keyframings gleich. Wird die Option „Keyframing“ gewählt, werden die aktuellen Einstellungen dupliziert und in zwei Keyframes am Anfang bzw. am Ende des Effektes umgewandelt. Beim Einfügen eines neuen Keyframes werden ebenfalls die Einstellungen des aktuell ausgewählten dupliziert. Wird das lokale Keyframing wieder deaktiviert, werden alle Keyframes bis auf den aktuell ausgewählten gelöscht und der verbleibende als statische Einstellung verwendet. Intern wird vom Programm immer mit Keyframes gearbeitet, nur daß im Falle von nur einem Keyframe dieser als gleichzeitig am Anfang und am Ende befindlich angesehen und somit ein statisches Verhalten erreicht wird.

Für die Interpolation zwischen zwei Keyframes werden zunächst die Attributwerte an diesen beiden Keyframes bestimmt. Anschließend wird zwischen diesen beiden Werten linear interpoliert. Bei den beeinflußten Linienstilattributen sind die Veränderungen jedoch so gering, daß sie im Gegensatz zu Bewegungen vom Betrachter als kontinuierlich wahrgenommen werden und ein unstetiger Übergang nicht auffällt. Daher ist eine lineare Interpolation in diesem Falle ausreichend.

Neben diesen Einstellmöglichkeiten wird auch in jedem Fall eine kurze Beschreibung des Effektes angezeigt, die die Wirkungsweise erläutert.

In den folgenden Abschnitten werden Implementierungsdetails zu den einzelnen umgesetzten Effekten besprochen.

5.3.3 Plane-Sweep-Effekt

Das Fenster zur Einstellung der Parameter des Plane-Sweep-Effektes enthält neben den eben genannten Elementen Möglichkeiten zur Spezifikation des Bezugskoordinatensystems (Welt- bzw. Kamerakoordinatensystem), der Wirkungsweise (global bzw. lokal), der Art der Berechnung des umschließenden Objektes (in bezug auf die betrachteten Punkte bzw. auf zwei einzugebende Koordinaten), des zu verwendenden umschließenden Objektes (konvexe Hülle bzw. umschließende Kugel) sowie der zu verwendenden Funktion (vgl. Abbildung 5.5). Eine Erweiterung der über dem aktiven Segment definierten Funktion ist die Möglichkeit der Verschiebung der Grenzen des Definitionsbereiches von 0 bzw. 1 auf andere Werte. Damit kann der Definitionsbereich schmaler oder weiter als das eigentliche aktive Segment werden.

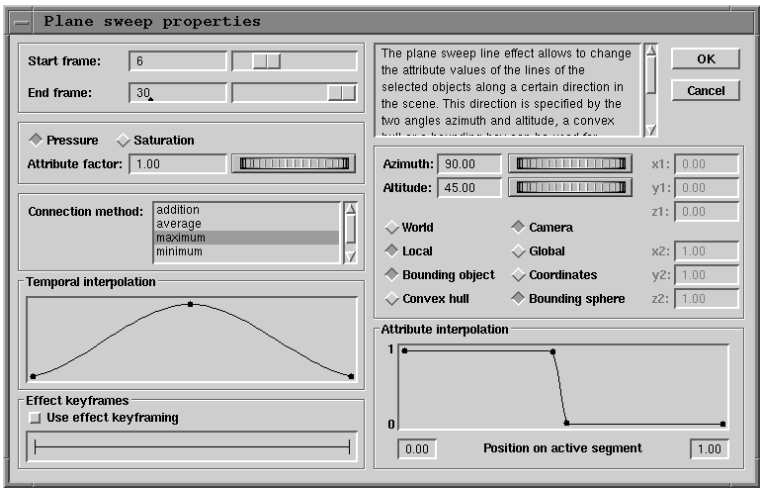


Abbildung 5.5: Dialog zum Plane-Sweep-Effekt.

Neben der Berechnung der Grenzen des aktiven Segmentes ist es beim Plane-Sweep-Effekt auch notwendig, einen Mittelpunkt der betrachteten Punktmenge nach der Transformation zu berechnen. Der Mittelpunkt wird aus dem Mittelpunkt des umschließenden Quaders der Punktmenge nach der Transformation bestimmt. Der so berechnete Punkt wird für die Positionierung eines zusätzlichen Objektes verwendet, das die Visualisierung des aktiven Segmentes ermöglicht (vgl. Abschnitt 5.4.1).

5.3.4 Volumeneffekt

Beim Volumeneffekt ist neben der zu verwendenden Funktion im wesentlichen der Radius und die Position der verwendeten Kugel einstellbar (vgl. Abbildung 5.6). Darüber

hinaus sind zwei automatische Einstellmöglichkeiten implementiert worden. Die erste zentriert die Kugel im Mittelpunkt des umschließenden Quaders der betrachteten Objekte. Dies ist insbesondere bei Objekten von Vorteil, die sich weit vom Koordinatenursprung entfernt befinden. Ohne diese Funktion wäre es mangels direkter Manipulationsmöglichkeit in der OpenGL-Ansicht schwierig, die Kugel zu den betrachteten Objekten zu bewegen. Umgekehrt kann die Kugel mit der zweiten automatischen Einstellung wieder in den Koordinatenursprung verschoben werden.

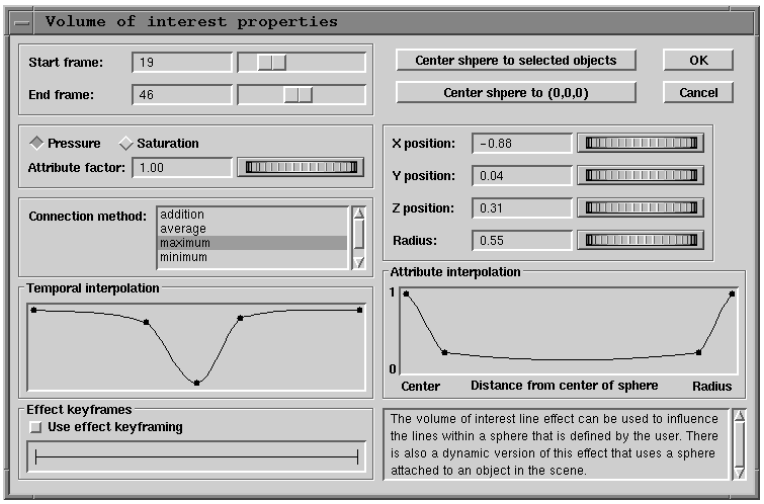


Abbildung 5.6: Dialog zum Volumeneffekt.

Abbildung 5.7 zeigt das Fenster zur Einstellung des dynamischen Volumeneffektes. Im Gegensatz zur eben beschriebenen statischen Variante kann hier nur der Radius der Kugel verändert werden. Die Position ergibt sich aus der Position eines dem Effekt zugeordneten Objektes. Dabei können sowohl „normale“ Objekte wie auch Kameras oder Lichter als Referenzobjekt benutzt werden. Es bietet sich jedoch an, in der Szene ein virtuelles Objekt (z. B. eine Kamera oder eine Lichtquelle) zu definieren, da dieses im Ausgabebild nicht auftaucht.

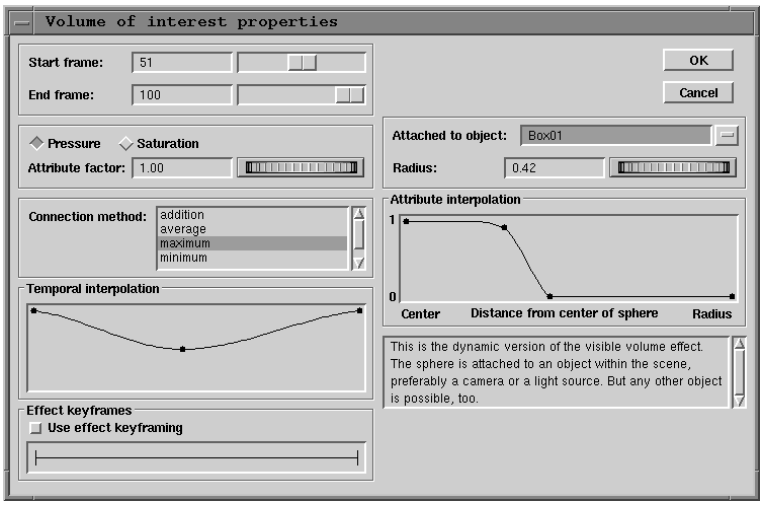


Abbildung 5.7: Dialog zum dynamischen Volumeneffekt.

5.3.5 Kamera-Effekt

Bei Kamera-Effekten muß jeweils neben der anzuwendenden Funktion eine Referenzkamera gewählt werden, deren Sichtfeld hervorgehoben werden soll (vgl. Abbildung 5.8). Des weiteren muß das Seitenverhältnis der Kamera eingestellt werden, da dieser Wert nach dem Laden der Modelldatei nicht in der Datenstruktur zur Verfügung steht.

Wie in Abschnitt 4.4.1 erläutert wurde, kann die einzustellende Funktion auf verschiedene Weisen angewendet werden. Implementiert wurde die Berechnung des Funktionsparameters t nach Gleichungen 4.3 und 4.4, also als die aktuelle Entfernung vom Mittelpunkt im Verhältnis zum aktuellen Radius und als der aktuelle minimale Abstand vom Rand des Kamerafensters im Verhältnis zum maximal möglichen minimalen Abstand.

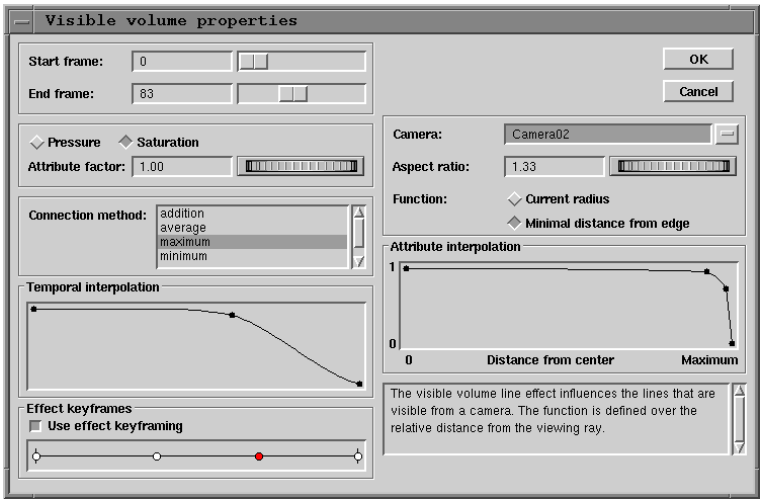


Abbildung 5.8: Dialog zum Kamera-Effekt.

5.3.6 Beleuchtungseffekt

Analog zum Kamera-Effekt kann beim Beleuchtungseffekt neben der zu verwendenden Funktion die Lichtquelle gewählt werden, deren Verhalten mit dem Effekt simuliert werden soll. Des weiteren kann, wie in Abschnitt 4.5 beschrieben, die Funktion über dem Winkel zwischen Beleuchtungsvektor und Normalenvektor oder über dem Cosinus dieses Winkels angewendet werden. Beide Varianten wurden implementiert und sind im Dialog wählbar (vgl. Abbildung 5.9).

5.3.7 Weitere Effekte

Neben den oben beschriebenen Effekten wurden noch weitere Effekte implementiert. Dazu gehört ein einfacher Kombinationseffekt, der selbst keine Attributbeeinflussung vornimmt und nur die Ergebnisse seiner Teilhierarchie entsprechend einer Funktion zusammenfaßt. Um auch herkömmliche Effekte zu ermöglichen, wurde ein Effekt implementiert, der die Linienstilattribute objektgebunden beeinflußt. Ein weiterer Effekt

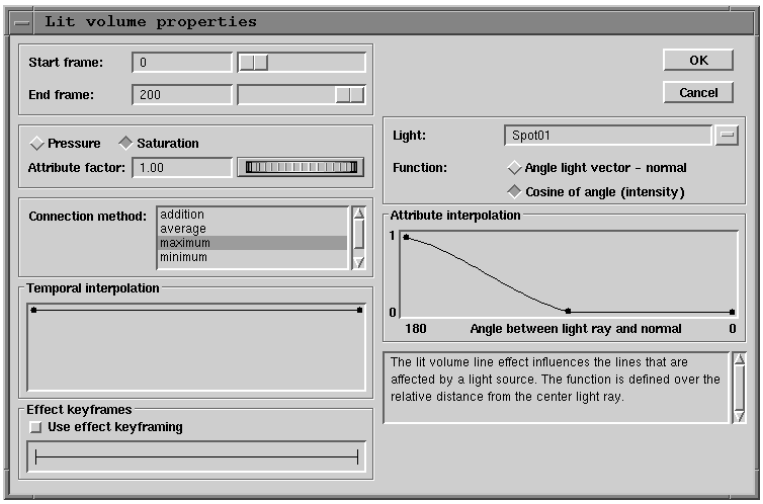


Abbildung 5.9: Dialog zum Beleuchtungseffekt.

implementiert die Zuordnung von vorher spezifizierten Linienstilen zu einzelnen Objekten, wobei durch lokales Keyframing verschiedene Stile ineinander überführt werden können. Hierbei ist aber darauf zu achten, daß die Stile zueinander kompatibel sind, d. h., daß ihre die Position und die Attribute beschreibenden Kurven u. a. auch die gleiche Anzahl von Kontrollpunkten haben müssen.

Der in jeder Hierarchie vorhandene Wurzelknoten steuert das Erscheinungsbild von Linien, die nicht durch einen Effekt beeinflußt werden. Er hat aber auch entsprechend der gewählten Kombinationsfunktion Einfluß auf die anderen Linien. Im Wurzelknoten sind neben der Linienstärke und der Liniensättigung auch ein vorab festgelegter, d. h. als Datei vorliegender Linienstil sowie die zufällige Abweichung von diesem Stil wählbar. Alle diese Werte können dabei ins lokale Keyframing mit einbezogen werden.

5.4 Betrachtungen zur Interaktion

Das Ziel des vorgestellten Programmes ist es, ein dreidimensionales Geometriemodell interaktiv mit Linienstileffekten zu versehen, um damit eine Liniengraphik oder eine Liniengraphik-Animation zu illustrativen Zwecken zu erstellen.

Das dabei angewendete analytische Rendering ist in den meisten Fällen jedoch sehr aufwendig, da zum Bestimmen der Sichtbarkeitsbeziehungen alle Kanten gegeneinander auf Überschneidungen getestet werden müssen. Dies bedeutet einen hohen rechentechnischen Aufwand, so daß selbst auf Hochleistungsrechnern der Renderingprozeß sehr lange dauert.[5] Selbst das eigentliche Anzeigen der Ausgabelinien ist aufwendig, da aus den berechneten sichtbaren Linien erst unter Anwendung der oben beschriebenen Linienstileffekte die eigentlichen Ausgabelinien erzeugt werden müssen.

[5] Ein Modell mit 10 325 Polygonen benötigte für das Rendering auf einer SGI Onyx 2 IR beispielsweise ungefähr 9 Minuten (vgl. auch Anhang C.2).

Aufgrund dieser Komplexität des Renderingprozesses ist nicht jede Aktion gleichzeitig anhand eines Ausgabebildes bzw. einer Ausgabeanimation überprüfbar. Daher müßten Methoden zur Verbesserung der Interaktivität des Programmes gefunden werden. Im folgenden werden die implementierten Methoden zur alternativen Visualisierung der Linienstileffekte sowie zur Beschleunigung des Ablaufes bestimmter Programmteile besprochen.

5.4.1 Visualisierung der Parameter

Da es, wie eben erläutert, nicht möglich ist, die Wirkung von Veränderungen an den Parametern der Linienstileffekte interaktiv anhand eines Ausgabebildes zu überprüfen, müssen andere Methoden zur Visualisierung der vorgenommenen Aktionen angewendet werden. Dazu bietet sich die OpenGL-Ansicht der Szene an, da OpenGL aufgrund der Implementierung verschiedener Algorithmen in Hardware eine Darstellung in Echtzeit ermöglicht und damit Veränderungen interaktiv dargestellt werden können.

OpenGL unterstützt jedoch nur ein pixelorientiertes Rendering. Eine vollständige Simulation eines Linienstileffektes in der OpenGL-Ansicht ist somit aufgrund der Unterschiede im Darstellungsstil zur Liniengraphik nicht möglich. Es ist jedoch realisierbar, den Einflußbereich bzw. die Einflußrichtung der Effekte darzustellen, damit der Benutzer eine visuelle Rückmeldung für seine Manipulation an den Parametern erhält. Dazu werden sogenannte *Visualisierungselemente* der OpenGL-Ansicht hinzugefügt, die spezifisch für den jeweiligen Effekt sind. Da diese Darstellung nur zur Visualisierung der Parameter bei der Definition eines Effektes verwendet wird, wird dabei jeweils nur das Visualisierungselement des aktuell bearbeiteten Effektes angezeigt. Probleme durch mehrere, gleichzeitig angezeigte Visualisierungselemente treten daher nicht auf.

Für die Visualisierung der Wirkungsweise des Plane-Sweep-Effektes wird ein dreidimensionaler Pfeil genutzt, der sowohl die Richtung des aktiven Segmentes als auch dessen Anfang und Ende darstellt (vgl. Abbildung 5.10). Die für den Effekt bereits berechneten Transformationsmatrizen werden genutzt, um ein im Segment von $(0;0;0)$ bis $(1;0;0)$ definiertes Pfeilprimitiv in die Szene zu transformieren. Durch die Nutzung einer proportionalen Skalierung wird gewährleistet, daß die Größe des Pfeils immer der Größe der betrachteten Objekte angepaßt ist.

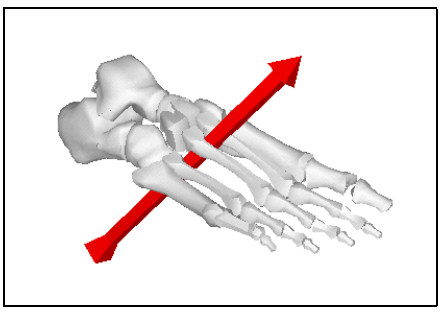


Abbildung 5.10: Visualisierungselement für den Plane-Sweep-Effekt. Der Pfeil visualisiert sowohl die Lage als auch die Länge des aktiven Segmentes.

Der Einflußbereich des Volumeneffektes wird durch die Drahtgitterdarstellung[6] der im Effekt definierten Kugel visualisiert (vgl. Abbildung 5.11). Durch die Drahtgitterdarstellung der Kugel bleiben die innerhalb der Kugel befindlichen und damit durch den Linienstileffekt beeinflußten Objekte sichtbar, soweit sie sich nicht gegenseitig verdecken. Dies ist notwendig, da andernfalls der Einflußbereich des Volumeneffektes schlecht abschätzbar wäre.

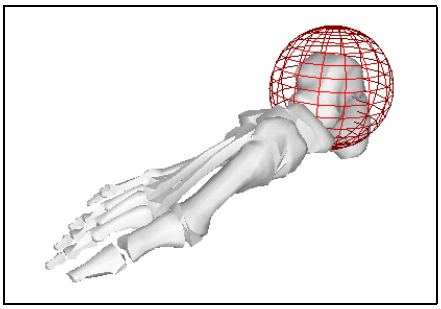


Abbildung 5.11: Visualisierungselement für den Volumeneffekt. Der Effektkörper wird durch eine Drahtgitterdarstellung visualisiert.

Die Visualisierung des Beleuchtungseffektes erfolgt durch die Einbeziehung der im Modell definierten Lichtquellen in die OpenGL-Darstellung. Dies ist besonders bei gerichteten Lichtquellen mit schmalem Lichtkegel effektiv, da diese sehr gut in der OpenGL-Ansicht erkennbar sind. Ein Nachteil besteht darin, daß alle im Modell definierten Lichtquellen gleichzeitig in die OpenGL-Darstellung einfließen. Die vom Effekt benutzte Lichtquelle ließe sich aber durch eine Drahtgitterdarstellung des Lichtkegels hervorheben, sofern es sich um eine gerichtete Lichtquelle handelt. Dabei stellt sich allerdings die Frage, wodurch die Höhe des Kegels bestimmt sein soll. Eine Möglichkeit ist das Abschneiden des Kegels an der Stelle, an der das Licht das letzte beleuchtbare Objekt verläßt. Die Berechnung dieser Größe ist aber sehr aufwendig. Eine weniger aufwendige Alternative besteht in der Nutzung der Diagonalen des umschließenden Quaders der Szene inklusive Lichtquellen, da ein Kegel mit einer solchen Höhe in jedem Fall alle möglichen Objekte einschließt. In der Implementierung wird allerdings die Entfernung von der Lichtquelle zum Lichtzielpunkt als Höhe des Kegels benutzt, da dies die einfachste Möglichkeit ist und sich auch an den Konventionen des zur Modellierung benutzten 3D Studio MAX R2.5 orientiert (vgl. Abbildung 5.12). Punktlichtquellen werden im Gegensatz dazu durch Kugeln visualisiert.

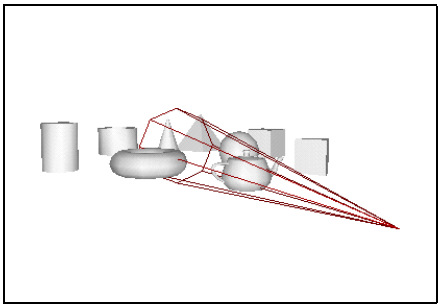


Abbildung 5.12: Visualisierungselement für den Beleuchtungseffekt. Die verwendete Lichtquelle wird durch die Drahtgitterdarstellung eines Kegels dargestellt.

Der Kamera-Effekt wird ähnlich dem Beleuchtungseffekt behandelt, nur daß statt eines Kegels eine Pyramide zur Visualisierung des Einflußbereiches des Effektes verwendet

[6] Darstellung eines Objektes durch Anzeige der Kanten des polygonalen Gitters

wird (vgl. Abbildung 5.13). Die Überlegungen zur Höhe des Kegels treffen entsprechend auch für die Pyramide zu. Die Grundfläche der Pyramide wird anhand des Öffnungswinkels und des eingestellten Seitenverhältnisses bestimmt.

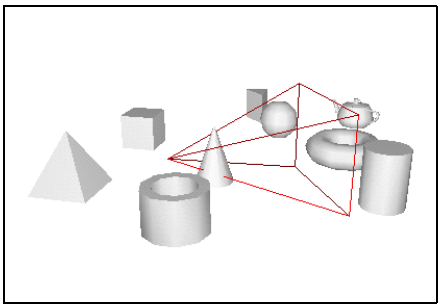


Abbildung 5.13: Visualisierungselement für den Kamera-Effekt. Die dem Effekt zugeordnete Kamera wird durch die Kanten einer Pyramide visualisiert.

5.4.2 Geschwindigkeitsoptimierung

Neben der Visualisierung der vom Benutzer verursachten Aktionen spielt die Geschwindigkeit von Programmabläufen eine große Rolle für die Benutzbarkeit von Systemen. Eine häufig angewendete Methode zur Beschleunigung von zeitaufwendigen Prozessen ist das Caching von Zwischenergebnissen, also das Zwischenspeichern wiederverwertbarer Informationen. Im konkreten Fall bietet sich Caching bei Szenen mit Objekten an, die mit Ausnahme von Lichtern in bezug auf eine Kamera nicht animiert sind, d. h., daß die eigentliche Linienstilanimation von den sich verändernden Parametern der Linienstileffekte erzeugt wird. In diesem Fall hat sich das Zwischenspeichern der berechneten sichtbaren Linien als sehr hilfreich erwiesen. Die Szene muß nur ein einziges Mal gerendert werden, für alle weiteren Frames wird auf die gespeicherten Informationen zurückgegriffen.

Des weiteren ist eine flüssige Darstellung der Animation bei Bildern mit vielen Ausgabelinien nicht möglich, da die Ausgabe der Linien aufgrund des notwendigen Rasterns der erzeugten Polygone zu lange dauert. Um den Prozeß der Ausgabe zu beschleunigen, werden sowohl die Ausgabelinien (Linien nach Anwendung der Linienstileffekte) zwischengespeichert als auch anschließend das Rastern der Ausgabelinien auf einer Bitmap durchgeführt, die im zweiten Schritt des Cachings abgespeichert wird. Anschließend ist nur noch die Anzeige der gespeicherten Bitmaps notwendig, was eine enorme Beschleunigung der Ausgabe zur Folge hat, so daß Bildwiederholraten wie bei der Animation der OpenGL-Ansicht erreicht werden.[7] Ein Flackern der Ausgabe wird durch die Anwendung von Double-Buffering unterbunden.

Die OpenGL-Ansicht des Programms dient nur zur Darstellung der Animation in einem photorealistischen Stil, d. h., sie wird nicht von Veränderungen der Linienstileffekte beeinflußt. Da die Modellierung der Szene vor der Benutzung des Linienstil-Animationssystems abgeschlossen wurde, verändert sich der photorealistische Teil der

[7] Darstellung von bis zu 70 Bildern pro Sekunde auf der benutzten SGI Onyx 2 IR im Gegensatz zu ungefähr 1,3 Bildern pro Sekunde ohne Anwendung von Caching bei einem sehr einfachen Modell mit 12 Polygonen

Animation durch die Benutzung des Programms nicht mehr. Daher ist es sinnvoll, die für die OpenGL-Darstellung benötigten Transformationen nicht bei jedem Abspielen der Animation neu zu berechnen, sondern ebenfalls nach einmaliger Berechnung zwischenzuspeichern. Damit lassen sich insbesondere bei komplexen Szenen deutliche Beschleunigungen erreichen (vgl. Anhang C.1).

5.5 Bildserien

Während der Arbeit mit dem System tauchte die Frage auf, wie Animationen geeignet auszugsweise auf Papier wiedergegeben werden können. Es wurde daher die Implementierung von Algorithmen notwendig, die die Erzeugung von Bildserien ermöglichen, welche die wesentlichen Informationen der spezifizierten Animationen enthalten. Die einfachste Möglichkeit zur Erzeugung einer solchen Bildserie ist die Generierung einer konstanten Anzahl von Einzelbildern, bezogen auf einzelne Effekte oder auf die gesamte Animation. Die zuletzt genannte Variante wurde bei mehreren der in Kapitel 6 gezeigten Abbildungen angewendet, wobei sich die Anzahl von 6 Bildern als geeignet herausgestellt hat. Bei einem Bezug auf einzelne Effekte werden pro Effekt eine bestimmte Anzahl von Einzelbildern erzeugt. In beiden Fällen sind die Einzelbilder gleichmäßig über den Bezugszeitraum verteilt.

Eine komplexere Möglichkeit zur Erzeugung von Bildserien, besteht in der Untersuchung der einzelnen Parameter der Effekte. Insbesondere die Einstellungen der zeitlichen Einflußfunktion der Effekte (vgl. Abschnitt 5.3.2) und des lokalen Keyframings bieten sich dazu an. Hat ein Effekt eine konstante zeitliche Einflußfunktion und verwendet kein lokales Keyframing, so ist er während seines gesamten zeitlichen Wirkungsbereiches gleichmäßig wichtig und es muß nur ein Bild erzeugt werden. Ist die Funktion jedoch nicht konstant, so kann man Einzelbilder an den lokalen Maxima der zeitlichen Einflußfunktion sowie an Beginn und Ende des Effektes generieren. Die lokalen Keyframes stellen definitionsgemäß ebenfalls besonders charakteristische Zeitpunkte dar, an denen die Animation der Linienstileffekte näher spezifiziert ist. Es bietet sich daher an, auch an diesen Zeitpunkten jeweils Einzelbilder zu generieren.

Die hier vorgestellten Methoden wurden implementiert und lassen sich separat verwenden. Sie lassen natürlich nur teilweise eine Automatisierung des Prozesses der Bildsequenzgenerierung zu, da dabei auch Aspekte wie Dramaturgie und Ästhetik eine Rolle spielen. Die implementierten Verfahren erlauben es jedoch, zumindest eine Vorauswahl für die Bildsequenzen zu treffen.

5.6 Anforderungen an Modelle

Für die computerunterstützte Erzeugung von Liniengraphiken sind einige Punkte in bezug auf die Auswahl der zu verwendenden Modelle zu beachten. Insbesondere aufgrund

des rechentechnisch aufwendigen analytischen Renderings sind Modelle mit einer geringen Anzahl von Polygonen zu bevorzugen. Besonders bei animierten Szenen, bei denen der Renderingvorgang für jedes Einzelbild zu wiederholen ist (vgl. Abschnitt 5.4.2), muß auf eine geringe Polygonanzahl geachtet werden, weil die Speicheranforderung und die notwendige Renderingzeit sonst zu hoch sind. Während der Tests des Programms hat sich eine Szenenkomplexität von ungefähr 10 000 Polygonen als noch vertretbar herausgestellt.

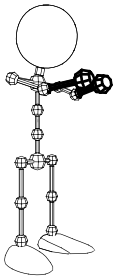
Diese Einschränkung auf einfachere Szenen als beim photorealistischen Rendering erscheint zunächst sehr negativ. Würden jedoch sehr komplexe Modelle mit feinen Strukturen für die computerunterstützte Generierung von Liniengraphiken verwendet, so kann das Resultat u. U. viele kleine Details enthalten, die das Bild überladen würden. Außerdem bildet die verwendete Linienstärke eine natürliche Grenze für die Genauigkeit, so daß feine Details einfach als schwarze Fläche dargestellt würden. Auch ist eine feine Triangulierung von Objekten oft unnötig, da nur die Kanten erzeugenden Flächen Einfluß auf das Ergebnis haben. Selbst mit einem durch die Triangulierung erzeugten Kantenzug mit vielen einzelnen Kanten kann in den meisten Fällen keine bessere Darstellung erreicht werden, da ein rund verlaufender Kantenzug (der also Kanten von Flächen der gleichen *Smoothing Group*[8] enthält) in eine Kurve umgewandelt wird, die den Kantenzug approximiert. Diese Approximation kann mit wenigeren Kanten ähnlich gut erreicht werden, bei weniger Rechenzeit und kleinerer Ausgabedatei.

Aus diesen Gründen würden sich direkt für die Erzeugung von Liniengraphiken erstellte 3D-Modelle von den in dieser Arbeit meistens verwendeten, ursprünglich für photorealistische Zwecke hergestellten Modellen unterscheiden. So würden beispielsweise für die Modellierung von Rundungen wesentlich weniger Flächen verwendet, da Rundungen durch die geeignete Zuweisung von Smoothing Groups erzeugt werden können. So können Geometriemodelle mit einem Bruchteil der Polygonflächen auskommen und trotzdem visuell identisch erscheinen.

Weitere Anforderungen an Eingabemodelle resultieren aus der in der Arbeit verwendeten Implementierung des Renderers. Da dieser Renderer Probleme beim Erzeugen von Schnittkanten sich durchdringender Flächen hat, muß darauf geachtet werden, daß solche Durchdringungen nicht auftreten. Objekte müssen daher teilweise voneinander abgerückt werden, damit die notwendigen Kanten erzeugt werden. Des weiteren erzeugt der Renderer Triangulierungskanten bei Objekten, die aus dem Blickfeld herausragen. Daher sollten Ansichten, die Objekte nicht vollständig enthalten, bei der Modellierung vermieden werden.

Unter Nutzung derartiger Modelle ist es mit dem in diesem Kapitel vorgestellten System jetzt möglich, illustrative Liniengraphiken zu erzeugen. Beispiele für so erstellte Graphiken werden nun im folgenden Kapitel präsentiert.

[8] Flächen werden im Modellierwerkzeug 3D Studio MAX R2.5 sogenannten Smoothing Groups zugeordnet, wobei Flächen einer gleichen Gruppe als abgerundete Oberfläche aufgefaßt werden.



Kapitel 6

Fallstudien

Im diesem Kapitel werden einige Beispiele vorgestellt, die die Einsatzmöglichkeiten der entwickelten Effekte und des Linienstileffekt-Animationssystems erläutern. Dabei werden die Effekte zumeist einzeln und unabhängig voneinander vorgestellt, wobei sicher auch Kombinationen von Effekten möglich sind. Neben den Beispielen werden auch Hinweise zum Einsatz der Effekte gegeben, wie sie sich während der Arbeit mit dem System herausgestellt haben.

6.1 Plane-Sweep-Effekt

Die in Abbildung 6.1 durch Einzelbilder auszugsweise wiedergegebene Animation soll den Zusammenhang zwischen den aufgrund von in einer Ausgrabung gefundenen Resten rekonstruierten Türmen und dem daraus geschlossenen zweiten Geschoß eines historischen Gebäudes visualisieren. Die hier benutzten Effekte sind der Plane-Sweep-Effekt (für die Visualisierung der oberen Hälfte der Gebäudeteile) und die Beeinflussung der Attribute auf Objektebene.

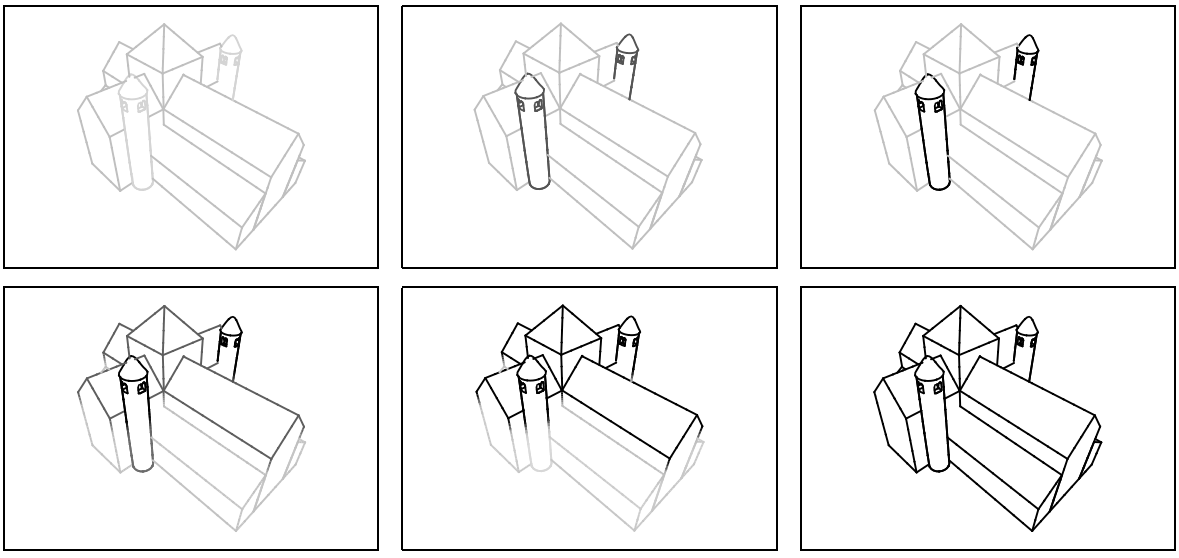


Abbildung 6.1: Sechs Einzelbilder aus einer Animation, die den Zusammenhang zwischen gefundenen Turmresten und dem zweiten Geschoß sowie der ungefähren Höhe eines historischen Gebäudes visualisiert (994 Polygone).

Zuerst werden die Türme mit voller Liniensättigung angezeigt, um ihren Charakter als Befund zu zeigen. Anschließend wird das obere Geschoß und wenig später auch der obere Teil der Türme mit dem Plane-Sweep-Effekt hervorgehoben, um den Schluß von der Existenz der Türme auf das zweite Geschoß zu visualisieren. Schließlich wird das ganze Gebäude mit voller Liniensättigung gezeigt, wodurch u. a. der Schluß von der Existenz der Türme und des Obergeschosses auf die ungefähre Höhe des Gebäudes gezeigt wird.

Die Existenz der Türme im eben genannten Beispiel wurde aus Turmresten geschlossen. Um diesen Schluß zu visualisieren, wurde das stückweise Erscheinen der Treppenstufen der Turmtreppe animiert. Diese in Abbildung 6.2 gezeigte Animation wurde ebenfalls mit Hilfe des Plane-Sweep-Effektes realisiert. Dabei wurde die benutzte Funktion durch lokales Keyframing animiert, so daß zunächst die unteren Stufen erscheinen und dann allmählich alle weiteren.

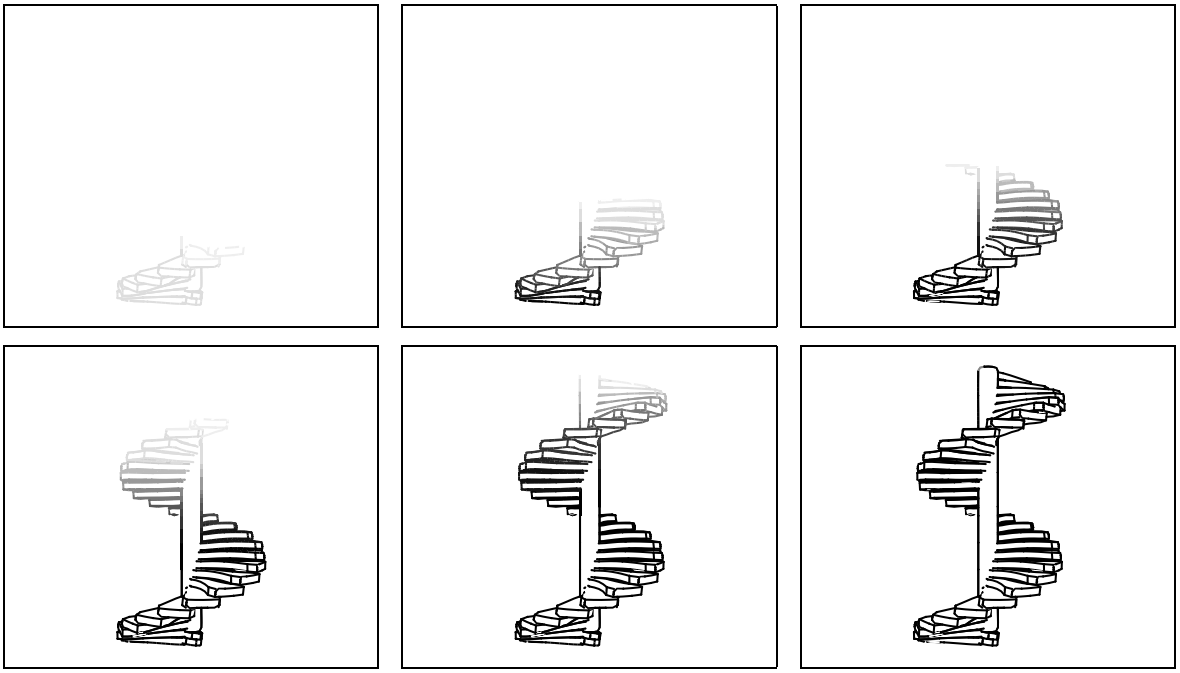


Abbildung 6.2: Visualisierung des Schlusses von Turmresten auf eine Turmtreppe: allmähliches Erscheinen der Treppenstufen (3 040 Polygone). Für die Animation wurde ein einziger Plane-Sweep-Effekt verwendet, der mit lokalem Keyframing animiert wurde.

Abbildung 6.3 zeigt die Visualisierung von logischen Eigenschaften eines Objektes, die sich nicht in der Objektstruktur des verwendeten Modells widerspiegeln (vgl. auch Beispiel 3.3 in Abschnitt 3.1). Im konkreten Fall wurde die rechte Seite eines VW-Käfers hervorgehoben. Obwohl das Modell keine Unterteilung der Objekte in rechte und linke Objekte enthält, ist mit Hilfe des Plane-Sweep-Effektes eine entsprechende Visualisierung möglich.

In der in Abbildung 6.4 auszugsweise dargestellten Animation wurde ein animierter Plane-Sweep-Effekt genutzt, um den Verlauf der Muskel am Modell eines menschli-

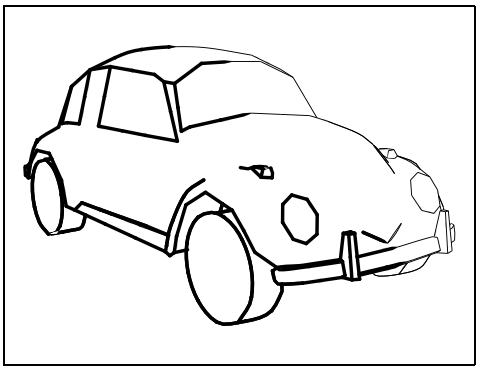


Abbildung 6.3: Hervorhebung der rechten Seite des VW-Käfers, obwohl keine entsprechende Objektunterteilung im Modell vorhanden ist (862 Polygone).

chen Fußes zu zeigen. Dazu wurde schrittweise die Linienstärke der zu den Muskeln gehörenden Linien erhöht und somit deren Verlauf visualisiert.

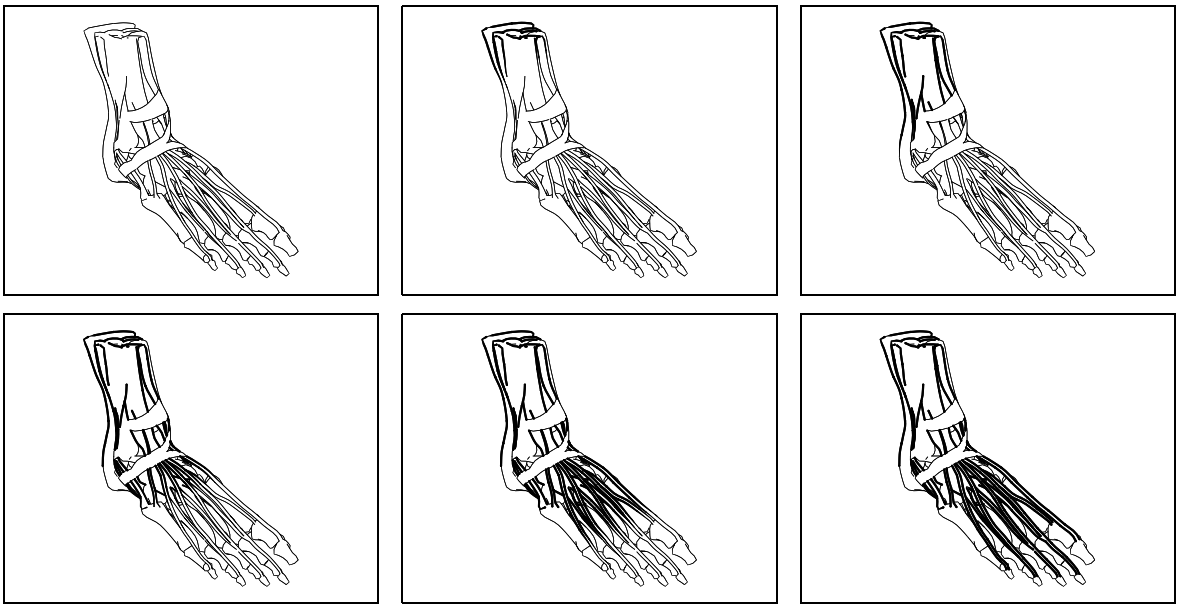


Abbildung 6.4: Visualisierung des Verlaufs der Muskeln an einem menschlichen Fuß durch schrittweise Erhöhung der Linienstärke der entsprechenden Linien (11 244 Polygone).

Mit dem Plane-Sweep-Effekt ist es auch möglich, sehr schmale Ebenen hervorzuheben. Abbildung 6.5 zeigt eine ähnliche Szene wie das in Abschnitt 1.2 angesprochene Beispiel der Visualisierung der Grundfläche von Kristallstrukturen (vgl. Abbildung 1.3 und Anhang A). In der Abbildung wurde ein Plane-Sweep-Effekt verwendet, wobei die Funktion $f(t)$ nur bei Werten, die sehr nah an 1 liegen, einen hohen Wert liefert.

6.2 Volumeneffekt

Abbildung 6.6 zeigt die Anwendung des Volumeneffektes. Im gezeigten Fall sind die Radnaben eines Fahrrades hervorgehoben, im ersten Bild unter Benutzung der Linienstärke (vgl. Abbildung 6.6(a)) und im zweiten Bild unter Benutzung der Liniensättigung

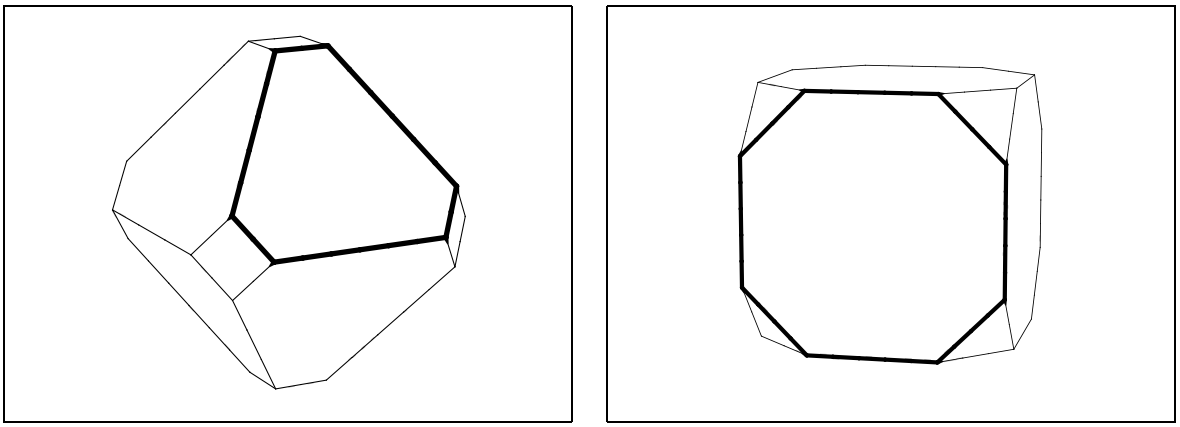
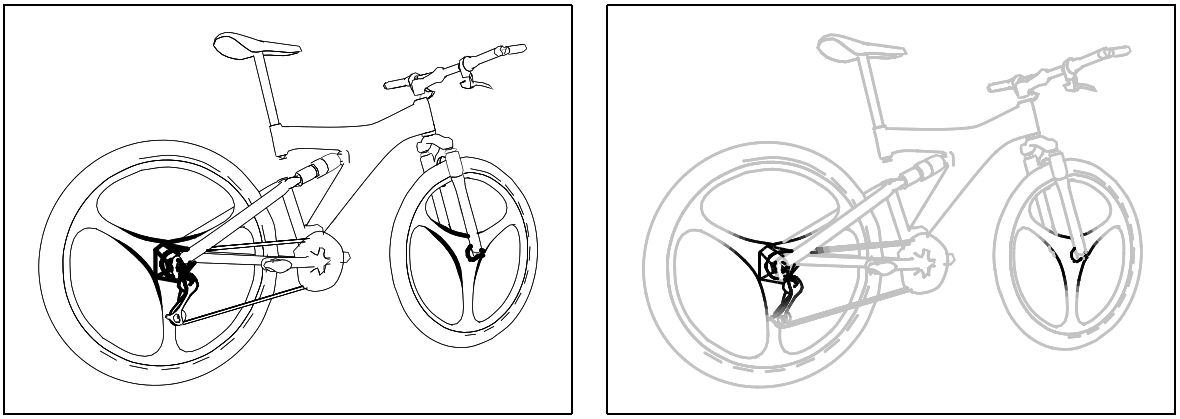


Abbildung 6.5: Visualisierung der Grundfläche von Kristallen (je 44 Polygone).

(vgl. Abbildung 6.6(b)). Hierbei ist auch die hierarchische Anwendung der Effekte gut zu erkennen. Die Objekte des Modells wurden hierarchisch gruppiert und die Effekte nur auf einen Teil der Hierarchie angewendet. So wirkt der Effekt beim Vorderrad nur auf die Teile des Rades selbst, die Radgabel befindet sich nicht in dem ausgewählten Teilbaum und wurde dementsprechend auch nicht beeinflußt.



(a) Beeinflussung der Linienstärke

(b) Beeinflussung der Liniensättigung

Abbildung 6.6: Einsatz des Volumeneffektes: Betonung der Radnaben und Zahnräder eines Fahrrades (6 073 Polygone).

An diesem Beispiel wird auch eine mögliche Anwendung des Programms deutlich: der Einsatz zur Generierung von Illustrationen in Bedienungsanleitungen. Zum einen können mit dem Programm statische Bilder erzeugt werden, wie sie in traditionellen Bedienungsanleitungen verwendet werden. Zum anderen können aber auch animierte Anleitungen generiert werden, mit denen auch komplexere Zusammenhänge vermittelt werden können.

In Abbildung 6.7 ist ein weiteres Beispiel für die Anwendung des Volumeneffektes dargestellt. In der Abbildung sind durch mehrere Effektkörper die Tempelanlagen bei den Pyramiden von Gizeh mittels einer stärkeren Liniensättigung hervorgehoben.

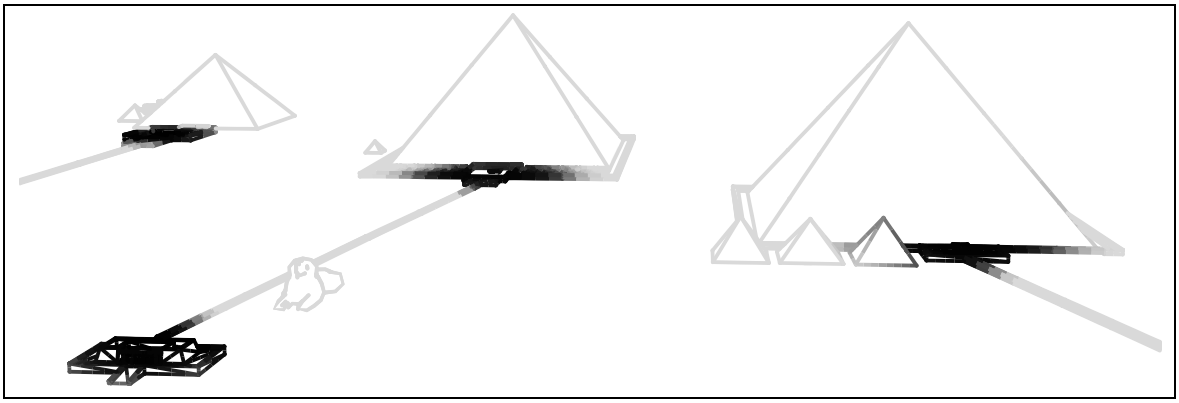


Abbildung 6.7: Hervorhebung mittels Volumeneffekt: Die Gebiete der Tempelanlagen an den Pyramiden von Gizeh in Ägypten (3 658 Polygone).

Abbildung 6.8 zeigt die Anwendung des dynamischen Volumeneffektes. Die Position einer sich durch die Szene bewegenden Kamera wird als Punkt des Interesses aufgefaßt und auf diese Position ein Volumeneffekt angewendet. Dadurch werden nacheinander die verschiedenen Objekte der Szene hervorgehoben. Der Volumeneffekt ermöglicht somit das Lenken der Aufmerksamkeit des Betrachters nacheinander auf verschiedene Objekte einer Szene.

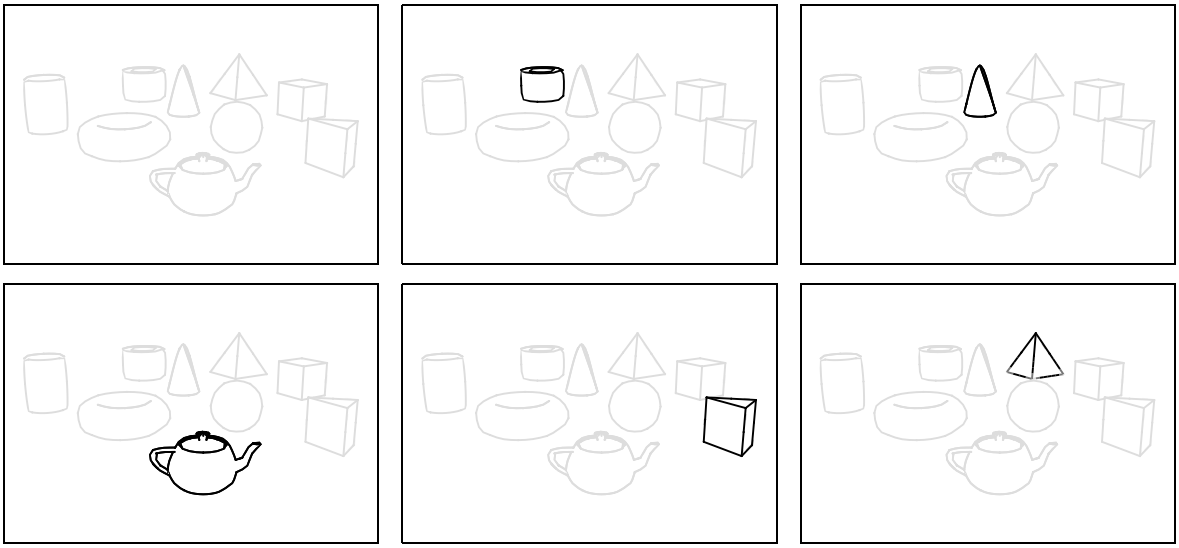


Abbildung 6.8: Visualisierung des wandernden Fokus des Interesses bzw. Leiten der Aufmerksamkeit eines Betrachters entlang verschiedener Objekte einer Szene (1 104 Polygone).

Eine weitere Anwendung findet der dynamische Volumeneffekt bei animierten Modellen, bei denen jedoch ein Teil von sich bewegenden Objekten hervorgehoben werden soll. Zu diesem Zweck können mit der Modellierungssoftware virtuelle Objekte definiert werden, wie in Abschnitt 5.3.4 angesprochen, und diese als Referenzobjekte für

den dynamischen Volumeneffekt verwendet werden. Ein solches Beispiel ist in Abbildung 6.9 zu sehen. Die Gelenke eines Knochenmodells einer menschlichen Hand wurden mit virtuellen Objekten markiert, die zusammen mit den restlichen Objekten animiert werden.

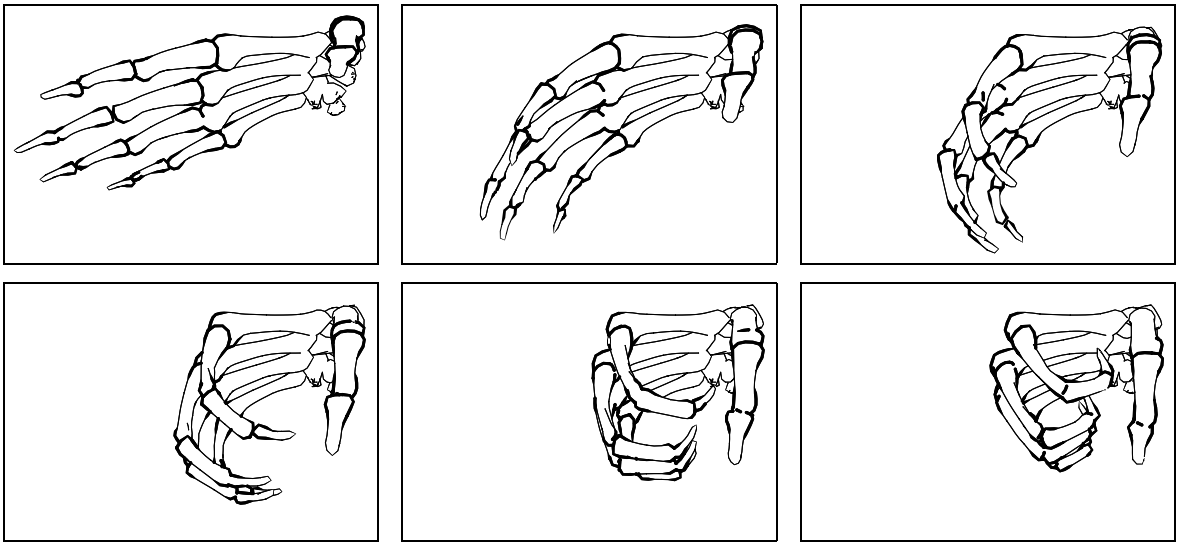


Abbildung 6.9: Hervorhebung der Gelenke am Knochenmodell einer sich schließenden menschlichen Hand (6 404 Polygone).

6.3 Kamera-Effekt

Abbildung 6.10 zeigt Einzelbilder aus einer Animation zur Visualisierung einer Kamerafahrt. Die Szene wird dabei durch eine statische Kamera betrachtet, während der Kamera-Effekt auf alle Objekte der Szene in bezug auf eine sich durch die Szene bewegende Kamera angewendet wurde. Damit können die von einer in einer Szene befindlichen Kamera aus sichtbaren Objekte visualisiert werden, was beispielsweise für die Planung von Animationen hilfreich ist. Einen ähnlichen dreidimensionalen Effekt mit photorealistischen Mitteln zu erzeugen, erscheint schwierig, da es kein Attribut gibt, das intuitiv mit einer solchen Wirkung assoziiert werden kann.

Abbildung 6.11 zeigt die gleiche Szene wie Abbildung 6.10, nur daß die Szene diesmal von einer sich um die Szene herum bewegenden Kamera aus betrachtet wird. Durch die sich dadurch ständig verändernde Sicht auf die Szene wird der dreidimensionale Charakter der Animation deutlich.

In Abbildung 6.12 wird die unterschiedliche Wirkungsweise der Darstellung von unbeeinflußten Linien gezeigt. In Abbildung 6.12(a) hat der Effekt Einfluß auf die Liniensättigung. Die unbeeinflußten Linien haben eine Liniensättigung von 0 und erscheinen daher nicht im Bild. Dadurch ist es schwierig, die Szene als eine Einheit zu erfassen, was insbesondere bei einer Animation verwirrend wirkt. Im Gegensatz dazu sind in Abbildung 6.12(b) die unbeeinflußten Linien durch Einstellungen im Wurzeleffekt mit

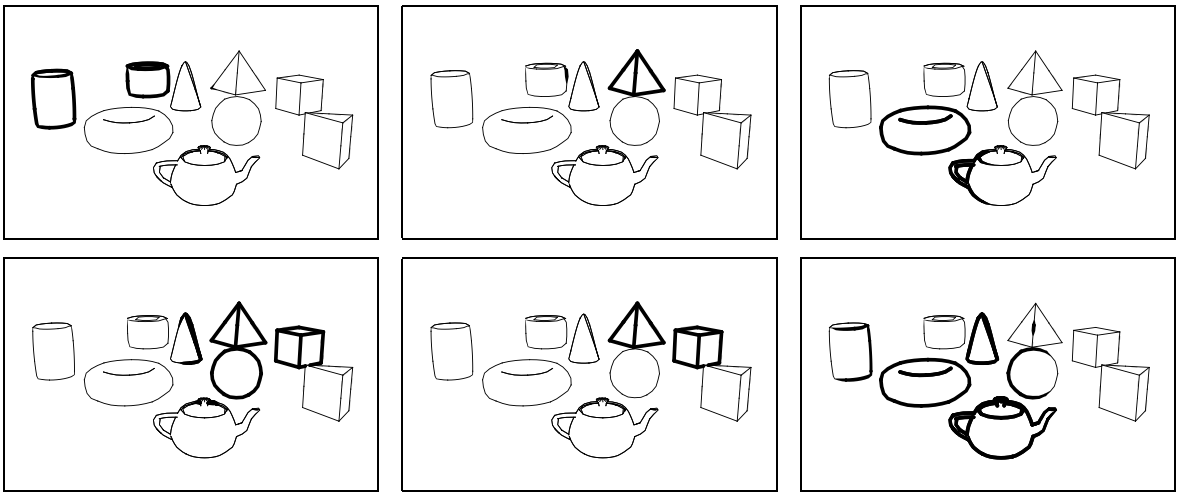


Abbildung 6.10: Visualisierung des Sichtfeldes einer sich durch die Szene bewegenden Kamera, gesehen von einer statischen Kamera (1 104 Polygone).

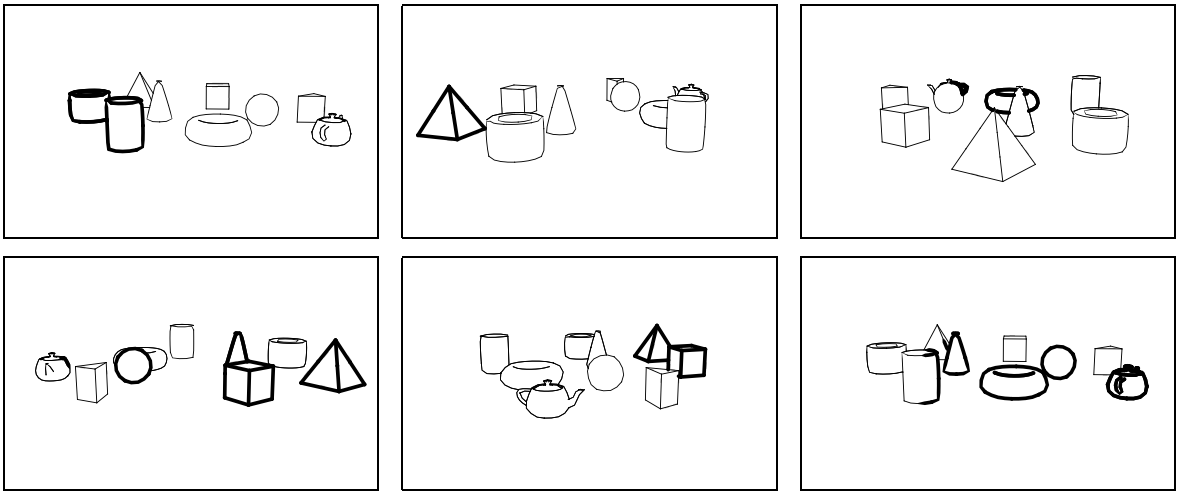
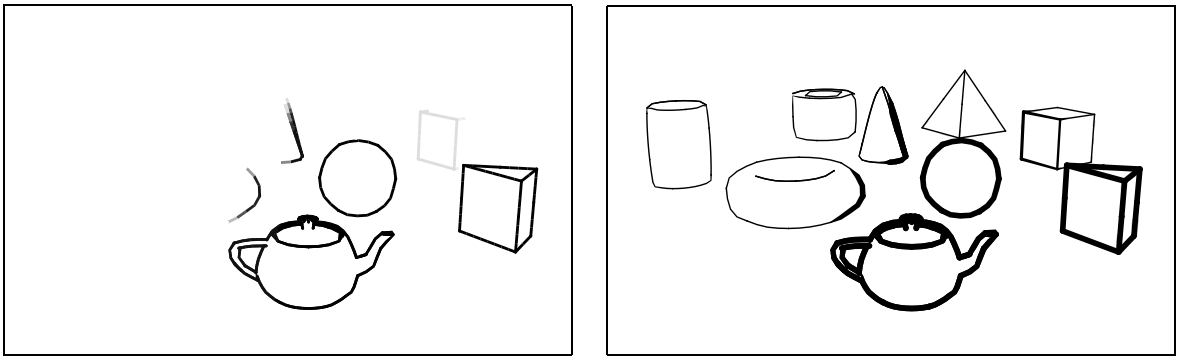


Abbildung 6.11: Visualisierung des Sichtfeldes einer sich durch die Szene bewegenden Kamera, gesehen von einer sich um die Szene herum bewegenden Kamera (1 104 Polygone).

einer geringen Linienstärke versehen und der Kamera-Effekt beeinflußt die Linienstärke. Dadurch wird der Rest der Szene sichtbar und die Auswirkungen des Kamera-Effektes sind besser erkennbar. In den meisten Fällen empfiehlt es sich also, mit Hilfe des Wurzeleffektes zumindest eine geringe Linienstärke und Liniensättigung einzustellen.

6.4 Beleuchtungseffekt

Die Anwendung des Beleuchtungseffektes ist in Abbildung 6.13 dargestellt. Ein schmaler Lichtkegel streicht über die Szene und die beleuchteten Linien werden durch eine höhere Liniensättigung hervorgehoben. Zur besseren Erkennbarkeit der Szene wurden die nicht vom Effekt beeinflußten Linien mit einer geringen Liniensättigung versehen.



(a) Unbeeinflußte Linien sind nicht dargestellt.

(b) Unbeeinflußte Linien sind mit einer geringen Linienstärke dargestellt.

Abbildung 6.12: Unterschiedliche Wirkung der gleichen Sicht auf eine Szene ohne und mit Darstellung der nicht vom Effekt beeinflußten Linien (1 104 Polygone).

Würde dies nicht der Fall sein, wäre das Resultat etwa dem Anblick eines Radarschirms bzw. der Wirkung des Lichtkegels eines Leuchtturm vergleichbar.

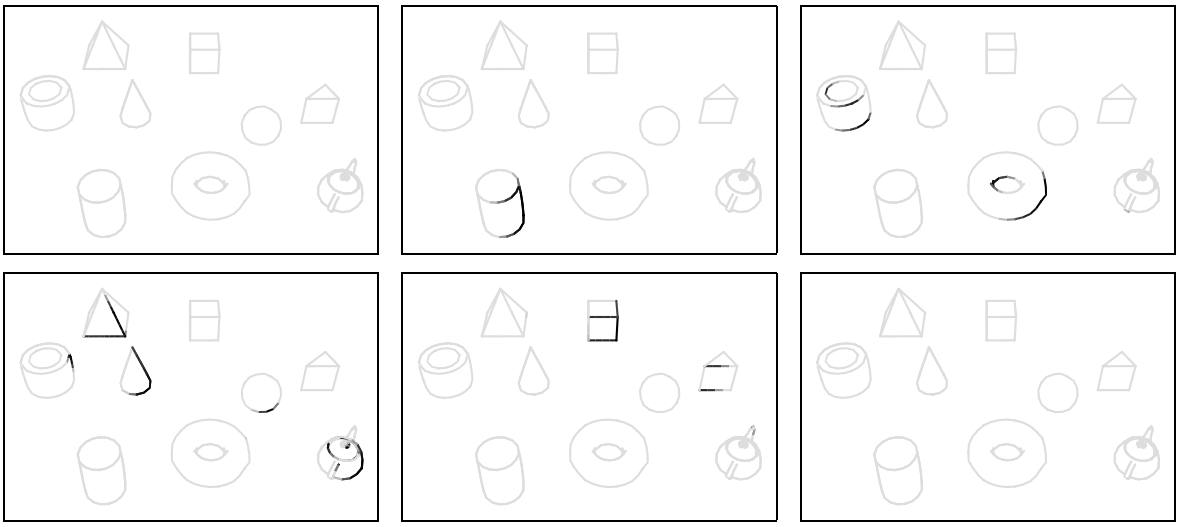
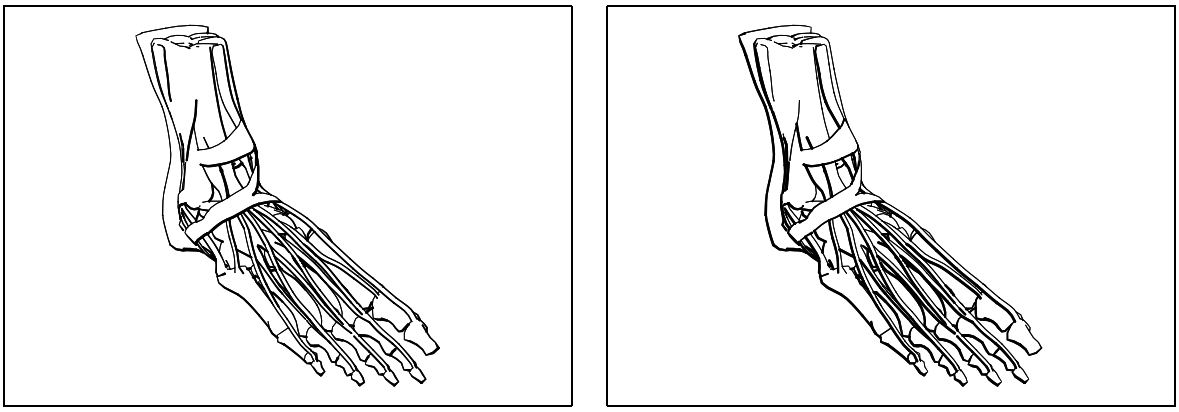


Abbildung 6.13: Visualisierung eines über die Szene streichenden schmalen Lichtkegels (1 104 Polygone).

Die Verwendung von verschieden starken Linienstärken zu Darstellung einer Beleuchtung wird oft in traditionellen Illustrationen angewendet (vgl. z. B. Abbildung 1.4), da so ein Rückschluß auf den räumlichen Aufbau der dargestellten Objekte ermöglicht wird. Dieser Effekt kann durch Einsatz des Beleuchtungseffektes erreicht werden. Abbildung 6.14 zeigt am Beispiel eines Fußes die Anwendung des Beleuchtungseffektes unter Beachtung unterschiedlicher Lichtquellen. In Abbildung 6.14(a) wird eine Lichtquelle links oberhalb vom Modell verwendet, in Abbildung 6.14(b) ist es eine Lichtquelle rechts oberhalb vom Modell.



(a) Lichtquelle links oben (b) Lichtquelle rechts oben

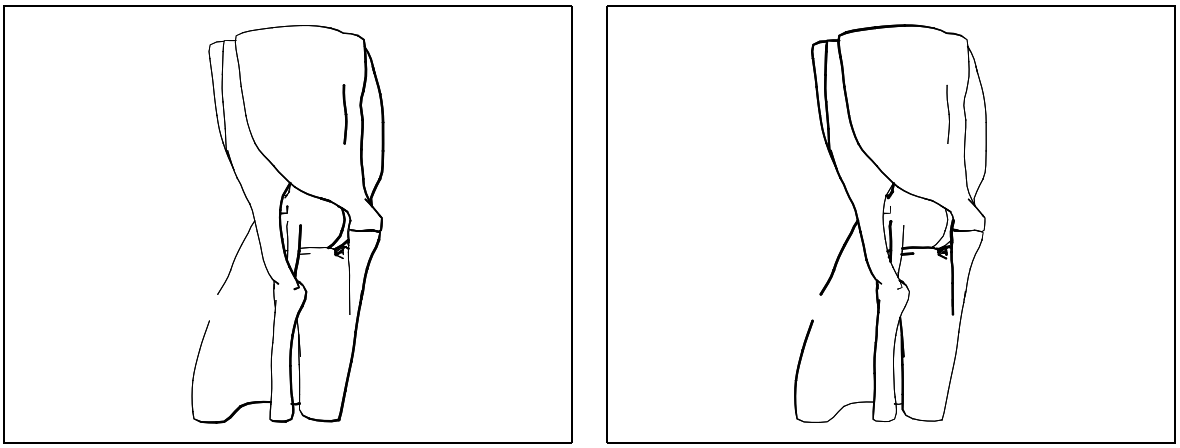
Abbildung 6.14: Verschiedene Wirkungsweise unterschiedlich zum Objekt positionierter Lichtquellen bei Anwendung des Beleuchtungseffektes am Beispiel eines menschlichen Fußes (11 244 Polygone).

Betrachtet man traditionelle Illustrationen, wird bei der Darstellung von Beleuchtung in Liniengraphiken meistens eine Lichtquelle links oberhalb der Objekte verwendet, wie es beispielsweise auch in Abbildung 1.4 gut erkennbar ist. RAMACHANDRAN begründet die Annahme einer Lichtquelle oberhalb der Objekte mit dem Fakt, daß die Sonnenstrahlung meist von oben kommt [Ram88]. Daß eine Lichtquelle links oberhalb der betrachteten Objekte angenommen wird, mag daran liegen, daß die meisten Menschen rechtshändig sind. Sie sind daher Lichtquellen von links oben bzw. links hinten gewohnt, um Schatten beim Schreiben zu vermeiden. Allerdings konnte keine Referenz zur Bevorzugung der Beleuchtung von links in der Literatur gefunden werden, obwohl bei Beleuchtungseffekten in Liniengraphiken fast ausschließlich diese Konvention verwendet wird (vgl. z. B. liniengraphische Abbildungen in [Hod89]).

Unterschiedliche Möglichkeiten bieten sich auch durch die Verwendung verschiedener Verläufe der Funktion $f(t)$, die über dem Winkel zwischen der Normalen und dem Beleuchtungswinkel oder über der Beleuchtungsintensität definiert ist. Durch einen abfallenden Verlauf werden die der Lichtquelle zugewandten Linien stärker hervorgehoben (vgl. Abbildung 6.15(a)), durch einen ansteigenden Verlauf werden die von der Lichtquelle abgewandten Linien stärker beeinflußt (vgl. Abbildung 6.15(b)). Üblicherweise wird meist letztere Variante bevorzugt, wie auch in [Hod89, Seite 91] angeführt wird. Somit erscheinen Linien im Schatten stärker als der Lichtquelle zugewandte Linien.

6.5 Kombinationsfunktionen

An einem Beispiel werden im folgenden die verschiedenen Wirkungsweisen der verschiedenen Kombinationsfunktionen demonstriert. Abbildung 6.16 zeigt zunächst zwei verschiedene Effekte, die jeweils auf dasselbe Objekt angewendet wurden. Konkret



(a) abfallende Funktion, d. h. beleuchtete Linien dünner als von Lichtquelle abgewandte Linien

(b) ansteigende Funktion, d. h. beleuchtete Linien stärker als von Lichtquelle abgewandte Linien

Abbildung 6.15: Unterschiedliche Wirkungsweise verschiedener Funktionsverläufe bei Anwendung des Beleuchtungseffektes am Beispiel eines menschlichen Knies (6 470 Polygone). Üblicherweise wird eher Variante (a) verwendet.

handelt es sich dabei um zwei Plane-Sweep-Effekte mit gleichem Effektstrahl, wobei beim ersten eine abfallende und beim zweiten eine ansteigende Effektfunktion verwendet wurde.

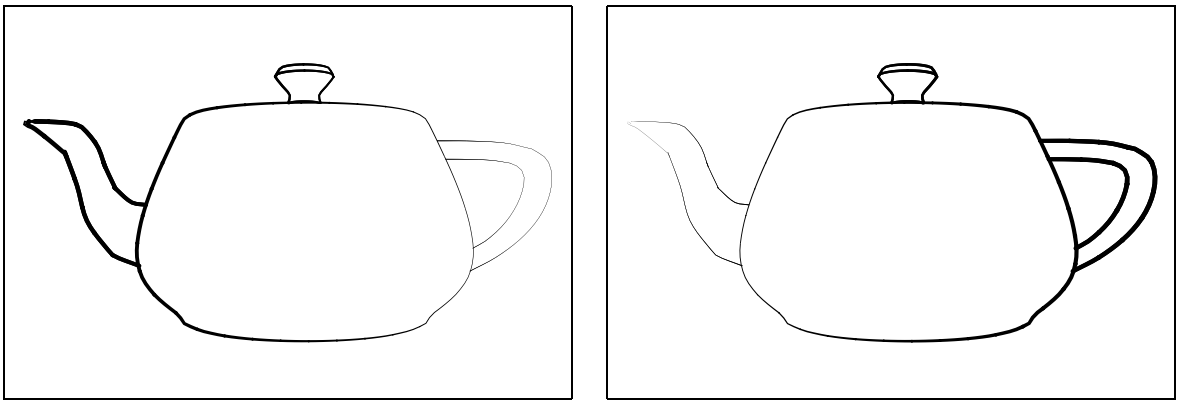
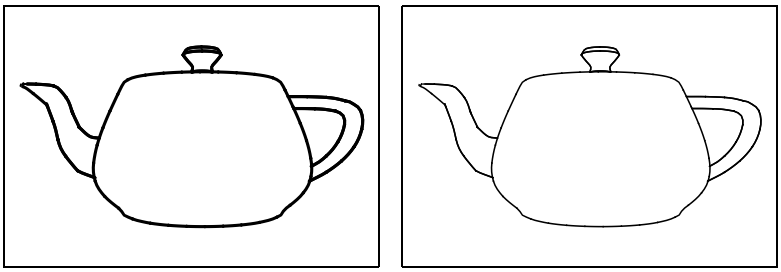


Abbildung 6.16: Zwei verschiedene Plane-Sweep-Effekte, mit abfallender bzw. ansteigender Effektfunktion (1 560 Polygone).

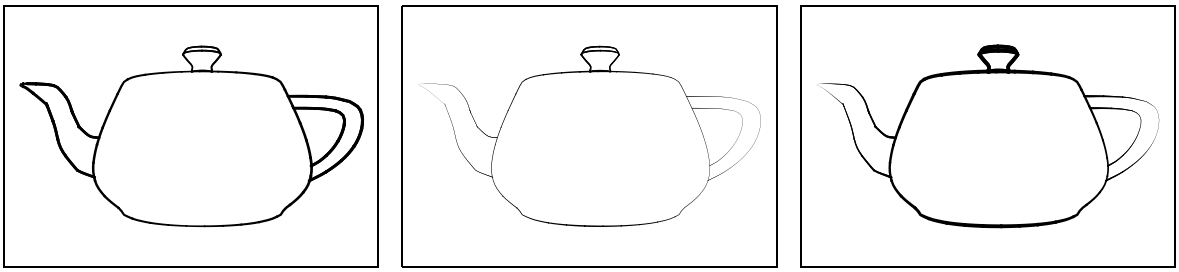
Diese beiden Plane-Sweep-Effekte wurden durch verschiedene Kombinationsfunktionen miteinander verknüpft. In Abbildung 6.17 sind die Ergebnisse der Anwendung der verschiedenen im System implementierten Verknüpfungsfunktionen (Addition, Durchschnitt, Maximum, Minimum und Multiplikation) dargestellt.

Es fällt auf, daß die Maximumfunktion (vgl. Abbildung 6.17(c)) eine Ausgabe erzeugt, in der man noch am besten die beiden ursprünglichen Effekte erkennen kann. Sie eignet sich daher auch am besten, um verschiedene Effekte gleichberechtigt miteinander zu

kombinieren und wurde auch fast ausschließlich in dieser Arbeit verwendet. Im Unterschied dazu sind die anderen Kombinationsfunktionen eher dazu geeignet, spezielle Effekte zu erzielen.



(a) Addition (b) Durchschnitt



(c) Maximum (d) Minimum (e) Multiplikation

Abbildung 6.17: Anwendung verschiedener Kombinationsfunktionen, mit denen die in Abbildung 6.16 gezeigten Effekte verknüpft wurden (1 560 Polygone).

6.6 Hinweise zur Verwendung des Programms

Während der Arbeit mit dem entwickelten System hat es sich im Interesse der Erzeugung qualitativ hochwertiger Graphiken als sinnvoll herausgestellt, einige Punkte zu beachten. Im folgenden werden diese Erfahrungen noch einmal zusammengefaßt:

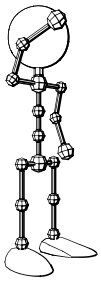
- Es sollten nicht zu viele Effekte in einem Bild oder in einer Animation kombiniert werden, da sonst das Bild oder die Animation überladen wird und die Übersichtlichkeit verloren geht.
- Mittels des Wurzeleffektes sollte eine von Null verschiedene Linienstärke und Liniensättigung eingestellt werden, um die Erkennbarkeit der gesamten Szene zu ermöglichen.
- Meist ist die Maximumfunktion als Kombinationsfunktion für mehrere Linienstileffekte am besten geeignet.

- Die Linienstärke sollte einen gewissen Wert nicht übersteigen, da sonst die Linien zu dick werden, was zum einen störend wirkt und zum anderen auch Details überdecken kann. Bei Versuchen mit dem System wurden Werte für die Linienstärke bis etwa Fünf verwendet.
- Bei zum Ausdruck bestimmten Einzelbildern sollte nur das Attribut Linienstärke verwendet werden, da die Verwendung von Liniensättigungen unterhalb von Eins beim Ausdruck zur Zeit noch Rasterungsprobleme aufwirft.
- Falls möglich sollten Kamerabewegungen bei Animationen vermieden werden und die darzustellenden Informationen durch Animation von Linienstileffekten visualisiert werden, um Rechenzeit zu sparen.
- Bei der Erstellung von Animationen sollte Oversampling (entweder direkt im Programm oder später durch eine Stapelverarbeitung der Einzelbilder) verwendet werden, um Aliasing-Effekte zu vermeiden und damit die Bildqualität zu verbessern.
- Die Anforderungen an die zu verwendenden Modelle, wie in Abschnitt 5.6 angeführt, sollten beachtet werden. So sind insbesondere Modelle mit geringerer Auflösung zu bevorzugen, da diese gegenüber fein triangulierten Modellen eine gleichwertige Ausgabe bei wesentlich geringeren Renderingzeiten erzeugen.

6.7 Zusammenfassung

Die angeführten Beispiele haben gezeigt, daß mit dem vorgestellten System sowohl illustrative Einzelbilder als auch Animationen erstellt werden können. Für einen Vergleich zwischen den in Abschnitt 1.2 besprochenen Vorbildern und mit dem vorgestellten System erzeugten Graphiken wird auf Anhang A verwiesen.

Animationen konnten aufgrund der Beschränkungen des Mediums Ausdruck auf Papier nur in Form von Bildserien vorgestellt werden. Zum Betrachten dieser als eigentliche Animationen sei auf die dieser Arbeit beigelegten CD-ROM verwiesen, die einige Beispielanimationen enthält (zum Inhalt vgl. Anhang B).



Kapitel 7

Zusammenfassung und Ausblick

7.1 Erreichte Ergebnisse

Illustrationen spielen bei der Vermittlung von Wissen eine große Rolle. Insbesondere Liniengraphiken sind gut geeignet, Lernende beim Verstehen von Sachverhalten zu unterstützen. Trotz enormer Fortschritte auf dem Gebiet der Computergraphik war die computerunterstützte Generierung von illustrativen Liniengraphiken bisher nur ungenügend möglich. In dieser Arbeit wurden Konzepte vorgestellt, mit denen es ermöglicht wird, aus dreidimensionalen Geometriemodellen geeignete liniengraphische Illustrationen zu erzeugen.

Es wurde erläutert, daß eine zweidimensionale Beeinflussung eines Bildes nicht für die computerunterstützte Erstellung qualitativ hochwertiger Illustrationen ausreicht. Statt dessen ist es notwendig, Informationen über den dreidimensionalen Aufbau der verwendeten Szene zu haben. Diese Informationen bilden eine Voraussetzung für den Einsatz sogenannter Linienstileffekte, die ebenfalls dreidimensionalen Charakter haben und sowohl eine objektspezifische als auch eine objektunspezifische Beeinflussung der Linienstilattribute ermöglichen.

Mit Linienstileffekten ist es somit möglich, Attribute auch unterhalb der Objektebene zu beeinflussen. Dies ist notwendig, da Modelle andernfalls in ihrer Objektstruktur auf die jeweilige Visualisierungsaufgabe speziell angepaßt werden müßten. Mit der Verwendung von Linienstileffekten können Modelle auch unabhängig von dieser Unterteilung verwendet werden.

Zur Verknüpfung von Linienstileffekten wird die Verwendung einer Effekthierarchie mit lokalem Keyframing vorgeschlagen. Diese Technik ermöglicht nicht nur die einfache Kombination verschiedener Effekte, sondern erhält zusätzlich auch noch die Flexibilität und Unabhängigkeit der einzelnen Effekte.

Es wurden exemplarisch vier Linienstileffekte entworfen und implementiert, die für die Erstellung von Illustrationen als besonders geeignet angesehen werden. Der *Plane-Sweep-Effekt* ermöglicht die Beeinflussung der Attributwerte entlang einer in der Szene definierbaren Richtung. Die Beeinflussung innerhalb eines Raumbereichs ist mit dem *Volumeneffekt* möglich, der in der hier vorgestellten Implementierung eine Beeinflussung entlang des Radius einer in der Szene spezifizierten Kugel ermöglicht. Die Visualisierung von Kameras bzw. von Lichteffekten ist mit Hilfe des *Kamera-* bzw. des

Beleuchtungseffektes möglich. Es wurde weiterhin gezeigt, daß die Komplexität der Berechnung der Effekte die Komplexität des gesamten Bilderzeugungsverfahrens nicht erhöht.

Das vorgestellte Linienstileffekt-Animationssystem eignet sich insbesondere für die Visualisierung unscharfer Größen wie z. B. Interesse, bei denen man keine binäre Entscheidung über deren Vorhandensein oder Ausprägung treffen kann, sondern wo kontinuierliche Übergänge dargestellt werden müssen. Da die vom System verwendeten Linienstilattribute *Linienstärke* und *Liniensättigung* auch in traditionellen Illustrationen dazu verwendet werden, lassen sich also unscharfe Eigenschaften durch geeignete Anwendung der Linienstileffekte und ihrer Effektfunktionen visualisieren. Des weiteren ist das System geeignet, logische Unterteilungen zu visualisieren, die in der Objektstruktur nicht enthalten sind. Insbesondere der Plane-Sweep-Effekt und der Volumeneffekt eigenen sich hierfür.

Mit der Animation von Linienstileffekten bietet sich die Möglichkeit, eine weitere Dimension zur Vermittlung von Wissen zu verwenden. Durch sich mit der Zeit verändernde Effekte können beispielsweise Zusammenhänge und Schlußfolgerungen gut visualisiert werden, was in Einzelbildern meist nicht einfach realisierbar ist.

Eine direkte Interaktion mit den Linienstileffekten in Echtzeit ist aufgrund des hohen Rechenaufwandes des analytischen Renderings nicht möglich. Um die Anwenderfreundlichkeit des Programms zu verbessern, wurden Verfahren wie Caching verwendet, die eine Beschleunigung gewisser Abläufe ermöglichen. Des weiteren wurden sogenannte Visualisierungselemente implementiert, die die Wirkungsweise der Effekte in Echtzeit in einer photorealistischen Vorschau anzeigen und damit die Interaktivität des Programms verbessern.

Das entwickelte Programm bietet sich für die Anwendung als Illustrationswerkzeug insbesondere für wissenschaftliche Illustrationen an. Speziell die Möglichkeit der Anwendung im medizinischen Bereich wurde in Kapitel 6 an mehreren Beispielen gezeigt. Aber auch für die computergestützte Generierung von Illustrationen für Bedienungsanleitungen eignet es sich, wie beispielsweise in Abbildung 6.6 gezeigt wurde. Bei diesem Anwendungsszenario ist von Vorteil, daß dreidimensionale Geometriemodelle der zu illustrierenden Objekte in den meisten Fällen bereits aus dem Herstellungsprozeß vorhanden sind und so einfach weiterverwendet werden können. Ein weiteres Anwendungsfeld ist die Visualisierung von Animationsparametern. Besonders der Kamera-Effekt und der Beleuchtungseffekt eignen sich für diese Anwendung.

7.2 Aufgetauchte Probleme

Ein wesentliches Problem tritt bei der Reduktion der Linienstilattribute *Linienstärke* und *Liniensättigung* auf Null auf. Die entsprechenden Linien der nicht dargestellten Teile der Objekte sind somit nicht mehr sichtbar, die dazugehörigen Kanten werden aber noch mit in die Sichtbarkeitsberechnung einbezogen. Somit werden andere Linien

bzw. Objekte von nicht mehr sichtbaren Objektteilen verdeckt, was zu inkonsistenten Bildern führt. Eine Lösung des Problems auf Objektebene wäre das Ausschließen der entsprechenden Objekte aus der Sichtbarkeitsberechnung. Für eine Behandlung unterhalb der Objektebene müßten die Ergebnisse der Linienstileffekt-Berechnung wieder in eine erneute Sichtbarkeitsberechnung einbezogen werden. Dieser Prozeß müßte solange wiederholt werden, bis keine Inkonsistenzen mehr auftreten. Dies würde allerdings zu einem höheren Berechnungsaufwand führen.

Technische Probleme wurden vor allem durch die verwendete Implementierung des analytischen Linienrenderers verursacht. So tauchen beispielsweise zeitliche Kohärenzprobleme auf. Außerdem generiert der verwendete Linienrenderer bei Modellen, bei denen Objekte den Sichtkörper der verwendeten Kamera schneiden, unerwünschte Triangulierungskanten. Des weiteren generiert der Renderer an sich überlappenden Objekten nicht immer alle notwendigen Schnittkanten. So mußte bei verwendeten Modellen darauf geachtet werden, daß die Objekte geringfügig voneinander abgerückt wurden, damit die notwendigen Kanten erzeugt wurden.

Ein weiteres Problem wird durch die Implementierung der Linienstile aufgeworfen. Bei der Ausgabe der erzeugten Linien werden die Attribute, wie in Abschnitt 3.5 beschrieben, in Polygonzüge umgewandelt. Damit wird eine nicht notwendige Auflösungsabhängigkeit erzeugt. Vermeiden ließe sich diese durch Verwendung von Kurvenzügen und Graustufenverläufen, die beispielsweise von der Seitenbeschreibungssprache PostScript Level 3 unterstützt werden. Die Beschreibung von Linienbegrenzungen als Kurvenzüge wird bereits von früheren Versionen von PostScript unterstützt.

7.3 Mögliche Erweiterungen und weiterführende Fragestellungen

Erweiterungsmöglichkeiten des vorgestellten Systems bestehen zunächst in technischer Hinsicht. Die Geschwindigkeitsprobleme einiger der verwendeten Algorithmen ließen sich durch Einbeziehung von vorcompiliertem Code mittels des DLL/C-Connect-Moduls von VisualWorks 2.5 zumindest abschwächen. Dies trifft u. a. auf die Implementierung des Oversamplings bei der Bitmapausgabe der Bilder, auf die Vorbereitungsphase des Plane-Sweep-Effektes sowie auf die eigentlichen Berechnungen aller Effekte zu.

Des weiteren wäre eine direkt-manipulative Benutzungsoberfläche wünschenswert. So könnten die Visualisierungselemente der einzelnen Linienstileffekte genutzt werden, um einzelne Parameter durch Maus-Interaktion in der OpenGL-Ansicht zu verändern. Somit könnte eine intuitivere Art der Parametereinstellung zusätzlich zu den bisher verwendeten Dialogen geschaffen werden.

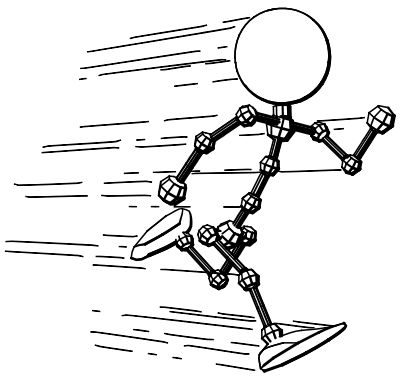
Weitere Erweiterungsmöglichkeiten bestehen in der Implementierung zusätzlicher Effekte bzw. in der Erweiterung der bestehenden, wie dies u. a. bereits in Kapitel 4 bei

der Beschreibung der einzelnen Effekte angesprochen wurde. Weiterhin wäre das Einbeziehen von Schraffuren nicht nur in die Darstellung, sondern auch in die Berechnung der Linienstilattribut-Beeinflussung möglich. Die Schraffurlinien müssen dazu ebenfalls durch dreidimensionale Koordinaten gegeben sein, um die Beeinflussung ausrechnen zu können. Die dazu benötigte Rechenzeit wäre aufgrund der Anzahl der Schraffurlinien natürlich sehr viel höher als bei Graphiken, die nur Konturlinien enthalten.

Die vorgestellte Implementierung und damit auch die gezeigten Beispiele beschränken sich auf die Beeinflussung der Linienstilattribute *Linienstärke* und *Liniensättigung*. Vor allem diese Attribute werden in traditionellen Illustrationen verwendet und wurden daher als ausreichend betrachtet, um die vorgestellten Algorithmen zu illustrieren. Es wäre wünschenswert, weitere Attribute mit in die Betrachtung einzubeziehen. Dabei bietet sich vor allem *Farbe* an, ein Attribut das eine zusätzliche Dimension für die Darstellung eröffnen würde. Derartige Graphiken sind zwar keine Liniengraphiken im ursprünglichen Sinne mehr, könnten aber weitere Ausdrucksmöglichkeiten eröffnen. Daneben sollte auch der Einfluß von verschiedenen Linienstilen auf die erstellten Graphiken untersucht werden.

Eine interessante weiterführende Fragestellung ist es, inwieweit die in Abschnitt 5.5 angesprochene Generierung von Bildserien erweitert werden könnte. Es ist beispielsweise vorstellbar, daß die lokalen Keyframes der einzelnen Effekte näher untersucht werden. Die sich tatsächlich verändernden Parameter könnten festgestellt, charakteristische Zeitpunkte bei den durch die lokalen Keyframes festgelegten Parameterfunktionen bestimmt und an diesen Zeitpunkten Einzelbilder generiert werden.

Als letzte Möglichkeit der Erweiterung des Systems soll die Kombination der Linienstileffekte mit dem von Schulz in [Sch99] entwickelten Ansatz zur Darstellung von Bewegung genannt werden (vgl. auch [MSS99]). Dazu müssen die in der genannten Arbeit entwickelten Bewegungslinien so erweitert werden, daß sie die Erzeugung von zeitlich frame-kohärenten Animationen ermöglichen. Die Anwendung einer Kombination von Linienstileffekten und Bewegungslinien kann den illustrativen Charakter von derart erzeugten Liniengraphiken sicher verstärken.



Literaturverzeichnis

[Cur98] Cassidy J. Curtis. Loose and Sketchy Animation. In *SIGGRAPH 98 Conference Abstracts and Applications*, Seite 317, New York, 1998. ACM SIGGRAPH, ACM Press.

[DC90a] Debra L. Dooley und Michael F. Cohen. Automatic Illustration of 3D Geometric Models: Lines. In Rich Riesenfeld und Carlo Sequin, Hrsg., *Proceedings of 1990 Symposium on Interactive 3D Graphics (Snowbird, Utah, March 1990)*, Seiten 77–82, New York, 1990. ACM SIGGRAPH, ACM Press.

[DC90b] Debra L. Dooley und Michael F. Cohen. Automatic Illustration of 3D Geometric Models: Surfaces. In *Proceedings of Visualization'90 (San Francisco, California, October 23–26 1990)*, Seiten 307–314, Los Alamitos, CA, 1990. IEEE Computer Society.

[Dud77] *Duden: Bildwörterbuch der deutschen Sprache*. Bibliographisches Institut, Mannheim, 1977.

[Dud96] *Duden. Rechtschreibung der deutschen Sprache*. Dudenverlag, Mannheim · Leipzig · Wien · Zürich, 21. Auflage, 1996.

[FBC+95] Jean-Daniel Fekete, Érick Bizouarn, Éric Cournarie, Thierry Galas und Frédéric Taillefer. TicTacToon: A Paperless System for Professional 2D Animation. In *Proceedings of SIGGRAPH 95 (Los Angeles, CA, August 6–11, 1995)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 79–89, New York, 1995. ACM SIGGRAPH, ACM Press.

[FS94] Adam Finkelstein und David H. Salesin. Multiresolution Curves. In *Proceedings of SIGGRAPH 94 (Orlando, Florida, July 24–29, 1994)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 261–267, New York, 1994. ACM SIGGRAPH, ACM Press.

[FvDFH90] James D. Foley, Andries van Dam, Steve K. Feiner und John F. Hughes. *Computer Graphics. Principle and Practice*. Addison Wesley Publishing Company, Reading, MA, 2. Auflage, 1990.

[Ger98] Nahum D. Gershon. Visualization of an Imperfect World. *IEEE Computer Graphics and Applications*, 18(4):43–45, Juli/August 1998.

[GP99] Christian Geiger und Volker Paelke. Zum Einsatz von Animation in interaktiven Benutzungsschnittstellen. In O. Deussen, V. Hinz und P Lorenz, Hrsg., *Simulation und Visualisierung '99*, Seiten 337–357, San Diego · Erlangen · Ghent · Delft, 1999. SCS – Society for Computer Simulation Int.

[HL94] Siu Chi Hsu und Irene H. H. Lee. Drawing and Animation Using Skeletal Strokes. In *Proceedings of SIGGRAPH 94 (Orlando, Florida, July 24–29, 1994)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 109–118, New York, 1994. ACM SIGGRAPH, ACM Press.

[HLW93] Siu Chi Hsu, Irene H. H. Lee und H. E. Wiseman. Skeletal Strokes. In *UIST'93 Proceedings of the ACM SIGGRAPH and SIGCHI Symposium on User Interface Software and Technology (Atlanta, November 1993)*, Seiten 197–206, New York, 1993. ACM Press.

[Hod89] Elaine R. S. Hodges, Hrsg. *The Guild Handbook of Scientific Illustration.* Van Nostrad Reinhold, New York, 1989.

[HPP+96] Karl-Heinz Höhne, Bernhard Pflesser, Andreas Pommert, Martin Riemer, Thomas Schiemann, Rainer Schubert und Ulf Tiede. A Virtual Body Model for Surgical Education and Rehearsal. *IEEE Computer - Innovative Technology for Professionals*, 29(1):25–31, Januar 1996.

[Lei95] Wolfgang Leister. Über das Illustrieren. In *Integration von Bild, Modell und Text '95*, Nummer 46 in ASIM Mitteilungen aus den Arbeitskreisen, Seiten 241–246, Magdeburg, 1995.

[LS95] John Lansdown und Simon Schofield. Expressive Rendering: A Review of Nonphotorealistic Techniques. *IEEE Computer Graphics and Applications*, 15(3):29–37, Mai 1995.

[Mei96] Barbara J. Meier. Painterly Rendering for Animation. In *Proceedings of SIGGRAPH 96 (New Orleans, LA, August 4–9, 1996)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 477–484, Reading, MA, August 1996. ACM SIGGRAPH, Addison Wesley Publishing Company.

[MS98] Maic Masuch und Thomas Strothotte. Visualizing Ancient Architecture using Animated Line Drawings. In *Proceedings of the International Conference on Information Visualization '98*, Seiten 261–266, Los Alamitos, CA, 1998. IEEE Computer Society.

[MSS97] Maic Masuch, Stefan Schlechtweg und Bert Schönwälder. daLi! – Drawing Animated Lines! In O. Deussen und P. Lorenz, Hrsg., *Simulation und Animation '97*, Seiten 87–95, Erlangen · Ghent · Budapest · San Diego, 1997. SCS – Society for Computer Simulation Int.

[MSS98] Maic Masuch, Lars Schumann und Stefan Schlechtweg. Animating Frame-to-Frame-Coherent Line Drawings for Illustrative Purposes. In P. Lorenz und B. Preim, Hrsg., *Simulation und Visualisierung '98*, Seiten 101–112,

Delft · Erlangen · Ghent · San Diego, 1998. SCS – Society for Computer Simulation Int.

[MSS99] Maic Masuch, Stefan Schlechtweg und Ronny Schulz. Speedlines: Depicting Motion in Motionless Pictures. In *SIGGRAPH 99 Conference Abstracts and Applications*, *Computer Graphics* Proceedings, Annual Conference Series, Seite 277, New York, 1999. ACM SIGGRAPH.

[Ram88] Vilayanur S. Ramachandran. Formwahrnehmung aus Schattierung. *Spektrum der Wissenschaft*, Seiten 94–103, Oktober 1988.

[Rog92] Andrew W. Rogers. *Textbook of Anatomy*. Churchill Livingstone, Edinburgh, 1992.

[SABS94] Michael P. Salisbury, Sean E. Anderson, Ronen Barzel und David H. Salesin. Interactive Pen-and-Ink Illustration. In *Proceedings of SIGGRAPH 94 (Orlando, Florida, July 24–29, 1994)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 101–108, New York, 1994. ACM SIGGRAPH, ACM Press.

[Sch92] Jutta Schumann. Linienqualität in ungenauen Grafiken. Diplomarbeit am Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg, Magdeburg, 1992.

[Sch94] Simon Schofield. *Non-photorealistic Rendering: A critical examination and proposed system*. Dissertation, School of Art and Design, Middlesex University, 1994.

[Sch97a] Bert Schönwälder. Generierung charakteristischer Linienzüge aus 3D-Modellen. Diplomarbeit am Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg, Magdeburg, 1997.

[Sch97b] Lars Schumann. Ein parametrisierbares Modell zur Darstellung von Linien. Diplomarbeit am Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg, Magdeburg, 1997.

[Sch99] Ronny Schulz. Visualisierung von Bewegungen in Liniengraphiken. Diplomarbeit am Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg, 1999.

[SMI99] Thomas Strothotte, Maic Masuch und Tobias Isenberg. Visualizing Knowledge about Virtual Reconstructions of Ancient Architecture. In *Proceedings Computer Graphics International*, Seiten 36–43, Los Alamitos, CA, 1999. The Computer Graphics Society, IEEE Computer Society.

[SPRF94] Thomas Strothotte, Bernhard Preim, Andreas Raab und David R. Forsey. How to Render Frames and Influence People. *Computer Graphics Forum*, 13(3):455–466, September 1994.

[SS99] Stefan Schlechtweg und Thomas Strothotte. Illustrative Browsing: A New Method of Browsing in Long On-line Texts. In *Human-Computer Inter-*

action INTERACT '99, Seiten 466–473, Amsterdam · Berlin · Oxford · Tokyo · Washington, DC, 1999. International Federation for Information Processing, IOS Press.

[SSRL96] Jutta Schumann, Thomas Strothotte, Andreas Raab und Stefan Laser. Assessing the Effect of Non-Photorealistic Rendered Images in CAD. In *Proceedings of CHI'96 (Vancouver, April 13–18 1996)*, Seiten 35–42, New York, 1996. ACM SIGCHI, ACM Press.

[ST90] Takafumi Saito und Tokiichiro Takahashi. Comprehensible Rendering of 3-D Shapes. In Forest Baskett, Hrsg., *Proceedings of SIGGRAPH 90 (Dallas, Texas, August 6–10 1990)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 197–206, New York, 1990. ACM SIGGRAPH, ACM Press.

[Str86] Steve Strassmann. Hairy Brushes. In *Proceedings of SIGGRAPH 86 (Dallas, Texas, August 18–22 1986)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 225–232, New York, 1986. ACM SIGGRAPH, ACM Press.

[Str98] Thomas Strothotte. *Computational Visualisation. Graphics, Abstraction and Interactivity.* Springer Verlag, Berlin · Heidelberg · New York, 1998.

[Sut63] Ivan E. Sutherland. Sketchpad: A Man-Machine Graphical Communication System. In *Proceedings of the Spring Joint Computer Conference*, Seiten 329–346, Baltimore, MD, 1963. Spartan Books.

[TJ95] Frank Thomas und Olli Johnston. *The Illusion of Life: Disney Animation.* Hyperion, New York, 1995.

[Wat93] Alan Watt. *3D Computer Graphics.* Addison Wesley Publishing Company, Reading, MA, 1993.

[Web96] *Random House Webster's College Dictionary.* Random House Inc., New York, 1996.

[Wel91] Emo Welzl. Smallest Enclosing Disks (Balls and Ellipsoids). In H. Maurer, Hrsg., *New Results and New Trends in Computer Science*, Band 555 aus *Lecture Notes in Computer Science*, Seiten 359–370. Springer-Verlag, Berlin · Heidelberg · New York, 1991.

[WS94] Georges Winkenbach und David H. Salesin. Computer-Generated Pen-and-Ink Illustration. In *Proceedings of SIGGRAPH 94 (Orlando, Florida, July 24–29, 1994)*, *Computer Graphics* Proceedings, Annual Conference Series, Seiten 91–100, New York, 1994. ACM SIGGRAPH, ACM Press.

[WW92] Alan Watt und Mark Watt. *Advanced Animation and Rendering Techniques.* Addison Wesley Publishing Company, Reading, MA, 1992.

Thesen zur Diplomarbeit

1. Liniengraphiken sind ein effektives Medium, um Sachverhalte zu illustrieren.
2. Eine Voraussetzung für die Generierung qualitativ hochwertiger illustrativer Liniengraphiken sind Informationen über den dreidimensionalen Aufbau der darzustellenden Objekte.
3. Für die computergestützte Generierung liniengraphischer Visualisierungen und Illustrationen aus dreidimensionalen Geometriemodellen ist es wichtig, die Linienstilattribute auch unterhalb der Objektebene manipulieren zu können.
4. Bei liniengraphischen Animationen verlangt die Behandlung zeitlich und räumlich konkurrierender Attributwerte besondere Problemlösungsstrategien.
5. Sowohl reines Keyframing als auch die Nutzung einer reinen Effekthierarchie sind bei liniengraphischen Animationen für die Animation von Linienstileffekten nicht ausreichend.
6. Die Verwendung einer Kombination von Effekthierarchie und lokalem Keyframing stellt eine einfache und intuitive Möglichkeit dar, Linienstileffekte miteinander zu verknüpfen.
7. Der geeignete Einsatz von Linienstileffekten in einzelnen Liniengraphiken verbessert ihre Einsatzmöglichkeit als Illustration.
8. Die Animation von Liniengraphiken stellt eine gute Möglichkeit dar, Zusammenhänge durch Anwendung von Linienstileffekten zu visualisieren, da durch die Zeit eine zusätzliche Dimension eingebracht wird.

Anhang A

Vergleiche von Vorbildern und Ergebnissen

Im folgenden werden einige Vergleiche zwischen angesprochenen Vorbildern und mit dem Linienstileffekt-Animationssystem erzeugten Graphiken angeführt. Dabei wird versucht, ähnliche Graphiken wie die in Abschnitt 1.2 angegebenen Beispiele für Illustration zu erzeugen.

Mit Abbildung A.2 wurde versucht, die in Abbildung A.1 dargestellte Illustration einer menschlichen Hand mit Fokus auf den Daumen mittels eines Volumeneffektes nachzuahmen. Es ist festzustellen, daß zwar die Binnenstrukturen aus Abbildung A.1 aufgrund des verwendeten Renderingverfahrens nicht nachzubilden sind, daß aber das Prinzip des Darstellungsstils nachgeahmt werden konnte.

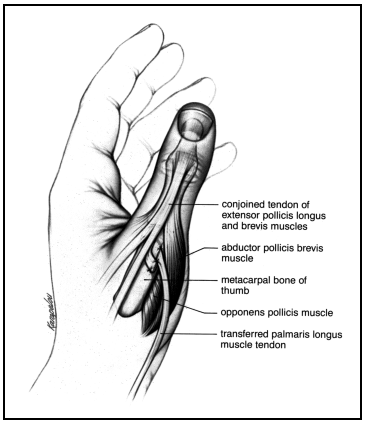


Abbildung A.1: Beispiel aus der Biologie: Hervorheben eines Gebietes, um das Interesse darauf zu lenken (aus [Hod89, Seite 56]). Wie Abbildung 1.1.

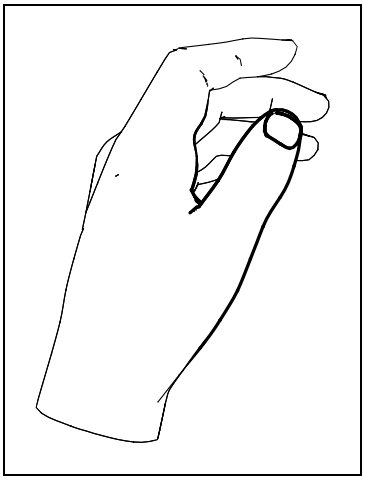


Abbildung A.2: Mit dem Linienstileffekt-Animationssystem erzeugtes Bild einer menschlichen Hand mit hervorgehobenem Daumen (3 161 Polygone).

Abbildung A.4 zeigt die Betonung des Kopfbereiches einer Figur ähnlich dem in Abbildung A.3 verwendeten Effekt. Verwendet wurde hier ebenfalls ein Volumeneffekt, der die Linienstärke im Bereich des Gesichtes anhebt.

Abbildung A.3: Beispiel aus einer Comic-Zeichnung: Betonung des Kopfes (aus [TJ95, Seite 442]). Wie Abbildung 1.2

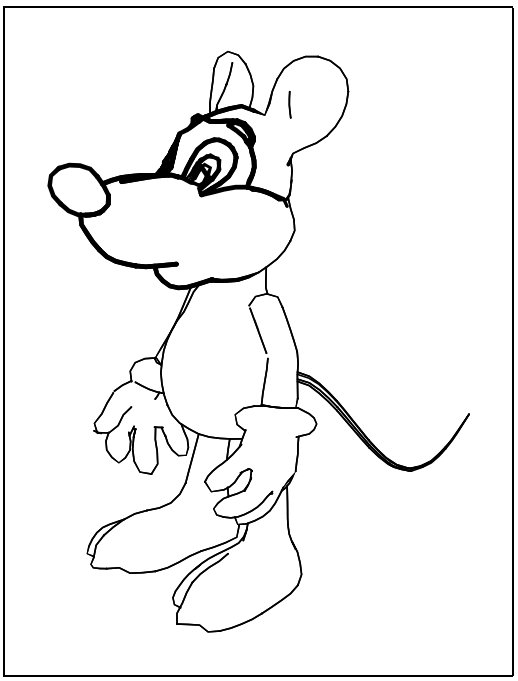


Abbildung A.4: Ähnlicher Effekt wie in Abbildung A.3 am Beispiel einer Comic-Maus (4 768 Polygone).

Ein weiteres Beispiel zeigt die in Abschnitt 1.2 angegebene Illustration von Kristallstrukturen (vgl. Abbildung A.5). Zur besseren Vergleichbarkeit wurden in einer zweiten Abbildung (vgl. Abbildung A.6) die verdeckten Kanten und die Beschriftungen entfernt. Abbildung A.7 zeigt schließlich die mit dem vorgestellten System erzeugten Graphiken, die unter Verwendung des Plane-Sweep-Effektes generiert wurden. Es ist zu erkennen, daß der Hervorhebungseffekt entsprechend dem Vorbild erreicht wurde.

Ein letztes Beispiel zeigt den Vergleich zwischen Vorbild und Realisierung des Beleuchtungseffektes. Abbildung A.8 ist ein Ausschnitt aus Abbildung 1.4 und zeigt eine Sichel, an der die Verwendung von Beleuchtung in Liniengraphiken besonders gut deutlich

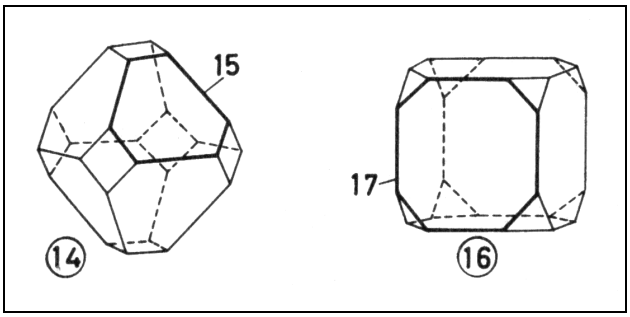


Abbildung A.5: Hervorhebung einer Seitenfläche der Körper (aus [Dud77, Seite 609]). Wie Abbildung 1.3.

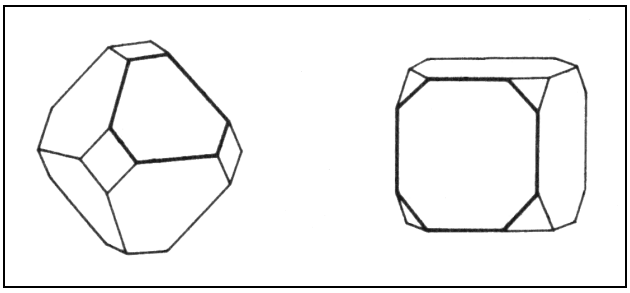


Abbildung A.6: Wie Abbildung A.5, nur verdeckte Kanten und Beschriftungen herausretuschiert (nach [Dud77, Seite 609]).

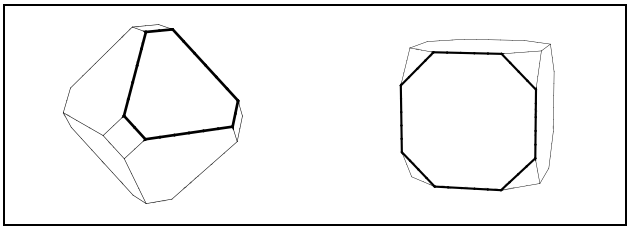


Abbildung A.7: Mit dem Linienstileffekt-Animationssystem erzeugtes Bild, modelliert nach Abbildung A.5 (88 Polygone). Wie Abbildung 6.5.

wird. Die Sichel wurde nachmodelliert und unter Anwendung des Beleuchtungseffektes gerendert. Das Ergebnis in Abbildung A.9 zeigt eindrucksvoll, daß eine gleichwertige Darstellung mit dem entwickelten System möglich ist.

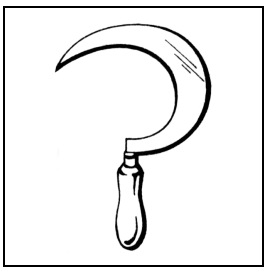


Abbildung A.8: Ausschnitt aus Abbildung 1.4, an dem die Anwendung von Beleuchtung in Illustrationen deutlich wird (aus [Dud77, Seite 105]).

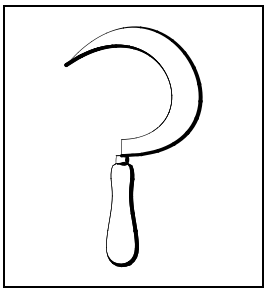


Abbildung A.9: In Anlehnung an Abbildung A.8 erzeugte Graphik unter Verwendung des Beleuchtungseffektes (2 000 Polygone).

Anhang B

Inhalt der CD-ROM

B.1 Übersicht

Auf der beigelegten CD-ROM sind im Verzeichnis `source/` die Dateien mit dem Quellcode abgelegt. Des weiteren ist im Verzeichnis `image/` ein Archiv (`*.tar.gz`) mit einem funktionsfähigen VisualWorks-Image inklusive aller notwendigen Dateien (Modelldaten, Linienstildateien, Applikationsdaten, etc.) vorhanden.

Darüber hinaus sind im Verzeichnis `video/` die Videodateien von einigen der in der Arbeit vorgestellten Beispiele abgespeichert. Die Videos wurden alle in jeweils drei Formaten erstellt:

- Audio Video Interleaved (`*.avi`), Codec Intel Indeo R5.0 (Verzeichnis `video/avi/`)
- Apple QuickTime (`*.mov`), Codec Intel Indeo R4.1 (Verzeichnis `video/mov/`)
- AnimatedGIF (`*.gif`) (Verzeichnis `video/gif/`)

Neben den Videodateien sind auch alle zugehörigen Einzelbilder sowohl als Vektorgraphiken im Format Encapsulated PostScript (`*.eps`) und als Pixelgraphiken im Windows-Bitmap-Format (`*.bmp`) in Archivform (`*.eps.tar.gz` bzw. `*.bmp.tar.gz`) im Verzeichnis `video/eps/` bzw. `video/bmp/` abgelegt. Für den Fall, daß die verwendeten Codecs oder das Programm zum Abspielen der QuickTime-Filme noch nicht vorhanden sind, sind im Verzeichnis `video/install/` die notwendigen Installationsdateien (für Win32-Betriebssysteme) zu finden.

B.2 Liste der Videos

`bending`: Skelett eines menschlichen Fußes mit sich bewegenden Zehen, dabei Hervorhebung eines Knochens durch Deakzentuierung aller anderen Knochen (vgl. Abbildung 2.13)

`dom_10sec`: Visualisierung des Schlusses von rekonstruierten Türmen auf das zweite Geschoß sowie die ungefähre Höhe eines historischen Gebäudes (vgl. Abbildung 6.1)

`dom_4sec`: wie `dom_10sec`, nur mit einer höheren Geschwindigkeit (vgl. Abbildung 6.1)

`domani2`: ähnlich `dom_10sec`, nur mit zusätzlich schrittweise von unten nach oben erscheinenden Türmen, was auf die Rekonstruktion der Türme aus den gefundenen Resten hinweisen soll

`hand`: Visualisierung der Gelenke an einem Knochenmodell einer sich schließenden menschlichen Hand (vgl. Abbildung 6.9)

`kamerafahrt_kamera`: Visualisierung des Sichtfeldes einer sich durch eine Szene bewegenden Kamera, betrachtet durch eine die ganze Szene erfassende Kamera (vgl. Abbildung 6.12(a))

`kamerafahrt_kamera2`: wie `kamerafahrt_kamera`, nur wurde die Linienstärke beeinflußt und die unbeeinflußten Linien sind mit einer geringen Linienstärke sichtbar gemacht worden (vgl. Abbildung 6.10)

`kamerafahrt_kamera3`: wie `kamerafahrt_kamera2`, nur bewegt sich die betrachtende Kamera um die Szene herum (vgl. Abbildung 6.11)

`kamerafahrt_kamera3_langsam`: wie `kamerafahrt_kamera3`, nur mit einer geringeren Geschwindigkeit (vgl. Abbildung 6.11)

`kamerafahrt_licht`: Verwendung eines gerichteten Lichtes mit sehr schmalen Lichtkegel, das über die Szene streicht

`kamerafahrt_licht2`: wie `kamerafahrt_licht`, nur sind die unbeeinflußten Linien mit einer geringen Liniensättigung sichtbar gemacht worden (vgl. Abbildung 6.13)

`kamerafahrt_voi`: Visualisierung eines sich ändernden Interessengebietes: der Fokus wandert entlang der Objekte der Szene, visualisiert durch einen dynamischen Volumeneffekt (vgl. Abbildung 6.8)

`thefoot`: Visualisierung des Verlaufs der Muskeln am Modell eines menschlichen Fußes (vgl. Abbildung 6.4)

`treppe`: Visualisierung des Zusammenhanges zwischen in einer Ausgrabung gefundenen Turmresten und der daraus rekonstruierten Turmtreppe (vgl. Abbildung 6.2)

Anhang C

Benchmarks

C.1 OpenGL-Transformations-Caching

Zur Messung der Beschleunigung der OpenGL-Anzeige der Animation durch Transformations-Caching wurden mit mehreren Modellen Versuche auf einer SGI Onyx 2 IR durchgeführt. Die Werte der folgenden Tabellen stellen Durchschnittswerte der gemessenen Bildwiederholraten (in fps – Bilder pro Sekunde) dar. Es wurden jeweils mehrere Versuche durchgeführt, um zufällige Einflüsse auszuschließen.

Tabelle C.1 zeigt die Ergebnisse in Abhängigkeit von der Größe der verwendeten Modelle, welche anhand der Polygonanzahl gemessen wurde. Wie zu erkennen ist, steigt die Beschleunigung durch das Caching mit steigender Modellgröße.

Polygonanzahl	ohne Caching (fps)	mit Caching (fps)	Beschleunigungsfaktor
12	62,30	71,04	1,14
144	39,86	70,65	1,77
876	19,15	62,09	3,24
9 300	4,75	26,78	5,64

Tabelle C.1: Beschleunigung durch Caching der OpenGL-Transformationsmatrizen.

Tabelle C.2 zeigt den Einfluß der Einbeziehung von Beleuchtung und Texturen in die Animation. Wie aus der Tabelle zu ersehen ist, hat diese zusätzliche Berechnung auch einen Einfluß auf den Beschleunigungsfaktor. Die Beschleunigung fällt nicht mehr so stark aus wie ohne Beleuchtungs- und Texturberechnung.

Einbezogene Elemente	ohne Caching (fps)	mit Caching (fps)	Beschleunigungsfaktor
Standardbeleuchtung	4,75	26,78	5,64
Beleuchtung und Texturen	4,71	22,58	4,79

Tabelle C.2: Einfluß der Einbeziehung von Licht und Texturen aus der Szenenbeschreibung in die OpenGL-Animation (bei einem Modell mit 9 300 Polygonen).

C.2 Analytisches Linienrendering

Für die Benchmarks zur benötigten Rechenzeit beim analytischen Linienrendering wurde ebenfalls eine SGI Onyx 2 IR verwendet. Tabelle C.3 zeigt die Ergebnisse einer Laufzeituntersuchung über den Zusammenhang von Modellgröße (gemessen in Polygonanzahl) und der benötigten Renderingzeit sowie der benötigten Zeit zum Anzeigen der Graphik auf dem Bildschirm. Die Werte sind jeweils Durchschnittswerte aus mehreren Versuchen, um zufällige Einflüsse zu minimieren.

Polygon-anzahl	Rendering-zeit (s)	Zeit pro Polygon (s/1000)	Ausgabe-zeit (s)	Zeit pro Polygon (s/1000)
12	0,19	15,8	0,42	35,0
88	1,13	12,8	1,11	12,6
144	1,37	9,5	0,86	6,0
862	9,20	10,7	5,86	6,8
876	11,48	13,1	2,64	3,0
994	34,15	34,4	3,63	3,7
1 104	12,33	11,2	3,13	2,8
3 040	55,96	18,4	11,17	3,7
4 208	71,36	17,0	7,97	1,9
6 073	110,07	18,1	20,60	3,4

Tabelle C.3: Einfluß der Modellgröße auf das analytische Linienrendering.

Wie zu erkennen ist, steigen die Werte beider Größen mit steigender Polygonzahl der Modelle, sind jedoch nicht nur davon abhängig. Als weitere Einflußfaktoren werden Objektzahl und Objektanordnung sowie die allgemeine Form der Objekte angenommen.

Anhang D

Kapitelbilder

Im folgenden werden die in den Kapitelüberschriften abgebildeten Graphiken noch einmal aufgeführt und die jeweils verwendeten Linienstileffekte benannt.

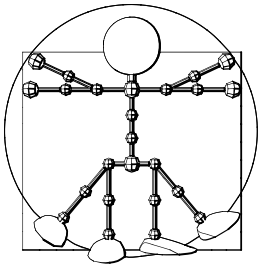


Abbildung D.1: (Schmutztitel) Beleuchtungseffekt – Lichtquelle oben links

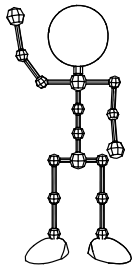


Abbildung D.2: (Kapitel 1) kein Effekt

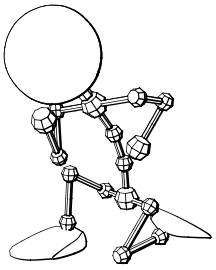


Abbildung D.3: (Kapitel 2) Beleuchtungseffekt – Lichtquelle oben links

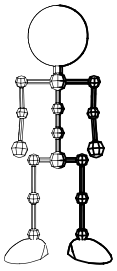


Abbildung D.4: (Kapitel 3) Plane-Sweep-Effekt – Visualisierung einer logischen Unterteilung (links-rechts)

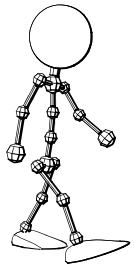


Abbildung D.5: (Kapitel 4) Beleuchtungseffekt – Lichtquelle oben links

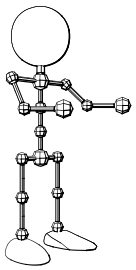


Abbildung D.6: (Kapitel 5) Beleuchtungseffekt – Lichtquelle oben links

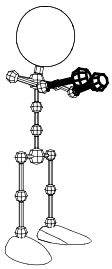


Abbildung D.7: (Kapitel 6) Kamera-Effekt – Visualisierung des von den Augen sichtbaren Bereiches

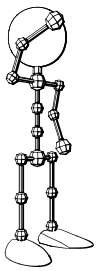


Abbildung D.8: (Kapitel 7) Beleuchtungseffekt – Lichtquelle oben links

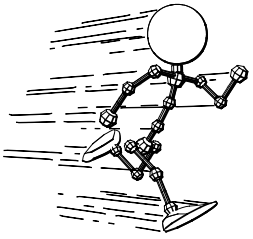


Abbildung D.9: (Kapitel 7, Ende) Beleuchtungseffekt mit zusätzlichen Bewegungslinien – Lichtquelle oben links