

Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik



Diplomarbeit

Parametrisierbare illustrative Darstellung
von medizinischen Oberflächenmodellen
mit Schattierungskarten

Roland Pfisterer

Institut für Simulation und Graphik

Parametrisierbare illustrative Darstellung
von medizinischen Oberflächenmodellen
mit Schattierungskarten

Diplomarbeit

an der
Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von:	ROLAND PFISTERER
geb. am:	24. November 1980
in:	Konstanz
Matrikelnummer:	163912
1. Gutachter:	Prof. Dr.-Ing. BERNHARD PREIM
2. Gutachter:	Dr.-Ing. FELIX RITTER
Betreuer:	Prof. Dr.-Ing. BERNHARD PREIM Dipl.-Ing. CHRISTIAN TIETJEN Dipl.-Ing. ALEXANDRA BAER
Zeit der Diplomarbeit:	12.07.2007 - 17.12.2007

Selbstständigkeitserklärung

Hiermit versichere ich, Roland Pfisterer (Matrikel-Nr. 163912), die vorliegende Arbeit allein und nur unter Verwendung der angegebenen Quellen angefertigt zu haben.

Roland Pfisterer

Magdeburg, 17.12.2007

Danksagung

Mindestens vier Fünftel meines Dankes gehen an meinen Betreuer Christian Tietjen, der immer wieder bereit war, meine wirren wissenschaftlichen Ausführungen zu durchforsten und in die richtigen Bahnen zu lenken.

Weiterer Dank ist gewiss:

...meinen Betreuern Prof. Dr. Bernhard Preim und Alexandra Baer, die trotz Kinder- bzw. Professorenorgen ebenfalls die Zeit fanden, diese Arbeit zu verbessern

...Rocco Gasteiger, der sich auch in der Endphase seiner eigenen Diplomarbeit nicht von seinem nervigen Labornachbarn aus der Ruhe bringen ließ, sondern alle Fragen freundlich beantwortete

...Mathias Neugebauer und Reis „Stefan“ Hiller, die immer wieder ein offenes Ohr und einen offenen Mund für Diskussionen über technische Probleme und Offscreen-Renderer hatten

...Hanna von Tenspolde, die sich ganz spontan zu den Korrektoren gesellte, aber zum Glück nicht viel zu beanstanden hatte

...meiner Mutter Gisela, die mit geistiger und finanzieller Unterstützung zum Gelingen meines Studiums beitrug und sich rechtschreibfehlersuchend durch Unmengen von Fachwörtern kämpfte

...meiner Schwester Sabine, die die schwere Diplomzeit mit einer lustigen Überraschung auflockerte und sich über Gleitkommaarithmetik wundern musste

... und meiner Freundin Natalie, die mir auf die Entfernung beistand und die beste Motivation lieferte, dieses Studium dann doch mal zu beenden.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Zielsetzung	1
1.2	Gliederung	3
2	Grundlagen und verwandte Arbeiten	5
2.1	Schattierung in medizinischen Illustrationen	5
2.1.1	Modellierung durch Licht	5
2.1.2	Modellierung durch Topografie	8
2.1.3	Räumliche Tiefe	9
2.1.4	Hervorhebung von Objekten	10
2.1.5	Gestaltungsprinzipien	10
2.2	Computergestützte Illustration von Objekten	12
2.2.1	Beleuchtungsmodelle	12
2.2.2	Eigenschaften der Oberfläche	20
2.2.3	Atmosphärische Perspektive	23
2.2.4	Kontrastverbesserung	23
2.2.5	Merkmalslinien	25
2.3	Szeneninformationen	26
2.3.1	Wichtigkeit	27
2.3.2	Distanz	29
2.3.3	Weitere Informationen	31
2.4	Zusammenfassung	31
3	Konzept	33
3.1	Anforderungsanalyse	34
3.2	Auswahl der Parameter	35
3.2.1	Oberflächengestaltung	36
3.2.2	Szeneninformationen	37
3.2.3	Bildoptimierung	37
3.3	Verschiedene Ansätze zur Parametrisierung	38
3.3.1	Standardwerte	40
3.3.2	Kombinierte Variablen	41
3.3.3	Verwendung von Templates	44
3.3.4	Aufwand der manuellen Parametrisierung	50

3.3.5	Abgestufte Einstellungen	51
3.3.6	Automatische Parametrisierung	51
3.3.7	Zusammenfassung	52
3.4	Entwurf des Systems	53
3.4.1	Kombination der Buffer	55
3.4.2	Vorverarbeitung	60
3.4.3	Transparenz	62
3.5	Zusammenfassung	63
4	Implementierung	65
4.1	Entwicklungsumgebung	65
4.2	Vorverarbeitung	67
4.3	Programmierung der Grafik-Hardware	69
4.3.1	Effiziente Nutzung von Texturen	69
4.3.2	Bildverarbeitung	70
4.3.3	Histogrammerzeugung	73
4.3.4	Framework in OpenGL	75
4.3.5	Berechnung der Schattierungskarte	76
4.4	Umsetzung des illustrativen Renderns	78
4.5	Zusammenfassung	79
5	Zusammenfassung	83
5.1	Ausblick	84
	Literaturverzeichnis	87

1 Einleitung

In der Computergrafik besteht seit gut 15 Jahren neben der Erzeugung möglichst realistischer Bilder auch eine gegensätzliche Zielsetzung: Unter dem Oberbegriff *Nicht-photorealistisches Rendering* werden Techniken zusammengefasst, die eine künstlerische und/oder abstrahierende Darstellung erreichen wollen. Dabei kann der Schwerpunkt mehr auf ästhetischer Visualisierung (Simulation von Tusche, Wasserfarbe etc.) oder auf illustrativer Darstellung liegen, wobei auch hier der ästhetische Eindruck eine Rolle spielen mag.

Dem zweiten Schwerpunkt ist das Gebiet der *Illustrativen Visualisierung* zuzurechnen. Hier geht es vor allem darum, visuelle Informationen mit den Mitteln der Illustration effektiv und sinnvoll zu vermitteln. So ist es bei Zeichnungen im Gegensatz zu Fotografien möglich, wichtige Aspekte zu betonen, unwichtige nur anzudeuten und so den jeweiligen Zweck der Illustration zu verstärken. In medizinischen, biologischen und technischen Fachbüchern finden sich überwiegend Zeichnungen, da mit ihrer Hilfe komplexe Strukturen wie Organe oder technische Bauteile auf ihre wesentlichen Aspekte reduziert werden können.

Illustrative Computergrafik dient auch dazu, Illustrationen und interaktive 3D-Grafiken zu verknüpfen. Eines der Anwendungsgebiete hierfür ist die medizinische Visualisierung, in der illustrative Techniken eingesetzt werden, um dreidimensionale Modelle von Organen für Lehre, Operationsplanung und Patientenaufklärung darzustellen. Zwei Techniken, die am Lehrstuhl für Visualisierung der Otto-von-Guericke-Universität verfolgt werden, sind *Stippling* und *Hatching*. Beide Verfahren stellen Objekte als Schwarz-Weiß-Zeichnung dar, wobei verschiedene Grautöne durch die Dichte von Punkten (Stippling) bzw. Schraffurlinien (Hatching) vermittelt werden. Traditionell werden sie schon seit mehreren Jahrhunderten in medizinischen Atlanten einzeln oder kombiniert eingesetzt, weshalb sie auch heute noch in der medizinischen Visualisierung eine Rolle spielen.

1.1 Zielsetzung

In einer Studie zur Ästhetik von Stippling stellen ISENBERG et al. [2006] fest, dass computergenerierte Stippling-Illustrationen sich gegenüber handgefertigten unter

anderem durch eine höhere Präzision auszeichnen. Als einer der Gründe wird die korrekte Schattierung genannt, da diese durch ein algorithmisches Beleuchtungsmodell gewonnen wird. Handgezeichnete Illustrationen weisen dagegen oft Lichtverhältnisse auf, die in der Realität nicht nachgestellt werden können. Dies dient dazu, die Formwahrnehmung eines Objektes zu unterstützen, die Aufmerksamkeit des Betrachters zu lenken und einen ausgewogenen Gesamteindruck zu erreichen. Die Studie ergab zwar, dass die leichte Identifizierung der computergenerierten Zeichnungen nicht als Nachteil empfunden wurde. Dennoch stellt sich die Frage, ob der wahrgenommene Unterschied zu Handzeichnungen durch einen eher gestalterischen Umgang mit Schattierung verringert werden könnte.

In bisherigen Implementierungen von Stippling und Hatching werden meist eine oder mehrere Lichtquellen eingesetzt, um über einfache lokale Beleuchtungsmodelle die Helligkeit jedes Oberflächenpunktes zu bestimmen. In der Regel ergibt sich dabei nur aus einer Blickrichtung ein optimaler Eindruck. Um ein 3D-Objekt interaktiv zu betrachten, müsste eine verständliche und ansprechende Schattierung aus allen möglichen Blickrichtungen garantiert sein. Dies würde für komplexe Objekte und Szenen eine ebenso komplexe Beleuchtung erfordern, welche für jeden Datensatz individuell erstellt werden müsste. Daher beschäftigt sich die vorliegende Diplomarbeit mit dem Auffinden zusätzlicher oder alternativer Parameter für Schattierung, die unabhängig von der Betrachtungsrichtung einen optimalen Eindruck der Szene vermitteln. Diese können sich aus der Form der Objekte, dem Zusammenhang der Szene oder der Betrachtungsrichtung ergeben. „Schattierung“ bezeichnet dabei im weiteren Verlauf den generellen Einsatz von Hell-Dunkel-Kontrasten zum Darstellen von Objekten, im Gegensatz zur reinen Schattierung nach Beleuchtung.

Ziel der Arbeit ist die Integration aller Parameter zu einer *Schattierungskarte*, mit der medizinische Oberflächenmodelle verständlich und optisch ansprechend schattiert werden können. Es ist auch denkbar, weitere optische Eigenschaften wie Transparenz oder, im Falle von farbigen Illustrationen, Farbton und Sättigung über solche Parameter zu steuern. Als Grundlage zum Auffinden der Parameter dienen medizinische Atlanten und Fachbücher, zusätzlich werden auch allgemeine Grundsätze der Gestaltung in Betracht gezogen. Da wahrscheinlich keine optimale Schattierung für beliebige Szenen existiert, erfordert die Einstellung der Parameter eine gewisse Interaktion des Benutzers. Somit ist es auch Teil dieser Arbeit, Ansätze dafür zu entwickeln, wie dem Benutzer Auswahl und Integration der Parameter erleichtert und in gewissem Rahmen vorgegeben werden können.

Ein wesentlicher Vorteil der Computervisualisierung gegenüber Druckmedien besteht darin, Objekte wie in der Realität frei betrachten zu können. Muss jede Ansicht erst langwierig berechnet werden, schwindet dieser Vorteil, daher spielt die Darstellung in Echtzeit eine große Rolle. Seit kurzer Zeit existiert die Möglichkeit, auf Grafik-Hardware *Shader-Programme* ablaufen zu lassen, mit denen der

Rendervorgang angepasst werden kann. Sie können auch dazu verwendet werden, allgemeinere Berechnungen, insbesondere Algorithmen der Bildverarbeitung, mit hoher Geschwindigkeit auf der Grafikkarte auszuführen. Dieses Potenzial wird für die Umsetzung genutzt, um eine Visualisierung in Echtzeit zu ermöglichen.

1.2 Gliederung

Die Arbeit ist wie folgt gegliedert:

Kapitel 2: In diesem Kapitel werden mögliche Schattierungs-Parameter erarbeitet. Dies geschieht durch Analyse von Illustrationen aus medizinischen Atlanten sowie Betrachtung allgemeiner gestalterischer Grundsätze. Im Anschluss werden Arbeiten der Computergrafik vorgestellt, die Methoden für die Erzeugung der genannten Parameter aufzeigen. Die Betrachtung von Arbeiten der medizinischen Visualisierung liefert Anregungen für weitere Parameter.

Kapitel 3: Dieses Kapitel beinhaltet eine Analyse der Anforderungen, die an das gewünschte System zur illustrativen Schattierung gestellt werden. Anschließend wird ein Systementwurf beschrieben, der diese Anforderungen erfüllt. Die ermittelten Parameter werden entsprechend ihrer Auswirkungen kategorisiert und in eine Renderpipeline integriert. Weiter wird ausführlich besprochen, wie dem Benutzer die Parametrisierung der Schattierungskarte erleichtert werden kann, sowie die Zweckmäßigkeit verschiedener Parametrisierungen anhand von Beispielen für diverse anatomische Strukturen gezeigt.

Kapitel 4: Es wird erläutert, wie der Systementwurf unter Verwendung von programmierbarer Grafik-Hardware realisierbar ist. Dabei wird auf das Offscreen-Rendern mittels *Framebuffer Objects*, den Einsatz von Shader-Programmen zur Bildverarbeitung und die Histogrammerzeugung in Echtzeit eingegangen.

Kapitel 5: Zum Abschluss wird das entworfene System zusammengefasst und mit der Zielsetzung verglichen. Es werden Möglichkeiten der Verbesserung und Erweiterung besprochen und weitere Projekte aufgezeigt, die sich aus der Zusammenführung mit anderen Arbeiten ergeben.

2 Grundlagen und verwandte Arbeiten

Zahlreiche Arbeiten beschäftigen sich mit der Erzeugung aussagekräftiger Illustrationen durch Anpassung der zeichnerischen Elemente. Die Verteilung von Helligkeit in Illustrationen wurde dagegen bisher vergleichsweise wenig beachtet. Das folgende Kapitel beschäftigt sich daher mit der Analyse medizinischer Illustrationen unter dem Gesichtspunkt der Schattierung. Dabei handelt es sich sowohl um Beleuchtungstechniken als auch um diverse andere Gestaltungsmittel, die sich Variationen der Helligkeit zunutze machen.

Zunächst findet eine Analyse entsprechender Gestaltungsmittel anhand von Beispielen aus medizinischen Atlanten statt. Anschließend werden Arbeiten diskutiert, die für die Umsetzung der analysierten Parameter auf 3D-Modellen hilfreich sind und weitere Parameter aufzeigen. Die betrachteten Arbeiten werden in zwei Bereiche unterteilt: die allgemeine Darstellung von Objekten und die Visualisierung zusätzlicher Informationen, die über den rein optischen Eindruck hinausgehen.

2.1 Schattierung in medizinischen Illustrationen

Im folgenden Abschnitt werden Illustrationen auf ihre Verwendung von Schattierung hin analysiert. Viele der Parameter leiten sich von generellen zeichnerischen Gestaltungsmitteln ab und sind daher auf verschiedene Gebiete der Visualisierung anwendbar. Dem Rahmen dieser Arbeit zufolge beschränkt sich die Analyse jedoch auf Schwarz-Weiß-Zeichnungen aus dem medizinischen Bereich.

2.1.1 Modellierung durch Licht

Die Darstellung der Objektform ist offensichtlich die Grundlage einer guten Illustration und wird deshalb als erstes betrachtet. In der Realität entsteht jeder visuelle Eindruck aus der Interaktion von Licht mit Materie. Daher wird auch in Illustrationen praktisch immer eine Form von Beleuchtung verwendet, um Objekte

zu visualisieren. Dies geschieht durch eine globale Lichtquelle, die meist als von schräg oben kommend dargestellt wird, was unserer täglichen Wahrnehmung des Sonnenlichtes entspricht (siehe Abb. 2.2). Häufig kommt auch eine Beleuchtung aus Richtung des Betrachters vor.

In der Realität sorgt reflektiertes Licht der Umgebung dafür, dass Schattenbereiche nicht völlig schwarz erscheinen. Dieses Streulicht wird von Zeichnern ebenfalls eingesetzt, um Strukturen in nicht direkt beleuchteten Bereichen erkennbar zu machen. Die gleiche Funktion erfüllt das Aufhellungslicht, das in der Studiofotografie und Filmproduktion verwendet wird. Es kommt meist aus einer Richtung, die etwa dem an der Betrachtungsrichtung gespiegelten Hauptlicht entspricht (siehe Abb. 2.1(a)).

Im Gegensatz zur Fotografie steht es dem Illustrator frei, die Lichtverhältnisse in einer Zeichnung so anzupassen, dass eine optimale Erkennbarkeit von Objektform und -oberfläche erreicht wird. Für eine genauere Analyse solcher Methoden wird auf ein Standardwerk der Illustration, „The Guild Handbook of Scientific Illustration“ [HODGES, 1989], zurückgegriffen. Im Folgenden werden die wichtigsten Beleuchtungstechniken neben Haupt- und Aufhellungslicht aus diesem Werk genannt. In Abb. 2.1 sind die jeweiligen Lichtverhältnisse schematisch dargestellt.

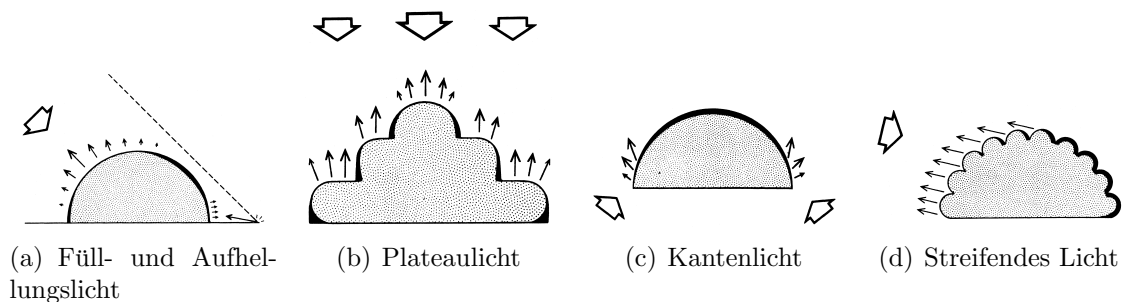


Abb. 2.1: Diagramme der Beleuchtungstechniken aus HODGES [1989]. Die breiten Pfeile zeigen das einfallende Licht an, die schmalen Pfeile das vom Objekt reflektierte Licht. Der Betrachter schaut jeweils senkrecht von oben auf das Objekt.

Plateaulicht: Eine lokale Version des Hauptlichtes aus Betrachtungsrichtung ist das Plateaulicht (Abb. 2.1(b)). Es dient dazu, die Silhouette hervorzuheben und Objekte besser von einem hellen Hintergrund zu trennen. Bei runden oder röhrenförmigen Objekten erscheint dadurch das Zentrum heller und die Randbereiche dunkler. Dies kann die Formwahrnehmung verstärken, wie an den Beinen des Torsos in Abb. 2.3 zu sehen ist. Der Kantenschatten, eine Variante des Plateaulichtes, wird hier nicht gesondert besprochen, da die beiden Techniken sich kaum unterscheiden.

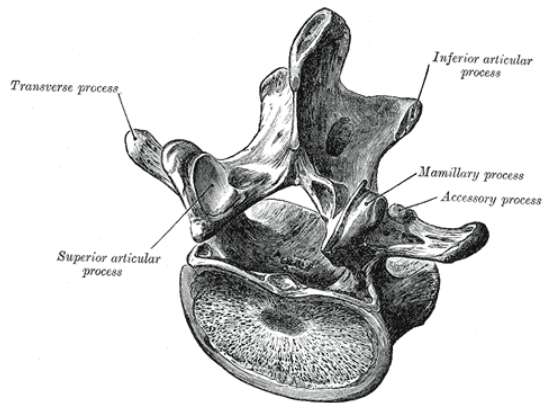


Abb. 2.2: Überwiegend realistische Beleuchtung mit Hauptlicht von links oben. Schattenwurf ist angedeutet, aber nicht konsistent, einige theoretisch im Schatten liegende Bereiche erscheinen beleuchtet. Quelle: GRAY [1918]

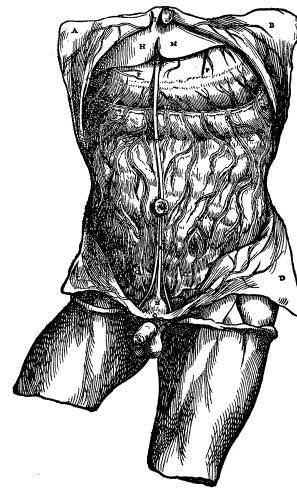


Abb. 2.3: Das Hauptlicht kommt von rechts oben, die runde Form der Beine wird jedoch durch den lokalen Einsatz von Plateaulicht stärker betont. Quelle: VESALIUS [1543]

Kantenlicht: Den Gegensatz zum Plateaulicht stellt das Kantenlicht dar, welches von hinten auf das Objekt treffend seinen Rand erhellt und es damit von dunklem Hintergrund abhebt (Abb. 2.1(c), Abb. 2.4). Kantenlicht hat meistens einen kleineren Einflussbereich als Plateaulicht und kann daher gleichzeitig eingesetzt werden.

Streifendes Licht: Details sind am besten zu erkennen, wenn das Licht etwa tangential zur Oberfläche steht (Abb. 2.1(d)). Dies machen sich Illustratoren zunutze, indem sie die Lichtrichtung jeweils an die Oberfläche anpassen, ohne den Gesamteindruck einer realistischen Hauptlichtquelle zu beeinträchtigen. So können Details und Unebenheiten einer Oberfläche wie bei dem Schädelknochen in Abb. 2.5 hervorgehoben werden.

Schattenwurf: Bei den vorigen Beleuchtungstechniken entsteht Schattierung aus der Lage der Oberfläche zum Licht, dem sogenannten Eigenschatten. Zusätzlich kann Schlagschatten eingesetzt werden, der von einem Objekt auf sich selbst oder auf andere Oberflächen geworfen wird. Er kann helfen, die Objektform und die Lage mehrerer Objekte zueinander zu verdeutlichen (siehe Abb. 2.2, Abb. 2.13). Meistens wird er nicht physikalisch korrekt gezeichnet, sondern nur angedeutet. Tatsächlich kann zu viel Schattenwurf auf die Objektfläche eher störend wirken und als Hinweis auf deren Form missverstanden werden, weshalb er relativ selten angewandt wird. Der Überdeckungsschatten, eine spezielle Form von Schattenwurf, wird in Abschnitt 2.1.3 besprochen.

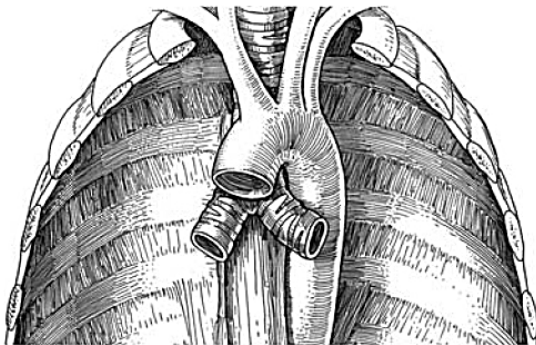


Abb. 2.4: Das Kantenlicht hebt den Rand der Speiseröhre besser vom dunklen Hintergrund ab. Quelle: ANDREWS [2006]

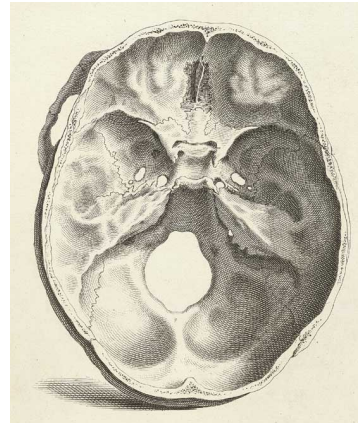
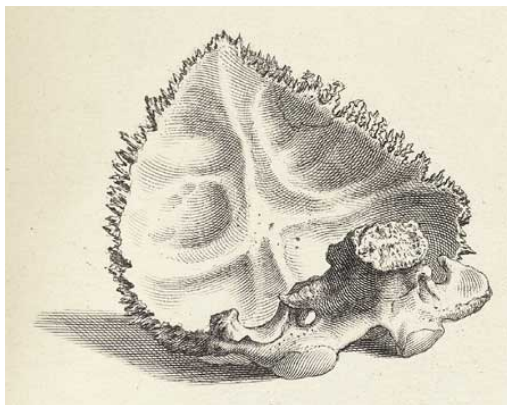


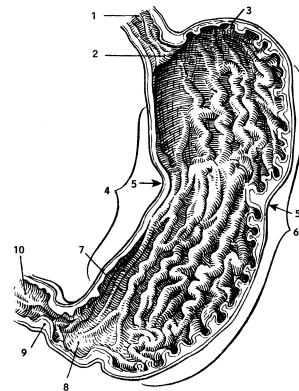
Abb. 2.5: Die Unebenheiten der Oberfläche werden durch streifendes Licht hervorgehoben. Quelle: CHESELDEN [1733]

2.1.2 Modellierung durch Topografie

Zur Illustration der Oberfläche werden auch Techniken benutzt, die nicht auf Lichteinfall zurückgehen. So können wie bei geologischen Höhenprofilen Erhöhungen und Vertiefungen, im weiteren Verlauf Berge und Täler genannt, besonders betont werden. Die Illustration der Magenwand in Abb. 2.6(b) verwendet fast ausschließlich diese Technik. Es ist aber anzumerken, dass sie in der Regel nicht ausschließlich, sondern wie in Abb. 2.6(a) zur Akzentuierung von üblichen Beleuchtungsmethoden eingesetzt wird, da sie für sich genommen nur selten eine ausreichende Darstellung ermöglicht.



(a) Quelle: CHESELDEN [1733]



(b) Quelle: TITTEL [1994]

Abb. 2.6: Betonung der Oberflächenstruktur durch zusätzliches Aufhellen von Bergen (beide) und Abdunkeln von Tälern (rechts).

2.1.3 Räumliche Tiefe

Da in einer Zeichnung die dritte Dimension fehlt, verwenden Illustratoren optische Hinweise, um die Tiefe einer Szene besser zu verdeutlichen. Zunächst gibt es den Effekt der atmosphärischen Perspektive. In der Natur ist zu beobachten, dass durch die Streuung von Licht in der Atmosphäre weit entfernte Objekte heller und kontrastärmer erscheinen. Dies wird seit Jahrhunderten als Technik in Landschaftsgemälden angewandt und kann auch sehr einfach in einer Zeichnung benutzt werden, um die räumliche Wirkung zu erhöhen (siehe Abb. 2.7). Die atmosphärische Perspektive gibt einen groben Eindruck von Tiefe wieder und sorgt für Übersichtlichkeit, indem sie Vorder- und Hintergrund optisch trennt. Im Gegensatz dazu kann auch speziell die räumliche Anordnung von Objekten betont werden. Dies geschieht eher selektiv auf einigen Objekten als für die gesamte Szene (siehe Abb. 2.8).

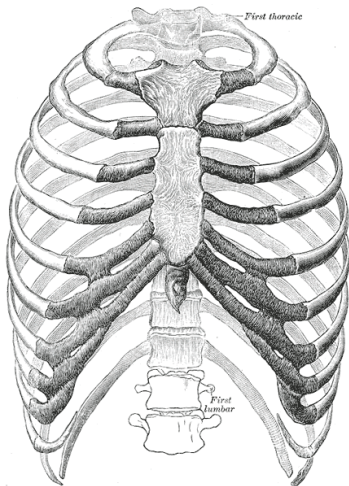


Abb. 2.7: Die räumliche Ausdehnung des Brustkorbes wird durch Kontrastverringern in der Tiefe betont. Zusätzlich kann auch die Stärke der Silhouette variiert werden. Quelle: GRAY [1918]

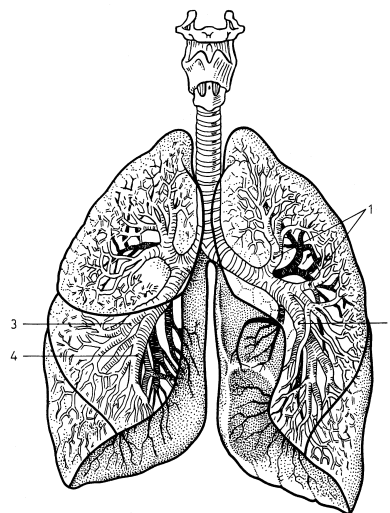


Abb. 2.8: Zur Visualisierung der räumlichen Lage werden weiter hinten liegende Gefäße graduell dunkler schattiert. Die Lungenflügel sind nur durch ihre Silhouette angedeutet. Quelle: TITTEL [1994]

Eine dritte Möglichkeit für die Hervorhebung der räumlichen Verhältnisse besteht darin, die Distanz zwischen überlappender Geometrie zu betonen. Dies kann durch Schattenwurf von Vorder- auf Hintergrundobjekte erreicht werden (siehe Abb. 2.9). Dieser Überdeckungsschatten scheint von einer Lichtquelle aus Richtung des Betrachters geworfen zu werden. Er ist unabhängig von der Richtung des Hauptlichtes und wird nur lokal eingesetzt, womit er sich von echtem Schattenwurf unterscheidet. Zusätzlich wird das vordere Objekt zum Rand hin oft noch aufgehellt, was den Kontrast ähnlich wie beim Kantenlicht in Abb. 2.4 verstärkt.

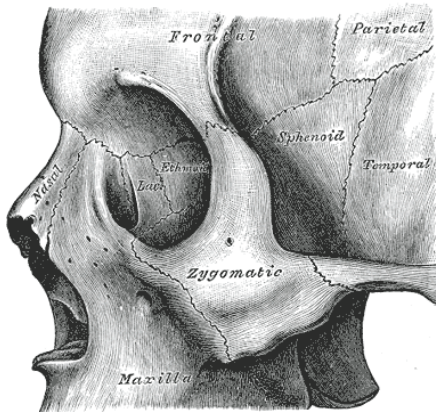


Abb. 2.9: An Stellen mit großem Tiefenunterschied sind hintere Flächen des Schädels abgedunkelt und vordere aufgehellte, um sie optisch zu trennen. Quelle: GRAY [1918]

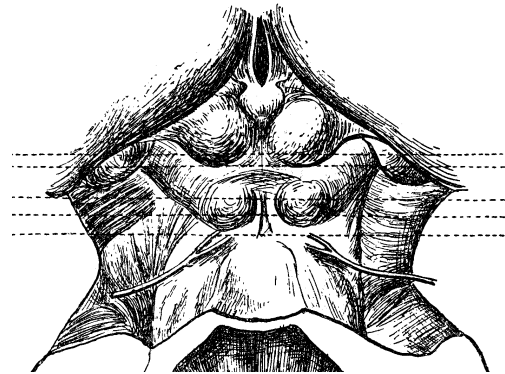


Abb. 2.10: In der oberen Bildhälfte wird der Kontrast zwischen dunklem Vorder- und hellem Hintergrund verstärkt. Quelle: KOPSCH und RAUBER [1950] (Ausschnitt)

Überdeckungsschatten dunkelt den Hintergrund unabhängig von der tatsächlichen Helligkeit ab, was ungewollt den Kontrast abschwächt, falls das vordere Objekt dunkler als das hintere ist. Alternativ dazu kann auch der bestehende Kontrast zwischen Objekten verstärkt werden. Dieser Effekt erscheint jedoch oft sehr subtil (siehe Abb. 2.10).

2.1.4 Hervorhebung von Objekten

Ein Vorteil von Illustrationen gegenüber Fotografien besteht darin, dass wichtige Elemente hervorgehoben und unwichtige abgeschwächt werden können. Häufig geschieht dies dadurch, dass Objekte, die als räumlicher Kontext dienen, nur durch ihre Silhouette angedeutet werden (siehe Abb. 2.8, Abb. 2.11). Statt einer diskreten Einteilung der Objekte nach Wichtigkeit kann die Abschwächung auch kontinuierlich erfolgen. Dies ist vor allem sinnvoll, wenn es einen Fokusbereich gibt, dessen Umgebung mit zunehmender Entfernung unwichtiger wird. In Abb. 2.12 ist dies am Beispiel einer Operationszeichnung zu sehen.

2.1.5 Gestaltungsprinzipien

Des Weiteren gibt es zahlreiche Gestaltungsprinzipien, die traditionell in Malerei und Grafik angewendet werden. Bei Zeichnungen besonders zu beachten ist der Hell-Dunkel-Kontrast. Ein großer Kontrastumfang lässt Zeichnungen dynamischer und

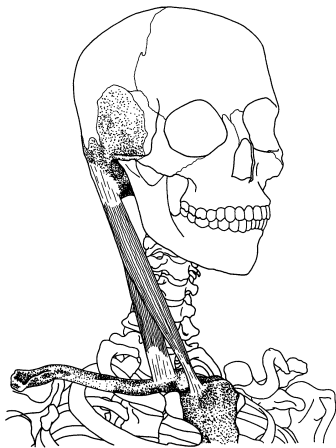


Abb. 2.11: Diskrete Abschwächung: Die Knochen dienen nur zur Orientierung und werden nicht schattiert, um den Blick auf die hier relevanten Halsmuskeln zu lenken. Quelle: STONE und STONE [2000]

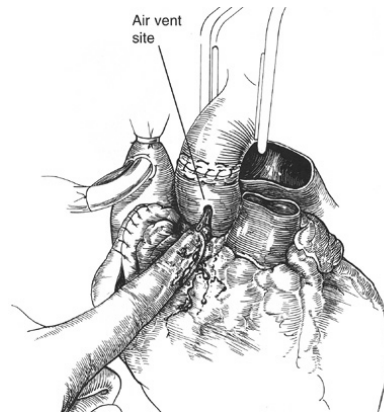


Abb. 2.12: Kontinuierliche Abschwächung: Mit zunehmendem Abstand vom Fokusbereich nimmt der Kontrast ab. Mit freundlicher Genehmigung von Jason Le Vasseur. Quelle: YOUNG [2007]

plastischer erscheinen und erhöht gleichzeitig die Menge an vermittelter Information. Deshalb sollte darauf geachtet werden, den gesamten Kontrastumfang wie in Abb. 2.13 auszunutzen. Andere Gestaltungsprinzipien, beispielsweise Komposition, Balance, Harmonie und Einheit, werden in dieser Arbeit jedoch nicht weiter verfolgt. Zum einen ist anzunehmen, dass sie nur sehr schwer algorithmisch umzusetzen sind. Zum anderen hängen sie auch stark von Blickwinkel und Bildausschnitt ab. Gerade dies kann in einer interaktiven Anwendung aber nicht kontrolliert werden, da die Ansicht für den Benutzer frei wählbar ist.

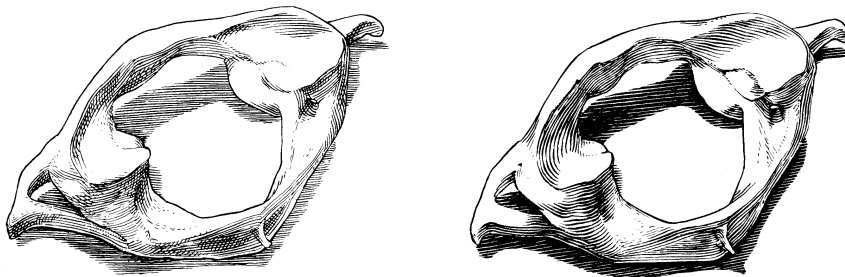


Abb. 2.13: Der stärkere Kontrast im rechten Bild verbessert den räumlichen Eindruck des Wirbelknochens. Schattenwurf auf die Auflagefläche steigert zusätzlich das Verständnis der Objektform. Quelle: HODGES [1989]

2.2 Computergestützte Illustration von Objekten

Im Anschluss an die Analyse werden nun Methoden betrachtet, mit deren Hilfe sich die ermittelten Parameter auf 3D-Objekten umsetzen lassen. Zunächst werden Parameter untersucht, die die Objekte als geometrische Körper visualisieren, d. h. ihre Form und Oberfläche auf ansprechende Weise darstellen und ihre Lage zueinander begreifbar machen.

2.2.1 Beleuchtungsmodelle

Dem Vorrang von Lichteinfall in der optischen Wahrnehmung entsprechend existieren viele mathematische Modelle, um realistisches Lichtverhalten zu berechnen. Gestalterisch ausgerichtete Beleuchtungsmodelle sind dagegen selten. Zwei dieser Modelle werden von AKERS et al. [2003] und HAMEL [2000] untersucht.

AKERS et al. [2003] entwickeln ein System, um aus mehreren Fotografien mit unterschiedlicher Beleuchtung aussagekräftige Bilder zu erstellen. Der Benutzer wählt dabei Bereiche aus den einzelnen Fotografien aus, die zu einem Gesamtbild kombiniert werden (siehe Abb. 2.14). Die Arbeit weist eine gute Analyse gestalterischer Beleuchtung auf, doch da das System nur auf Bildern basiert und manuelle Komposition verlangt, ist die technische Umsetzung für die 3D-Visualisierung nicht weiter hilfreich. Für die nicht-photorealistische Darstellung von 3D-Objekten schlägt HAMEL [2000] ein Lichtmodell vor, welches aus der gewichteten Summe mehrerer illustrativer Beleuchtungstechniken besteht. Beide Arbeiten beziehen sich hauptsächlich auf HODGES [1989] und verwenden daher ähnliche Beleuchtungstechniken wie die in Abschnitt 2.1.1 genannten.

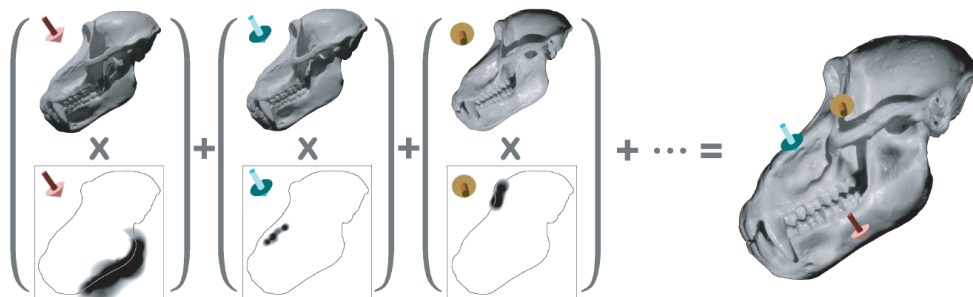


Abb. 2.14: Illustrative Lichtkomposition von AKERS et al. [2003]: Der Benutzer wählt mit einem Pinsel Bereiche von Fotografien aus, die automatisch in die Komposition einberechnet werden. Die Pfeile geben die jeweilige Lichtrichtung an.

Die Kunst des Illustrators besteht unter anderem darin, die jeweils passende Technik an den entsprechenden Stellen anzuwenden und dabei einen ausgewogenen Gesamteindruck zu erreichen. Es ist offensichtlich schwierig, diesen Vorgang komplett automatisch ablaufen zulassen. Zudem können an verschiedenen Stellen einer Illustration sehr unterschiedliche Techniken notwendig sein. Eine perfekte Kombination dieser Parameter wird daher schon für ein einzelnes 3D-Objekt kaum zu finden sein, geschweige denn für eine Szene aus mehreren Objekten. Es gibt jedoch verschiedene Möglichkeiten, die Beleuchtung so weit wie möglich zu optimieren.

Der Ansatz von HAMEL [2000], diese Techniken unabhängig voneinander umzusetzen und gewichtet aufzuaddieren, entspricht ihrer selektiven Anwendung in einer Illustration (siehe Abb. 2.15). Daher sollen zunächst Verfahren zur Realisierung der einzelnen Techniken betrachtet werden.

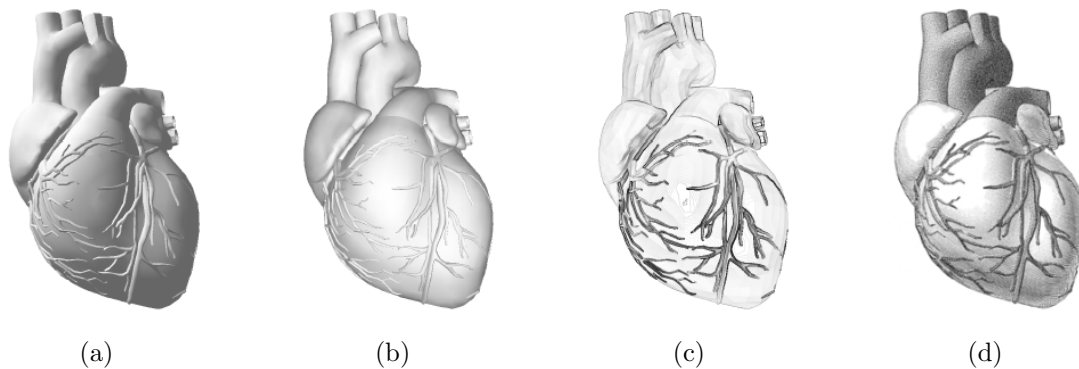


Abb. 2.15: Beispiel des Lichtmodells von HAMEL [2000]: Einzelne Beleuchtungstechniken Hauptlicht (a), Plateaulicht (b), Hervorhebung von Details mittels Krümmung (c). (d) ist eine gerenderte Illustration, deren Beleuchtung sich aus der gewichteten Summe der einzelnen Techniken ergibt.

Hauptlicht/Aufhellungslicht

Die einfachste Berechnung für das Hauptlicht ist die diffuse Beleuchtung, die nur vom Winkel zwischen Lichtrichtung und Oberflächennormale abhängig ist:

$$I = \max(0, \hat{l} \cdot \hat{n}) \quad (2.1)$$

Dabei ist \hat{l} der normalisierte Vektor vom betreffenden Oberflächenpunkt zur Lichtquelle, \hat{n} der Normaleneinheitsvektor an diesem Punkt. Handelt es sich um direktionales Licht, entspricht \hat{l} der umgekehrten Lichtrichtung. I ist die resultierende Intensität zwischen 0 (schwarz) und 1 (weiß). Nach dieser Gleichung sind alle lichtabgewandten Flächen schwarz, da Winkel von mehr als 90° ein negatives Skalarprodukt $\hat{l} \cdot \hat{n}$ ergeben. Ambiente Beleuchtung kann dies nicht ausgleichen, da sie die

Helligkeit konstant anhebt. Eine einfache Möglichkeit, diese Flächen aufzuhellen, besteht darin, auch negative Werte für den diffusen Term zuzulassen und die Intensität anschließend auf den Bereich $[0, 1]$ zu normalisieren. Die Gleichung dafür lautet

$$I = \frac{1}{2} + \frac{1}{2} \cdot (\hat{l} \cdot \hat{n}) \quad (2.2)$$

und entspricht einer Graustufen-Variante des Farbshadings, das in GOOCH et al. [1998] vorgestellt wird. Damit sind sowohl Hauptlicht als auch Aufhellung abgedeckt, ohne das Aufhellungslicht separat parametrisieren zu müssen.

Streifendes Licht

Simulation von streifendem Licht ist eine komplexere Aufgabe, da theoretisch für separate Bereiche der Oberfläche je ein lokales Licht gesetzt werden müsste. HAMEL [2000] simuliert streifendes Licht durch Zuhilfenahme von Oberflächenkrümmung, indem Bereiche mit hoher Krümmung eine dunklere Schattierung erhalten (siehe Abb. 2.15(c)). Dies stellt jedoch nur eine Annäherung dar. Ein einfaches Verfahren zur Berechnung von streifendem Licht wird von RUSINKIEWICZ et al. [2006] vorgestellt. Es verwendet diffuse Beleuchtung durch eine Hauptlichtquelle, deren Richtung lokal an die Oberfläche angepasst wird. Der diffuse Term wird unbeschränkt wie in Gleichung 2.2 verwendet, um auch lichtabgewandte Flächen ausreichend zu beleuchten. Zusätzlich kann der Helligkeitskontrast über einen „weichen Toon Shader“ übertrieben dargestellt werden:

$$I = \frac{1}{2} + \frac{1}{2} \cdot \underset{[-1,1]}{\text{clamp}}(a \cdot (\hat{l} \cdot \hat{n})) \quad (2.3)$$

Die Operation *clamp* beschneidet den diffusen Term wieder auf seinen normalen Wertebereich von $[-1, 1]$, nachdem er durch den Überzeichnungsfaktor a gespreizt wurde. Dadurch ist es möglich, die Stärke der Überzeichnung von Details zu regulieren (siehe Abb. 2.16).

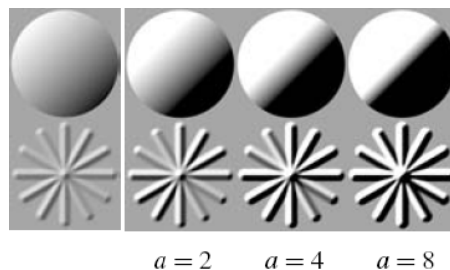


Abb. 2.16: Kontrastspreizung nach Gleichung 2.3 mit verschiedenen Werten für den Faktor a .
Quelle: RUSINKIEWICZ et al. [2006]

Die Normalen des Objektes werden in n Stufen geglättet und die Beleuchtung für jede Glättungsstufe einzeln berechnet. Die Anpassung der Lichtrichtung erfolgt dabei für Glättungsstufe i mittels der Normalen aus Stufe $i + 1$, indem die globale Lichtrichtung \hat{l}_{gl} auf eine zur geglätteten Normalen \hat{n}_{i+1} senkrechte Ebene projiziert wird (siehe Abb. 2.17). Diese Projektion erfolgt durch die Gleichung

$$l_{i+1} = \hat{l}_{gl} - \hat{n}_{i+1} \cdot (\hat{n}_{i+1} \cdot \hat{l}_{gl}) \quad (2.4)$$

So fällt das lokale Licht annähernd im 90° -Winkel auf die Oberfläche, unabhängig von ihrer Ausrichtung zum Hauptlicht. Für jeden Oberflächenpunkt wird aus der Normalen \hat{n}_i und der normalisierten lokalen Lichtrichtung \hat{l}_{i+1} die Intensität der streifenden Beleuchtung berechnet:

$$I_i = \frac{1}{2} + \frac{1}{2} \cdot \underset{[-1,1]}{\text{clamp}} \left(a \cdot (\hat{l}_{i+1} \cdot \hat{n}_i) \right) \quad (2.5)$$

Die Intensitäten I_1 bis I_{n-1} werden gewichtet aufaddiert. Durch die Gewichtung kann ausgewählt werden, ob größere oder feinere Details der Oberfläche betont werden (siehe Abb. 2.18). Dies wird im weiteren Verlauf als Detailfrequenz bezeichnet. Als Basis-Intensität dient unbeschränkte diffuse Beleuchtung auf der letzten Glättungsstufe n . Das Verhältnis zwischen I_n und der Summe aller lokalen Intensitäten bestimmt, wie stark der Eindruck einer globalen Lichtrichtung erhalten bleibt.

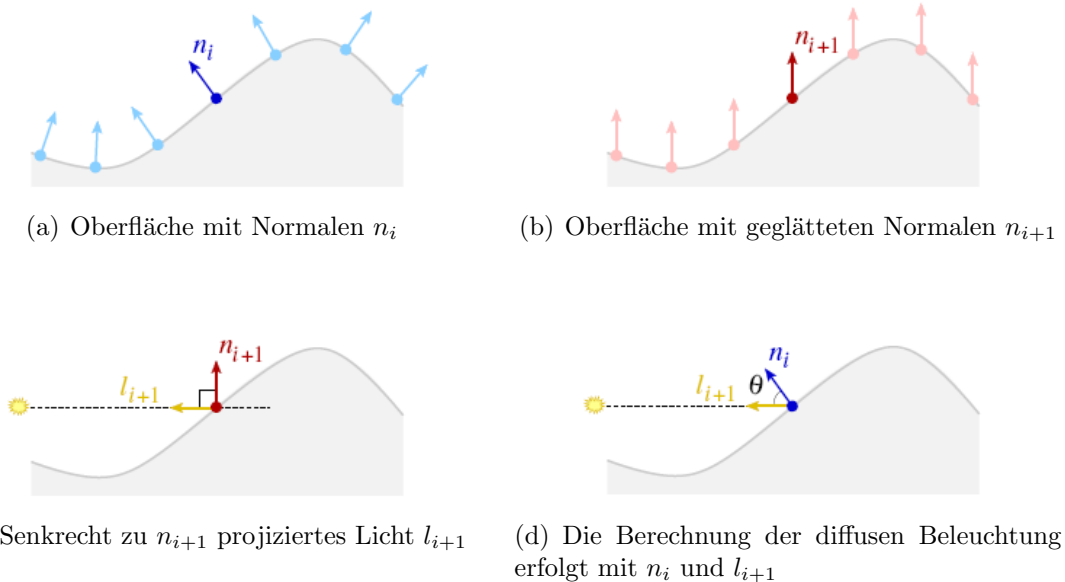


Abb. 2.17: Prinzip der Berechnung von streifendem Licht nach Gleichung 2.4 und 2.5. Alle Vektoren sind normalisiert. Quelle: RUSINKIEWICZ et al. [2006]



Abb. 2.18: Betonung verschiedener Detailfrequenzen durch entsprechende Gewichtung. Quelle: RUSINKIEWICZ et al. [2006]

Von CIGNONI et al. [2005] stammt ein weiteres Verfahren zur Betonung von Oberflächendetails. Im Gegensatz zur Normalenglättung wenden sie eine „Schärfung“ auf die Normalen an. Dadurch werden Kanten betont und ebene Flächen ungleichmäßig schattiert, was einen besseren Eindruck von glatten, flächigen Objekten vermittelt. Das Verfahren ist hauptsächlich für technische Modelle sinnvoll und wurde auch im Hinblick auf diese entworfen. Bei organischen Modellen, wie sie in der medizinischen Visualisierung vorkommen, ist der Effekt nicht sehr stark ausgeprägt.

Plateaulicht/Kantenlicht

Wie schon erwähnt, entsteht Plateaulicht bei Beleuchtung aus Betrachtungsrichtung und ist entsprechend realisierbar. Dabei treten keine Winkel $> 90^\circ$ auf, es kann also Gleichung 2.1 genutzt werden. HAMEL [2000] verwendet diese Methode und gibt an, dass Kantenlicht mit derselben Technik und negativer Gewichtung umsetzbar ist. Um so beide Techniken gleichzeitig einzusetzen, müsste der Einflussbereich des Kantenlichtes verringert werden. Dabei ist zu beachten, dass die Stärke dieses Effektes von der Objektform abhängt. Ein flaches Objekt wird größtenteils hell mit einem relativ schmalen dunklen Rand erscheinen.

Eine von der Objektform unabhängige Methode lässt sich aus der Arbeit von LUFT et al. [2006] entwickeln. Sie verwenden eine Variante der *Unschärfemaskierung*, um die Tiefenwirkung von 3D-Szenen zu steigern. Dabei wird der z -Buffer der Szene mit einem Gauß-Filter geglättet und die Differenz zum originalen z -Buffer gebildet (siehe Abb. 2.19). So entsteht eine Maske zur Identifikation von Tiefenunterschieden. Sie enthält Bildbereiche, in denen Vorder- und Hintergrundobjekte angrenzen und kann dazu verwendet werden, durch Veränderung der Helligkeit die räumliche Wirkung zu erhöhen (siehe Abb. 2.20). Abb. 2.21 zeigt einige der möglichen Manipulationen. Durch Aufhellen oder Abdunkeln von Objekträndern kann auch die Wirkung von Kantenlicht bzw. Plateaulicht erzeugt werden. Die Ausdehnung der betroffenen Bereiche ist über die Breite des Gauß-Filters einstellbar, hängt aber auch von der jeweiligen Tiefendifferenz ab.

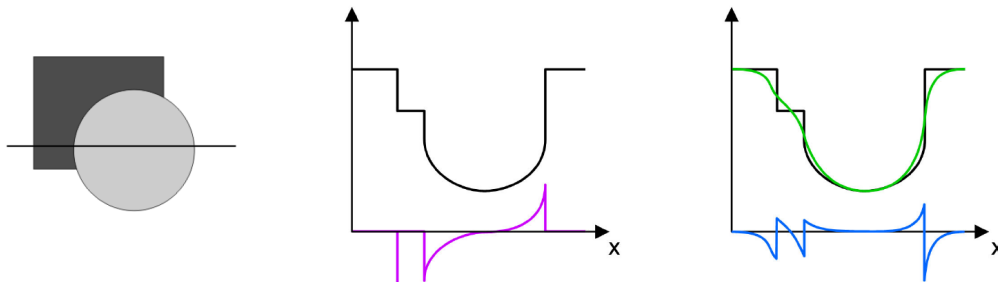


Abb. 2.19: Prinzip der Tiefenmaskierung. Links: Szene aus zwei Objekten mit Scanlinie. Mitte: Tiefenfunktion (schwarz) und ihre Ableitung (violett) entlang der Scanlinie. Rechts: Unschärf gefilterte Tiefenfunktion (grün) und die aus der Differenz resultierende Tiefenmaske (blau). Quelle: LUFT et al. [2006]

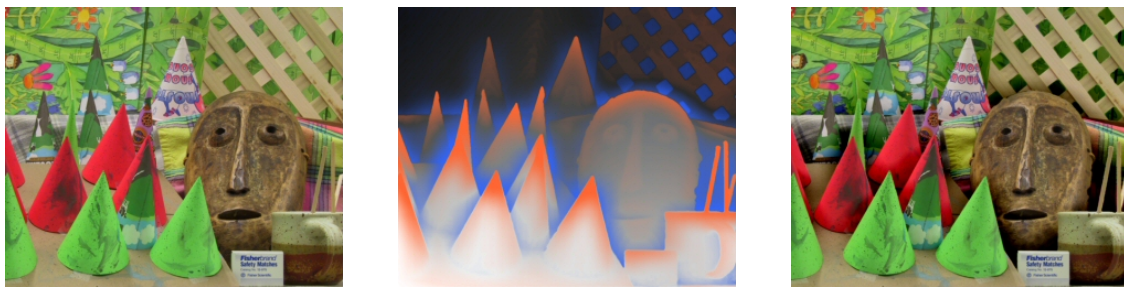


Abb. 2.20: Beispiel der Tiefenmaskierung. Links: Originalfoto, für das Tiefeninformation vorliegt. Mitte: In der Tiefenmaske können Randbereiche von Vordergrundobjekten (orange) und angrenzende Hintergrundbereiche (blau) unterschieden werden. Rechts: Hintergrundbereiche des Fotos wurden abgedunkelt. Quelle: LUFT et al. [2006]

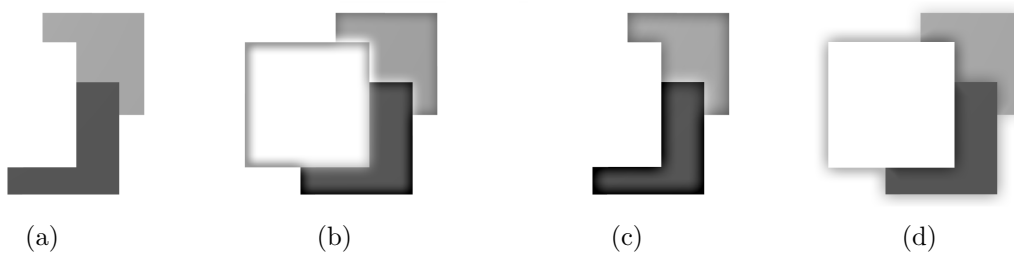


Abb. 2.21: Möglichkeiten zur Verstärkung der räumlichen Wirkung: (a) Originalszene. (b) Halos um Vordergrundobjekte. (c) Lokale Erhöhung des Kontrastes. (d) Abdunkeln des Hintergrundes. Quelle: LUFT et al. [2006]

Schattenwurf

Schattenwurf in Illustrationen dient meistens der Verankerung von Objekten in ihrer Umgebung, indem der Schatten auf eine Auflagefläche dargestellt wird (siehe Abb. 2.6(a), S. 8 und Abb. 2.13, S. 11). Eine separate Schattenebene wird z. B. von RITTER et al. [2001] verwendet, um in einer interaktiven Umgebung die räumliche Lage von Objekten zu verdeutlichen. Der Schattenwurf von Objekten auf sich selbst oder auf andere Objekte wird dagegen eher selten angewandt. Zudem wäre eine Diskussion realistischer Schattenwurfs im Rahmen dieser Arbeit nicht angemessen. Es existieren wie zur Beleuchtung eine Vielzahl an Techniken, zwei der bekanntesten darunter sind *Shadow Maps* und *Shadow Volumes*. Es wird auf den umfassenden Überblick von HASENFRATZ et al. [2003] verwiesen, dessen Schwerpunkt auf echtzeitfähigen Algorithmen liegt. Hier soll nur auf illustrative Anwendung von Schatten eingegangen werden, d. h. den in Abschnitt 2.1.3 genannten Überdeckungsschatten.

In der Arbeit von RITTER et al. [2006] wird Überdeckungsschatten verwendet, um den Tiefenabstand zwischen überschneidenden Gefäßästen zu visualisieren. Dazu wird zunächst die Originalszene gerendert, in einem zweiten Rendervorgang dagegen eine Variation, bei der alle Oberflächen vergrößert wurden. Die Differenz der Tiefenwerte beider Vorgänge bildet eine Maske, die jene Stellen markiert, an denen sich Objekte gegenseitig überlagern. Unter Beachtung des jeweiligen Tiefenabstandes der Objekte können diese Bereiche entsprechend abgedunkelt werden, wobei größerer Abstand durch breitere und hellere Schatten dargestellt wird (siehe Abb. 2.22). Dies bildet die reale Erscheinung nach, dass die Größe eines Schattens proportional mit der Entfernung der Projektionsfläche wächst und diffuser erscheint. Beides wird auch in Illustrationen als Tiefenhinweis eingesetzt.

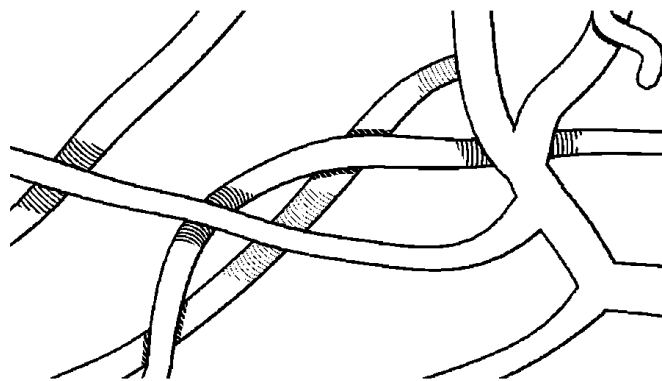


Abb. 2.22: Visualisierung des relativen Abstandes zwischen Gefäßästen. Größerer Tiefenabstand wird durch breitere, hellere Schatten dargestellt. Quelle: HANSEN [2006] (Grundlage von RITTER et al. [2006])

Wie in Abb. 2.21(d) gezeigt wird, kann mit der Methode von LUFT et al. [2006] ebenfalls der Eindruck von Überdeckungsschatten erzeugt werden. Da die Ausdehnung der maskierten Bereiche vom Tiefenabstand abhängt, erscheint der Schatten bei größerem Abstand zwischen Vorder- und Hintergrundobjekt breiter. Zusammen mit der im vorigen Abschnitt erwähnten Simulation von Kantenlicht kann mittels der Tiefenmaske eine ähnliche Darstellung wie in Abb. 2.9 (S. 10) erzielt werden.

Kombinierte Modelle

Eine Reihe von Arbeiten verfolgt den Ansatz, mehrere Lichtquellen halb- oder vollautomatisch so zu platzieren, dass einige oder alle der genannten Beleuchtungstechniken gleichzeitig realisiert sind. Kriterien sind dabei, dass alle Oberflächen ausreichende Beleuchtungsunterschiede aufweisen sowie Kanten und Silhouetten gut zu erkennen sind. Diese Arbeiten werden nun betrachtet.

SHACKED und LISCHINSKI [2001] entwickeln eine Funktion, mit deren Hilfe einer beleuchteten Szene ein Maß der Wahrnehmungsqualität zugeordnet werden kann. Unter anderem fließen Dynamikumfang, Schattierungsgradienten und Vorzugslichtrichtung in die Funktion ein. Position und Intensität mehrerer Lichter werden schrittweise optimiert, bis die Wahrnehmungsqualität ein Maximum erreicht. Die resultierende Beleuchtung ist speziell auf die aktuelle Ansicht optimiert, indem Kanten betont und Flächen mit ausreichender Variation schattiert werden, um ihre Form erkennbar zu machen. Ein ähnliches Ziel wird von GUMHOLD [2002] durch die Maximierung der Beleuchtungsentropie erreicht. Beide Verfahren sind jedoch ausdrücklich auf einzelne Ansichten ausgerichtet und benötigen teils umfangreiche Vorberechnungen, was sie für interaktive Betrachtung ungeeignet macht.

Die beiden vorigen Verfahren verwenden Lichter, die die gesamte Szene beleuchten, und optimieren daher global. Einen lokalen Ansatz entwerfen LEE und HAO [2006]: Das Objekt wird in Bereiche ähnlicher Krümmung unterteilt, für die jeweils eine optimale Lichtrichtung ermittelt und an den Bereichsgrenzen überblendet wird. Zusätzlich kann Überdeckungsschatten an überlappender Geometrie sowie Hervorhebung der Silhouette durch Plateaulicht hinzugefügt werden (siehe Abb. 2.23). Damit wird ein sehr detaillierter Oberflächeneindruck erreicht, was aber auf Kosten einer eindeutigen Hauptlichtrichtung geht. Diese Methode ist zwar zur interaktiven Anwendung gedacht, für Echtzeitfähigkeit aber noch zu berechnungsaufwändig. Auch ist nicht klar, ob bei wechselnder Betrachtungsrichtung eine kohärente Beleuchtung garantiert wäre.

Dies wird in der Arbeit von HALLE und MENG [2003] beachtet. Mit dem System LIGHTKIT setzen sie die klassische Drei-Punkt-Lichtführung aus Film und Fotografie um, bestehend aus Haupt-, Aufhellungs- und Kantenlicht. Das System stellt

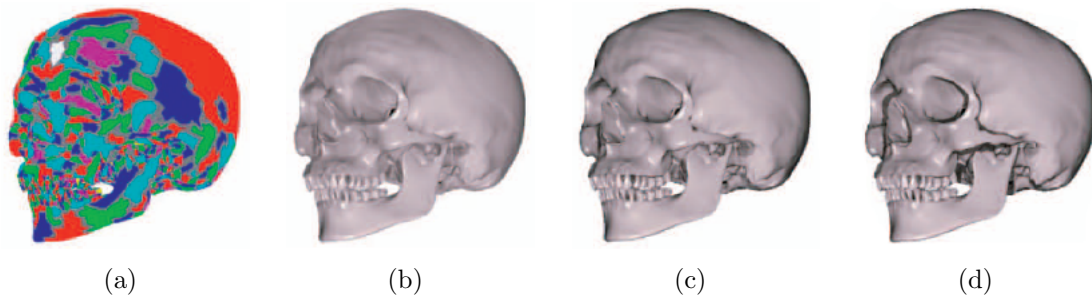


Abb. 2.23: Beispiel der automatischen Beleuchtung von LEE und HAO [2006]: (a) Segmentierung in Bereiche ähnlicher Krümmung. (b) Optimale Beleuchtung der Bereiche. (c) Zusätzlich Hervorhebung der Silhouette. (d) Zusätzlicher Einsatz von Überdeckungsschatten.

eine Hilfe zur manuellen Beleuchtung dar, die der Betrachter durch wenige Parameter regulieren kann. Die Lichter sind relativ zur Kamera positioniert, wodurch auf jeden Fall eine kohärente Beleuchtung gegeben ist. Im Gegensatz zu den vorigen Verfahren erfolgt keine spezielle Hervorhebung von Oberflächendetails, Kanten oder Silhouetten.

Ein halb-automatischer Ansatz aus dem Bereich des nicht-photorealistischen Renderings stammt von XIE et al. [2007]. In ihrer Arbeit definieren sie die *Photoc Extremum Lines*, eine neue Art von Merkmalslinien, die auf Unterschieden in der Beleuchtung basieren. Um die Lichtverhältnisse an das Objekt anzupassen, erlauben sie neben einem Hauptlicht die automatische Platzierung beliebig vieler lokaler Lichter, die jeweils in einem bestimmten Bereich den Kontrast maximieren oder minimieren. Diese Bereiche müssen jedoch vorher vom Benutzer ausgewählt werden.

2.2.2 Eigenschaften der Oberfläche

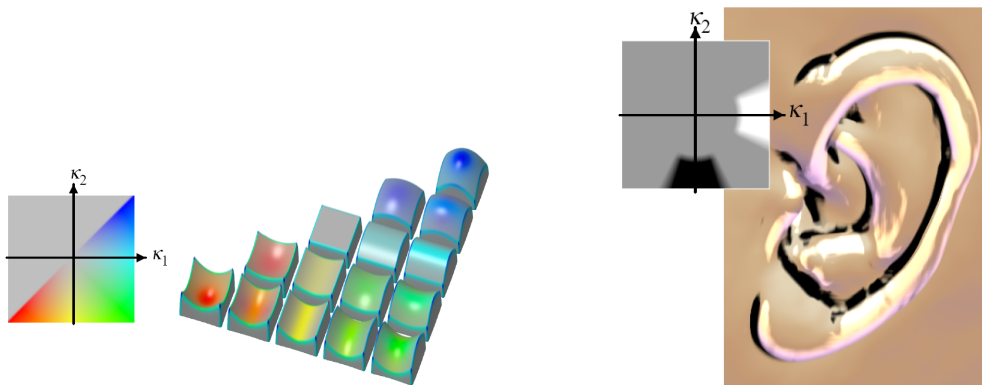
Als Nächstes werden Methoden betrachtet, die wie in Abschnitt 2.1.2 geometrische Eigenschaften der Oberfläche zur Visualisierung verwenden. Allen Ansätzen ist gemeinsam, dass sie unabhängig von Beleuchtung arbeiten. Wie bei Handzeichnungen werden sie jedoch vorwiegend zusätzlich zu Schattierung mit einer Hauptlichtquelle eingesetzt.

Krümmung

Eine wesentliche Eigenschaft von Oberflächen ist ihre Krümmung, da Gebiete mit hoher Krümmung in der Regel wichtiger für die Formerkennung sind als ebene Flächen. Dabei sind vor allem zwei Parameter relevant: die Stärke der Krümmung

und die Oberflächenform, z. B. konvexe oder konkave Krümmung. Eine grundsätzliche Auseinandersetzung mit Krümmung als Mittel der Schattierung nehmen HLADUVKA et al. [2000] vor, KINDLMANN et al. [2003] entwickeln die dort vorgestellten Ideen praktisch weiter. Beide Arbeiten beziehen sich auf Direktes Volumen-Rendering mittels Transferfunktionen, das Prinzip ist jedoch auch für Oberflächen-rendering verwendbar.

HLADUVKA et al. [2000] schlagen eine zweidimensionale Funktion vor, die jedem Punkt einer Oberfläche entsprechend seiner Hauptkrümmungen κ_1 und κ_2 eine Farbe zuordnet (siehe Abb. 2.24(a)). Dadurch ergeben sich verschiedene Möglichkeiten, sowohl Krümmungsstärke als auch Oberflächenform farblich zu kodieren. Im Fall einer Graustufenskala dagegen müssen beide Parameter getrennt betrachtet werden, da sie nicht gleichzeitig sinnvoll repräsentiert werden können. Für die Visualisierung der Oberflächenform in Graustufen geben KINDLMANN et al. [2003] eine Transferfunktion an, welche konvexe Gebiete heller und konkave Gebiete dunkler darstellt (siehe Abb. 2.24(b)). Dies wird mit der intuitiven Annahme begründet, dass Berge heller und Täler dunkler erscheinen müssen, wie auch in Abb. 2.6 (S. 8) gezeigt wurde.



(a) Transferfunktion und Beispiel der Abbildung von Krümmungsrichtung und -stärke auf Farbe.

(b) Transferfunktion und Beispiel der Hervorhebung von Bergen und Tälern. Grundlegende Schattierung nach GOOCH et al. [1998].

Abb. 2.24: Visualisierung von Krümmung bei KINDLMANN et al. [2003]. Es gilt $\kappa_1 \geq \kappa_2$ unter Beachtung des Vorzeichens. Ein positives Vorzeichen bezeichnet konvexe, ein negatives konkave Krümmung.

Wie erwähnt verwendet HAMEL [2000] Krümmung, um tangentialen Lichteinfall zu simulieren, wobei er auf die Krümmungsstärke zurückgreift. Um die für die Oberflächenstruktur interessanten Bereiche starker Krümmung hervorzuheben, werden sie bei ihm dunkel dargestellt, Bereiche schwacher Krümmung dagegen hell.

Weitere Merkmale

MILLER [1994] entwirft das *Accessibility Shading*, eine Schattierung auf dem Maß der Zugänglichkeit von Oberflächen. Die Zugänglichkeit wird ermittelt, indem an einen Punkt der Oberfläche eine Kugel mit maximalem Radius angepasst wird, die die Oberfläche nicht durchdringt (siehe Abb. 2.25). Je konkaver und enger die Umgebung des Punktes ist, desto kleiner der Radius und damit die Zugänglichkeit. Dieses Maß wird verwendet, um Bereiche mit geringerer Zugänglichkeit dunkel zu schattieren. Der Ansatz des *Depth Shading*, der von KUMAR et al. [2003] verwendet wird, hat ein ähnliches Ziel. Hier wird die Entfernung eines Punktes von einer die Umgebung annähernden Fläche als lokale Tiefe interpretiert und mit dunklerer Schattierung versehen.

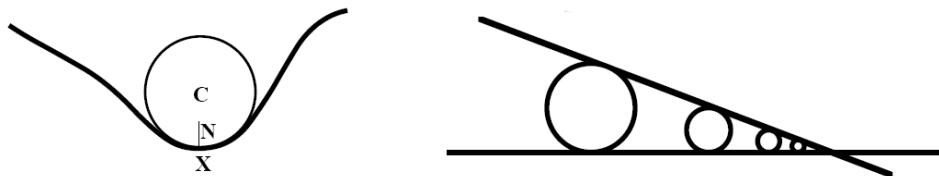


Abb. 2.25: Prinzip der Messung von Zugänglichkeit: Anpassung einer Kugel mit maximalem Radius, die Punkt X berührt (links). Der abnehmende Radius von Kugeln zwischen zwei Flächen entspricht abnehmender Zugänglichkeit (rechts). Quelle: MILLER [1994]

Beide Konzepte sind sinnvoll für die Visualisierung großflächiger Objekte mit engen Vertiefungen wie die in Abb. 2.26 gezeigten Keilschrifttafeln, wofür sie mehrfach angewendet wurden [ANDERSON und LEVOY, 2002; COHEN et al., 2004]. Insgesamt ergeben sie ein ähnliches Resultat wie die Betonung von Bergen und Tälern bei KINDLMANN et al. [2003]. Tatsächlich benutzen HADWIGER et al. [2005] die Oberflächenkrümmung, um *Accessibility Shading* zu approximieren. Aus diesem Grund wird nur die Visualisierung der Krümmung weiter betrachtet.



Abb. 2.26: Zwei Beispiele der Visualisierung von Keilschrifttafeln mit *Accessibility Shading* (links) und *Depth Shading* (rechts). Quelle: COHEN et al. [2004]

2.2.3 Atmosphärische Perspektive

Von FOLEY et al. [1995] wird das einfache Prinzip aufgestellt, atmosphärische Perspektive als lineare Interpolation zwischen dem originalen Farbwert und einer Hintergrundfarbe zu modellieren. Reduziert auf die Intensität ist dies mit folgender Gleichung möglich:

$$I' = (1 - d) \cdot I + d \cdot I_b \quad (2.6)$$

Dabei ist I die ursprüngliche Intensität, I_b die des Hintergrundes und d der Tiefenwert zwischen 0 (nah) und 1 (fern). Bei klassischer atmosphärischer Perspektive wäre die Hintergrundintensität I_b weiß, doch es ist auch möglich, eine andere Intensität als Zielwert zu verwenden, wie das mittlere Grau in Abb. 2.7 (S. 9).

Die tatsächliche Tiefe d_o , die bei üblichen Renderverfahren aus dem z -Buffer ausgelesen werden kann, wird dabei vereinfacht zwischen den Minimal- und Maximalwerten d_n und d_f interpoliert. Dadurch kann der Einflussbereich auf einen Teil der Szene beschränkt werden. Dies ist sinnvoll, da atmosphärische Perspektive üblicherweise nicht gleichförmig auf die gesamte Szene angewendet wird, sondern nur den Hintergrund betrifft. EBERT und RHEINGANS [2000] führen zusätzlich einen Exponenten f ein, mit dem ein nichtlinearer Anstieg des Effektes gesteuert werden kann. Beide Ansätze lassen sich zu folgender Berechnung des Tiefenwertes d aus Gleichung 2.6 kombinieren:

$$d = \begin{cases} 0, & \text{wenn } d_o < d_n \\ 1, & \text{wenn } d_o > d_f \\ \left(\frac{d_o - d_n}{d_f - d_n}\right)^f & \text{sonst} \end{cases} \quad (2.7)$$

2.2.4 Kontrastverbesserung

Um wie in Abb. 2.13 (S. 11) gezeigt den maximalen Kontrastumfang auszunutzen, können die Parameter einzeln oder als Summe entsprechend manipuliert werden. Dies ist am besten über Operationen auf dem Intensitätshistogramm zu erreichen. Zwei grundlegende Methoden dafür sind Histogrammspreizung und Histogramm-äqualisation [STEINBRECHER, 1993]. Bei beiden Verfahren liegen die resultierenden Intensitäten im Intervall $[0, 1]$, unabhängig vom ursprünglichen Wertebereich.

Histogrammspreizung

Bei der Histogrammspreizung werden Minimum und Maximum der Helligkeitsverteilung ermittelt und sämtliche Intensitätswerte so transformiert, dass der gesamte Bereich zwischen schwarz und weiß ausgenutzt wird. Mit I_{\min} und I_{\max} als Minimum

bzw. Maximum ergibt sich die neue Intensität I' aus der ursprünglichen Intensität I nach folgender Gleichung:

$$I' = \frac{I - I_{\min}}{I_{\max} - I_{\min}} \quad (2.8)$$

Dabei bleibt die ursprüngliche Verteilung der Werte proportional erhalten. In einer interaktiven Darstellung können die Extremwerte beim Verändern der Ansicht jedoch stark schwanken, die Spreizung daher von einem Frame zum nächsten sehr unterschiedliche Ergebnisse liefern. Außerdem genügt ein einziger Pixel mit einem sehr hohen oder niedrigen Wert, um die resultierende Verteilung stark zu verzerren. Um dies zu verhindern, können die Extremwerte so gewählt werden, dass ein bestimmter Prozentsatz von Pixeln unter bzw. über ihnen liegt, beispielsweise das 5. Perzentil des Histogramms als Minimum, das 95. Perzentil als Maximum. Dadurch wird der Einfluss von Ausreißern abgeschwächt und der Übergang zwischen einzelnen Frames geglättet.

Histogrammäqualisation

Die Histogrammäqualisation strebt eine Gleichverteilung aller Intensitäten an. Dazu wird jeder Grauwert auf seine relative Häufigkeit abgebildet, die aus dem kumulativen Histogramm hervorgeht. Mit N als Anzahl aller Pixel und N_I als Anzahl derjenigen Pixel, deren Intensität kleiner oder gleich I ist, wird die Histogrammäqualisation mit folgender Gleichung ausgedrückt:

$$I' = \frac{N_I}{N} \quad (2.9)$$

Im Gegensatz zur Spreizung bleibt hier die Relation der Grauwerte nicht erhalten, stattdessen werden sie annähernd gleichmäßig über den Wertebereich verteilt. Das Verfahren ist stabiler gegenüber Änderungen und Ausreißern, kann jedoch ein unbefriedigendes Ergebnis liefern, falls ein Grauwert im Originalbild besonders häufig vorkommt. Da jeder Grauwert auf nur einen anderen Grauwert abgebildet wird, bleibt diese Anhäufung erhalten und verzerrt die umliegenden Werte. Um dies zu vermeiden, kann das Verfahren adaptiv angewandt werden, indem die Transformation jeweils für kleinere Bildabschnitte durchgeführt wird. So wird jeweils lokal der Kontrast optimiert, ohne dass das globale Vorherrschen eines Grauwertes das Gesamtergebnis beeinflusst.

Histogrammerzeugung

Die Erzeugung eines Histogramms mittels CPU-Programmierung ist trivial. Es entsteht jedoch ein Engpass in der Renderpipeline, wenn ein auf der Grafikkarte er-

zeugter Frame erst von der CPU ausgelesen werden muss. Es wäre sinnvoll, das Histogramm auch auf der Grafikkarte zu erzeugen. Da sich die GPU-Programmierung nicht für globale Operationen auf größeren Datenstrukturen eignet, können herkömmliche Methoden jedoch nicht direkt umgesetzt werden.

Frühere Ansätze zur GPU-basierten Erzeugung von Histogrammen verwenden mehrere Rendervorgänge. Bei GREEN [2005] werden pro Vorgang alle Pixel in einem bestimmten Helligkeitsintervall gezählt, für ein Histogramm mit N Klassen sind also pro Frame N Vorgänge nötig. MITCHELL et al. [2006] erhöhen die Effizienz, indem pro Frame nur eine Histogrammkategorie aktualisiert wird, wodurch aber das Histogramm nie genau den aktuellen Frame repräsentiert. Durch Aufteilen des Frames in kleinere Bereiche und Berechnung lokaler Histogramme erreichen FLUCK et al. [2006] eine Reduktion der benötigten Rendervorgänge auf $\log N$.

SCHEUERMANN und HENSLEY [2007] stellen eine Methode vor, die nur einen Vorgang für ein komplettes Histogramm benötigt. Dazu verwenden sie *Pixel Buffer Objects*, eine Erweiterung neuerer Grafik-Hardware, mit der eine auf der Grafikkarte befindliche Textur als Vertexliste interpretiert werden kann. So entsprechen die Koordinaten eines Vertex jeweils der Farbe eines Pixels. Diese Vertexliste wird der Renderpipeline übergeben und in eine Textur der Dimension $N \times 1$ gerendert, wobei jeder Ausgabepixel einer der N Histogrammklassen entspricht. Im Vertexshader wird für jeden Vertex der Grauwert und die Histogrammkategorie ermittelt. Daraufhin erhält er neue Koordinaten, die ihn als Fragment auf den entsprechenden Pixel der Ausgabertextur abbilden. Dort wird durch additives Blending die Anzahl der Fragments aufsummiert. Die Ausgabertextur enthält anschließend das Histogramm der Eingabetextur und kann als Eingabe für einen weiteren Rendervorgang benutzt werden.

SCHEUERMANN und HENSLEY [2007] zeigen, dass ihr Algorithmus schneller als die vorgenannten läuft und sich sogar mit zunehmender Anzahl an Klassen beschleunigt, was auf die geringere Anzahl an Blending-Operationen zurückzuführen ist. Aufgrund der verfügbaren Texturformate ergeben sich jedoch Probleme mit der Genauigkeit, die in Abschnitt 4.3.3 angesprochen werden.

2.2.5 Merkmalslinien

Der Einsatz von Silhouetten in Zeichnungen ist praktisch selbstverständlich, weshalb er in Abschnitt 2.1 nicht gesondert genannt wurde. Ebenso üblich ist es, wichtige innere Kanten auf gleiche Art hervorzuheben. Der Einfachheit halber werden Silhouetten und innere Kanten im weiteren Verlauf unter dem Begriff *Merkmalslinien* zusammengefasst. Genaugenommen handelt es sich hierbei nicht um Schattierung, da Merkmalslinien üblicherweise als durchgezogene Linien und nicht mittels

Punkten bzw. Schraffurlinien dargestellt werden. Ob diese zeichnerischen Elemente für die Liniendarstellung auch ausreichen, wird im weiteren Verlauf untersucht, weshalb Merkmalslinien als Schattierungsparameter aufgenommen werden.

Die Extraktion von Merkmalslinien aus Oberflächenmodellen ist ein weites Gebiet, dass sich grob in objekt- und bildbasierte Methoden einteilen lässt. Eine grundlegende bildbasierte Technik, das Finden von Diskontinuitäten im Bildraum, wurde von SAITO und TAKAHASHI [1990] eingeführt. Dies kann mit üblichen Kantenfiltern durchgeführt werden und ist von der Geometrie unabhängig. Vereinfacht ergeben sich dabei Silhouetten aus der Filterung des Objekt-Buffers und alle Merkmalslinien aus der des z -Buffers.

Objektbasierte Methoden liefern in der Regel genauere Ergebnisse, sind dafür aber langsamer und anfällig für Fehler in der Geometrie. Vor- und Nachteile der verschiedenen Ansätze werden von ISENBERG et al. [2003] aufgeführt. Da die Merkmalslinien in dieser Arbeit in Echtzeit berechnet werden sollen und keine hohe Präzision verlangen, ist die Methode von SAITO und TAKAHASHI [1990] ausreichend. Für die gezielte Stilisierung der Linien sind bildbasierte Techniken nicht geeignet, dies ist jedoch auch nicht das Ziel dieser Arbeit.

Merkmalslinien können ebenfalls zur Verdeutlichung von Tiefe eingesetzt werden, indem ihre Stärke analog zur atmosphärischen Perspektive mit zunehmender Tiefe abnimmt. Dies wird z. B. von GOOCH et al. [1999] angewandt.

2.3 Szeneninformationen

Szeneninformationen sind alle Informationen, die sich nicht direkt aus dem Objekt selbst ergeben, sondern zusätzlich vermittelt werden, beispielsweise die in Abschnitt 2.1.4 genannte Wichtigkeit von Objekten oder ihre Beziehung zueinander. Somit sind sie in der Realität nicht am Objekt beobachtbar und dienen nicht der realistischen Darstellung, sondern stehen ihr möglicherweise sogar entgegen. Aus diesem Grund kann hier von der Kodierung von Informationen gesprochen werden.

In klassischen Illustrationen treten zusätzliche Informationen eher selten auf, daher gibt es meistens keine verbindliche Form der Kodierung. So ist es dem Illustrator überlassen, eine möglichst gut begreifbare Visualisierung zu finden. Es werden nun Arbeiten betrachtet, die verschiedene Methoden der Visualisierung verwenden, und auf ihre Verwendung speziell in Stippling- oder Hatching-Illustrationen untersucht. Da es theoretisch eine beliebige Anzahl solcher Informationen gibt, sollen nur wenige ausgewählte Beispiele betrachtet werden.

2.3.1 Wichtigkeit

Es existiert eine Anzahl von Ansätzen zur Hervorhebung von Objekten, die an verschiedenen Stellen der Visualisierung angreifen, z. B. durch Verzerrung der Geometrie, Einführung von Tiefenschärfe oder Variation der Auflösung. Ansätze, die die Objektgeometrie oder den Rendervorgang direkt manipulieren, werden hier nicht weiter erläutert, da dies nicht das Ziel dieser Arbeit ist. Für die Hervorhebung und Betonung mit illustrativen Mitteln lieferten VIOLA et al. [2004] und TIETJEN et al. [2005] erste Anstöße, eine Weiterentwicklung stammt von SALAH et al. [2006].

VIOLA et al. [2004] schlagen vor, Objekten einen Wichtigkeitswert zuzuweisen und ihre Darstellung entsprechend zu modifizieren, um so die Aufmerksamkeit des Betrachters auf wichtige Strukturen zu lenken. Dazu stellen sie das Konzept der *Level of Sparseness* auf, die beschreiben, wie viel optisches Gewicht einem Objekt in einer Illustration zufällt. Unterschiedliche Levels of Sparseness werden erreicht durch Veränderung der Farbsättigung, echte und simulierte Transparenz sowie spezielle Techniken des Volumen-Renderings, auf dem die Arbeit basiert (siehe Abb. 2.27).

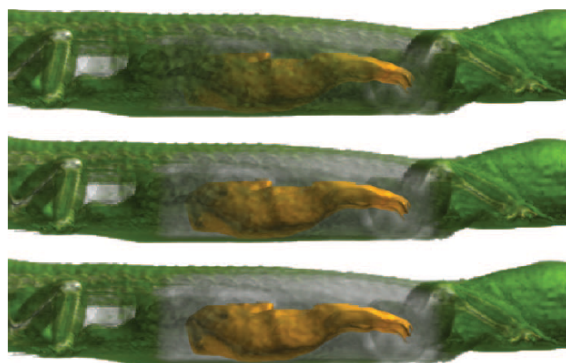


Abb. 2.27: Visualisierung von Wichtigkeit: Die äußere Struktur erhält durch zunehmende Transparenz und abnehmende Sättigung ein geringeres Level of Sparseness, die innere Struktur dagegen tritt stärker hervor. Quelle: VIOLA et al. [2004]

TIETJEN et al. [2005] sortieren Objekte nach ihrer Wichtigkeit in drei Klassen: Fokusobjekte, denen die Aufmerksamkeit gilt, fokusnahe Objekte, die für das Verständnis des Fokusobjektes wichtig sind, und Kontextobjekte, die nur den gesamten räumlichen Zusammenhang darlegen. In PREIM et al. [2005] wird darauf aufbauend zusätzlich zwischen Betonung und Fokussierung unterschieden. Fokussierung ist demnach eine stufenlose Bewertung von Objekten nach ihrer Wichtigkeit, die vom jeweiligen Zweck der Illustration abhängt. Betonung unterscheidet nur, ob eine Struktur zusätzlich hervorgehoben wird, beispielsweise um ein vom Benutzer ausgewähltes Objekt zu markieren. Im Gegensatz zu VIOLA et al. [2004] verwenden TIETJEN et al. [2005] zusätzlich zu Volumen- auch Oberflächenrendering sowie

Silhouetten und innere Kanten, um verschiedene Levels of Sparseness zu erreichen (siehe Abb. 2.28). So werden z. B. Kontextobjekte nur mittels ihrer Silhouette, Fokus- und fokusnahe Objekte dagegen zusätzlich durch Illustration der Oberfläche dargestellt, ähnlich wie in Abb. 2.11 (S. 11).

Die von SALAH et al. [2006] entworfene Visualisierungs-Pipeline erlaubt es, jedes Objekt durch eine eigene Kombination mehrerer Rendertechniken und Schattierungsstile darzustellen. Als sinnvolle Einschränkung der Vielzahl an Kombination schlagen sie eine ähnliche Einteilung der Objekte vor wie TIETJEN et al. [2005]. Beide Arbeiten geben die Möglichkeit an, Wichtigkeit durch variierende Stärke der Silhouette darzustellen. Bei SALAH et al. [2006] kann sie für Kontextobjekte auch ganz entfernt werden, was natürlich nur bei gleichzeitiger Illustration der Oberfläche möglich ist (siehe Abb. 2.29).

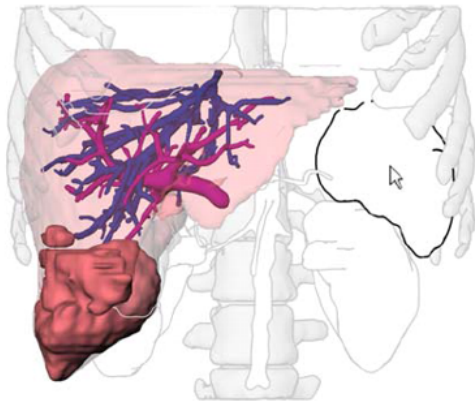


Abb. 2.28: Fokuskategorien bei TIETJEN et al. [2005]: Kontext (nur Silhouette), fokusnahe Objekte (zusätzlich Schattierung) und Fokusobjekte (zusätzlich Farbe). Das vom Nutzer mit der Maus gewählte Objekt wird durch eine stärkere Silhouette betont.

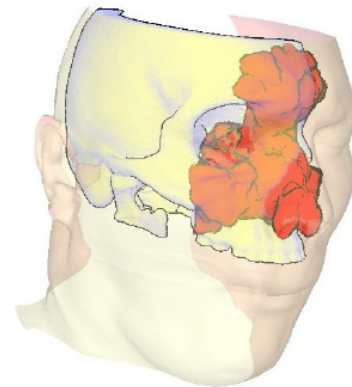


Abb. 2.29: Fokuskategorien bei SALAH et al. [2006]: Die Haut als Kontextobjekt wird schattiert, aber ohne Silhouette dargestellt.

Zwei der Parameter aus den genannten Arbeiten, Sättigung und Transparenz, werden nun etwas näher betrachtet. Gesättigte Farben ziehen die Aufmerksamkeit auf sich, wodurch sich Sättigung gut zur Illustration von Wichtigkeit eignet [VIOLA et al., 2004; SALAH et al., 2006]. Bei Verwendung von Grautönen fällt Sättigung mit Helligkeit zusammen, daher ist diese nicht separat anwendbar. Es ergeben sich zwei Möglichkeiten, Wichtigkeit durch Helligkeit zu kodieren: Zum einen kann bei einem schattierten Objekt der Kontrast mit abnehmender Wichtigkeit verringert werden wie in Abb. 2.12 (S. 11). Dies hat eine der Entsättigung ähnliche Wirkung und wird auch als Stilmittel in HODGES [1989] aufgeführt. Zum anderen kann die Wichtigkeit direkt als Helligkeit kodiert und als eigenständiger Schattierungsparameter betrachtet werden.

Alle drei Arbeiten verwenden Transparenz, um unwichtige Objekte optisch abzuschwächen. Zusätzlich wird dadurch der Blick auf verdeckte wichtigere Strukturen freigegeben. Transparenz als Gestaltungsmittel wird in klassischen medizinischen Illustrationen eher selten und sehr selektiv verwendet, kommt aber durchaus vor. Bei reinen Schwarz-Weiß-Zeichnungen ist zu bedenken, dass die Unterscheidung zwischen halbtransparenten Objekten schwierig sein könnte, da nicht notwendigerweise klar ist, welche der Zeichenelemente zu welchem Objekt gehören. In einer interaktiven Visualisierung tritt dieses Problem eventuell weniger auf: Beim Verändern der Sichtposition werden die entsprechenden Zeichenelemente dem Gesetz der gemeinsamen Bewegung zufolge als zusammengehörig wahrgenommen.

2.3.2 Distanz

Eine weitere Klasse von Szeneninformationen beinhaltet die Darstellung von Distanz. Da in zweidimensionalen Illustrationen räumliche Verhältnisse nicht immer korrekt zu erkennen sind, kann es sinnvoll sein, wichtige Distanzangaben zusätzlich darzustellen. Im technischen Bereich geschieht dies oft durch Maßangaben, die direkt in die Zeichnung eingetragen werden. Diese Methode ist zwar am genauesten, erfordert aber auch eine höhere Aufmerksamkeit des Betrachters. Wünschenswert ist daher die Kodierung von Distanz durch illustrative Mittel, was zwar kein Ablesen absoluter Werte, aber eine direktere Wahrnehmung ermöglicht.

In klassischen Handzeichnungen kommt die Visualisierung von Abständen generell nicht vor, da diese in allgemeinen anatomischen Illustrationen nicht von großer Bedeutung sind. Bei einer Operation und ihrer Vorbereitung kann es dagegen sehr wichtig sein, den genauen Abstand zwischen einer bestimmten Struktur und ihrer Umgebung zu kennen. Basierend auf HANSEN [2006] werden in der Arbeit von RITTER et al. [2006] mehrere Methoden zur Visualisierung von Abständen in Gefäßbäumen diskutiert. Sie haben das Ziel, während einer Operation die Struktur innerer Gefäße direkt auf das betroffene Organ zu projizieren. Dabei werden wichtige Informationen über die Gefäße kodiert, die für den Operierenden hilfreich sind. Da die Organoberfläche keine gute Farbprojektion zulässt, können hierfür nur schwarz-weiße Darstellungen verwendet werden. Bei der Kodierung von Abständen behandeln sie zwei Kategorien, die nun genauer ausgeführt werden.

Abstand zu einem Objekt

Im Gegensatz zum Tiefenabstand, der jeweils zwischen zwei Objekten und nur entlang der Tiefe bestimmt wird, handelt es sich hier um den absoluten Abstand eines Objektes zu allen anderen, gemessen in drei Dimensionen. RITTER et al.

[2006] betrachten das Beispiel von Gefäßbäumen, auf denen durch Veränderung der Textur der Abstand zu einem Tumor kodiert wird. Unter anderem schlagen sie vor, die Dichte von Punkten oder Linien auf einer Textur zu verändern, was der Variation von Helligkeit bei Stippling und Hatching entspricht (siehe Abb. 2.30). Wie auch bei der Visualisierung von Wichtigkeit wäre es möglich, den Abstand über eine Verringerung des Kontrastes darzustellen. Dies könnte als Fokussierung mit kontinuierlichem Wert betrachtet werden, bei der die Wichtigkeit mit größerem Abstand abnimmt (siehe Abb. 2.12, S. 11). Eine solcher Ansatz wird auch von ZHOU et al. [2004] verwendet, um Volumendaten mit zunehmendem Abstand von einer festgelegten Fokusregion auszublenden.

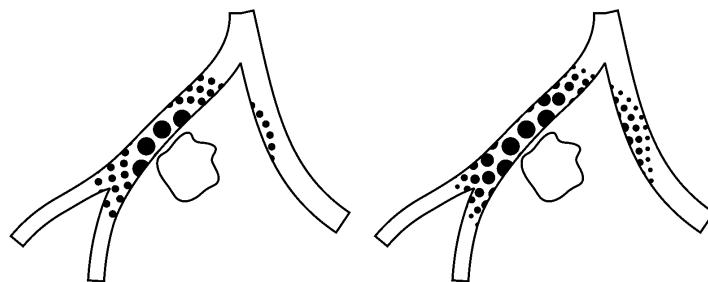


Abb. 2.30: Visualisierung des Abstandes von einer zentralen Struktur mittels Texturdichte: Links ist der Abstand in diskreten Schritten, recht kontinuierlich dargestellt. In einer dreidimensionalen Szene wird entsprechend der räumliche Abstand verwendet. Quelle: RITTER et al. [2006]

Abstand zum Betrachter

Die zweite Kategorie beinhaltet die Entfernung vom Betrachter. Vereinfacht handelt es sich dabei um die Tiefe der Szene, die im z -Buffer abzulesen ist, es kann aber auch explizit der Abstand zwischen Betrachterposition und Objektpunkt berechnet werden. RITTER et al. [2006] variieren für die Darstellung ebenfalls Textureigenschaften, die sich als Veränderung der Helligkeit äußern, wobei nah als schwarz und fern als weiß dargestellt wird (siehe Abb. 2.31). Dies ergibt eine direkte Kodierung von Tiefe durch Helligkeit wie in Abb. 2.8 (S. 9), hier ist die Zuordnung der Helligkeit jedoch genau umgekehrt.

Die von RITTER et al. [2006] gewählte Kodierung ähnelt der atmosphärischen Perspektive, die in Abschnitt 2.2.3 behandelt wurde. Die auch hier anwendbare Variante, statt der Helligkeit den Kontrast zu verändern, wäre ebenfalls ähnlich zu diesem Effekt. Beide Verfahren werden im weiteren Verlauf dennoch getrennt betrachtet, da die atmosphärische Perspektive nur einen illustrativen Eindruck von Räumlichkeit vermitteln, der Abstand zum Betrachter dagegen die korrekte Tiefe kodieren soll. Er unterliegt damit bei der Visualisierung einer strengeren Behandlung, wogegen die atmosphärische Perspektive mit größerer Freiheit angewandt werden kann.

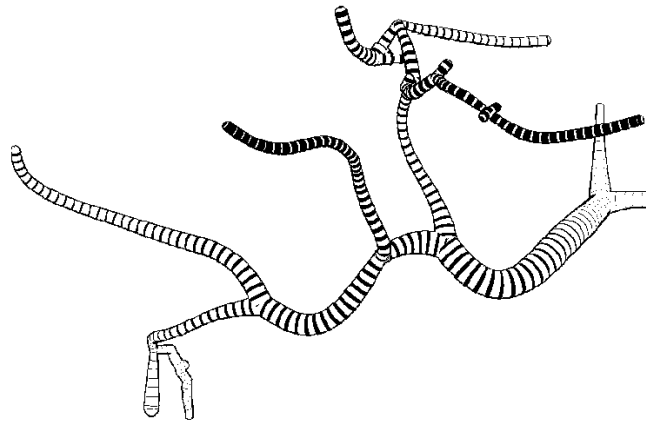


Abb. 2.31: Visualisierung der Szenentiefe: Dunklere Gefäßäste liegen näher am Betrachter. Quelle: RITTER et al. [2006]

2.3.3 Weitere Informationen

Generell können beliebige weitere Szeneninformationen kodiert werden, beispielsweise Objekttyp, Schweregrad einer Erkrankung oder Verzweigung von Gefäßbäumen. Auf ihre Besprechung wird hier aus zwei Gründen verzichtet: Zum einen beziehen sie sich teils auf besondere Anwendungsfälle oder Objekttypen und sind somit nicht allgemein anwendbar. Zum anderen wäre ihre Visualisierung mittels Schattierung analog zu den zuvor beschriebenen Informationen. Da sich schon die Visualisierungstechniken von Wichtigkeit, Abstand und Tiefe überschneiden, ist es fraglich, ob und wie viele dieser Informationen gleichzeitig sinnvoll dargestellt werden können. Für weitere Informationen sollte daher wohl eher auf alternative Visualisierungsparameter wie Farbe ausgewichen werden.

2.4 Zusammenfassung

In diesem Kapitel wurde eine grundlegende Analyse von Parametern zur Schattierung durchgeführt. Dazu wurden Stippling- und Hatching-Illustrationen aus medizinischen Atlanten auf Gebrauch von Schattierung hin untersucht. Es konnten einzelne Parameter extrahiert werden, die überwiegend der verständlichen und anschaulichen Darstellung von anatomischen Strukturen dienen. Die Nachbildung realer Lichtsituationen, wie von HODGES [1989] beschrieben, steht dabei im Vordergrund. Diese werden jedoch von Illustratoren auf eine Weise kombiniert, die in der Realität nicht möglich ist. Weitere Anwendung von Schattierung besteht in der Verdeutlichung räumlicher Verhältnisse, was durch atmosphärische Perspektive

und illustrativen Schattenwurf erreicht werden kann, sowie der Hervorhebung und Abschwächung von Teilen der Illustration.

Anschließend wurden Arbeiten und Algorithmen betrachtet, die die Umsetzung der ermittelten Parameter auf 3D-Modellen ermöglichen. Zunächst wurden Methoden der allgemeinen Computergrafik untersucht. Diese setzten häufig die analysierten Parametern direkt um, da die entsprechenden Arbeiten sich ebenfalls auf Handzeichnungen als Grundlage beziehen.

Die Betrachtung der Arbeiten ergab, dass die interaktive und illustrative Schattierung dreidimensionaler Objekte selten behandelt wurde. Bisherige Ansätze, die mehrere Beleuchtungstechniken kombinieren, arbeiten entweder im zweidimensionalen Bereich [AKERS et al., 2003], sind nicht für interaktive Betrachtung konzipiert [SHACKED und LISCHINSKI, 2001; GUMHOLD, 2002] oder nicht hinreichend anpassbar [LEE und HAO, 2006]. Der Ansatz von HAMEL [2000] kommt dem am nächsten, da dort Beleuchtungstechniken einzeln umgesetzt werden, wodurch sie frei kombinierbar sind. Die vorliegende Arbeit geht jedoch darüber hinaus: zum einen durch eine größere Anzahl an Parametern mit unterschiedlicher Funktion, zum anderen durch den Schwerpunkt, der auf interaktive Betrachtung und einfache Parametrisierung gelegt wird.

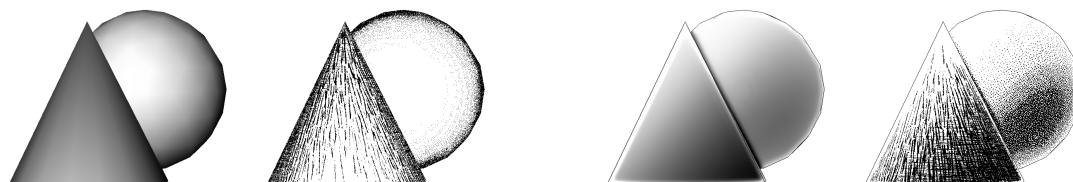
Eine weitere Klasse von Parameter ergibt sich aus Arbeiten der medizinischen Visualisierung, die die Vermittlung zusätzlicher Informationen über die gesamte Szene oder einzelne Objekte zum Ziel haben. Abgesehen von der Hervorhebung wichtiger Objekte sind solche Mittel der Visualisierung in traditionellen Illustrationen selten anzutreffen. Für ihre Umsetzung sind daher die untersuchten Arbeiten ausschlaggebend. Die dort verwendeten Methoden müssen aber dahingehend angepasst werden, dass Vermittlung von Informationen allein über Veränderung der Helligkeit stattfindet.

Auf Basis der ermittelten Parameter und ihrer praktischen Umsetzung wird nun ein Konzept entwickelt, mit dem das Ziel einer interaktiven illustrativen Schattierung umgesetzt werden kann.

3 Konzept

Die Zeichentechniken Stippling und Hatching basieren darauf, aus der Platzierung schwarzer Zeichenelemente den Eindruck von Graustufen zu erzeugen. Der tatsächliche Grauwert eines Pixels in der Illustration ist daher entweder schwarz oder weiß, dennoch muss für jeden Pixel ein Ziel-Grauwert bestimmt sein, der durch diese Verfahren angenähert wird. Üblicherweise wird in dem Renderer, der das Stippling/Hatching ausführt, der Ziel-Grauwert für jeden Pixel über ein lokales Beleuchtungsmodell errechnet (siehe Abb. 3.1(a)).

Ziel des zu entwerfenden Systems ist es, eine illustrative Helligkeitsverteilung zu berechnen, die Beleuchtungstechniken, Szeneninformationen und gestalterische Darstellungsmittel zu einer komplexen Schattierung kombiniert und damit über einfache Beleuchtung hinausgeht (siehe Abb. 3.1(b)).



(a) Einfache Schattierung mit frontaler Beleuchtung

(b) Komplexe Schattierung mit Beleuchtung, Kantenlicht, Schattenwurf, Silhouetten und Tiefeninformation

Abb. 3.1: Beispiele von Schattierung und daraus resultierendem Stippling/Hatching

Im Folgenden wird das Konzept für ein System entworfen, mit dem eine solche Helligkeitsverteilung erzeugt und zum Rendern mittels üblicher Stippling-/Hatching-Verfahren verwendet werden kann. Dazu wird zunächst analysiert, welche Anforderungen an das System sich aus dieser Zielsetzung ergeben. Im Anschluss werden die dafür verwendbaren Parameter betrachtet und Möglichkeiten diskutiert, wie diese vom Benutzer sinnvoll eingesetzt werden können. Abschließend wird eine Pipeline vorgeschlagen, mit der das System technisch umzusetzen ist.

3.1 Anforderungsanalyse

Medizinische Illustrationen sind je nach Anwendungsziel auf verschiedene Schwerpunkte ausgerichtet und benötigen somit entsprechend angepasste Visualisierungen. Die jeweils geeignete Schattierung zur Darstellung soll dabei als Kombination der im vorigen Kapitel untersuchten Parameter entstehen. Da es theoretisch eine unbegrenzte Anzahl an möglichen Kombinationen gibt, müssen sie auf sinnvolle Weise eingegrenzt werden. Hier ist es zunächst hilfreich, mögliche Anwendungsfälle zu betrachten, die zur Kategorisierung der Parameter dienen können. Es wird nicht auf spezielle Anwendungsszenarien, sondern auf die allgemeine Zielsetzung von interaktiver Illustration in der Medizin eingegangen, welche beispielhaft durch die Gebiete Lehre und Operationsplanung repräsentiert werden.

Wie die lange Tradition medizinischer Atlanten zeigt, sind Illustrationen als Lehrmittel geeignet. Die Zeichnungen haben dabei hauptsächlich das Ziel, die menschliche Anatomie verständlich darzustellen. Der Schwerpunkt liegt also auf der Illustration von Form, Oberfläche und räumlicher Lage der abgebildeten Objekte. Hier ist die Computergrafik hilfreich, um die zweidimensionale statische Zeichnung durch die dritte Dimension zu erweitern und interaktive Erkundung zu ermöglichen. Da dies eine Erweiterung der traditionellen Illustrierung darstellt, sind die in Abschnitt 2.1 analysierten Techniken dafür gut geeignet. Dieser Schwerpunkt wird in der vorliegenden Arbeit als Objektdarstellung bezeichnet.

Die virtuelle Operationsplanung dagegen hat kein historisches Vorbild und konnte erst mit der entsprechenden Technik entstehen: bildgebende Verfahren zur Erfassung patientenspezifischer Daten sowie Computergrafik zur Visualisierung und Simulation von Eingriffen. Hier kommt es mehr auf die Visualisierung operationsrelevanter Informationen an, wofür traditionelle Illustrationstechniken nicht ausreichend sind. Der Schwerpunkt liegt auf Visualisierungstechniken wie in Abschnitt 2.3, die in neuerer Zeit dafür entwickelt wurden. Im weiteren Verlauf wird dieses Anwendungsziel unter dem Begriff Informationsdarstellung zusammengefasst.

Konkrete Anwendungen erfordern diese Schwerpunkte zu unterschiedlichen Anteilen. Es ist anzunehmen, dass die Objektdarstellung immer eine gewisse Rolle spielt, zusätzliche Informationen wie Abstände zwischen Objekten oder Schweregrad einer Erkrankung dagegen nur bei Bedarf visualisiert werden. Andererseits wird bei der Informationsdarstellung die Darstellung der Objekte eher auf das Nötigste beschränkt sein, da der Einsatz von zusätzlichen Gestaltungsmitteln die Vermittlung präziser Szeneninformationen stören könnte. Damit der Schwerpunkt der Illustration entsprechend angepasst werden kann, muss die Kombination der Parameter variabel sein. Variablen sind dabei sowohl die Einstellungsmöglichkeiten der einzelnen Parameter als auch das Ausmaß, in dem sie in die Kombination einfließen. Da die Parameter teils recht unterschiedliche Auswirkungen auf die Visualisierung

haben, ist eine differenzierte Betrachtung ihrer Beteiligung am Endergebnis notwendig.

Die resultierende Schattierung sollte vom Benutzer beeinflussbar sein, um ihm im Rahmen der Interaktivität mehr Freiraum zu lassen. Hier ist ein Mittelweg zwischen darstellerischer Freiheit und Einfachheit der Benutzung zu finden. Der Benutzer sollte dazu in der Lage sein, die Variablen effektiv und zielgerichtet einstellen zu können. Beispielsweise könnte er dabei auf Vorgaben zurückgreifen oder das Visualisierungsziel auf abstrakter Ebene festlegen, wobei die konkrete Belegung der Variablen vom System übernommen wird.

Alle im weiteren Verlauf verwendeten Modelle wurden mit dem Marching-Cubes-Algorithmus [LORENSEN und CLINE, 1987] aus segmentierten CT- und MRT-Aufnahmen gewonnen. Durch den Partialvolumeneffekt und geringe Auflösung sind die Modelle teils mit Ungenauigkeiten und Stufenartefakten behaftet, die durch Glättung der Geometrie nicht vollständig entfernt werden können. Soweit möglich sollte dies bei der Visualisierung beachtet und ausgeglichen werden.

3.2 Auswahl der Parameter

In Kapitel 2 wurden einige Parameter vorgestellt, die für eine gestalterische Schattierung eingesetzt werden können, sowie teils mehrere Methoden zu ihrer Umsetzung. Einige der Parameter hängen von Variablen ab, die ihre genaue Gestaltung bestimmen. Um einen Überblick zu erhalten, folgt zunächst eine Auflistung der im Weiteren verwendeten Parameter. Dazu werden jeweils eine Methode der Umsetzung ausgewählt und die zugehörigen Variablen genannt. Im Interesse einer strukturierten Herangehensweise sind die Parameter in drei Kategorien eingeteilt, die sich aus den Anwendungsschwerpunkten ergeben.

Eine Sonderstellung nimmt die Kontrastverbesserung ein, da es sich um eine Methode der Bildverarbeitung handelt. Sie stellt selbst keine Helligkeitsverteilung dar und hängt nicht von der zugrunde liegenden Geometrie ab, weshalb sie außerhalb der folgenden Kategorien steht. Je nach technischer Umsetzung können bei Stippling und Hatching feine Unterschiede in der Schattierung verloren gehen, deshalb muss die Differenz zwischen Graustufen verstärkt werden. Histogrammäqualisation erreicht dies besser als einfache Histogrammspreizung und wird daher als Methode zur Kontrastverbesserung gewählt. Zudem ist sie stabiler gegenüber Änderungen und erlaubt eine frame-kohärente Darstellung, was mit Spreizung problematisch wäre.

3.2.1 Oberflächengestaltung

Diese Parameter betreffen die reine Objektdarstellung, d. h. Form und Details der Geometrie. Sie bilden eine Kategorie, da die Visualisierung der Objekte für eine Illustration grundlegend ist und in jedem Anwendungsfall eine Rolle spielt. Im Interesse der flexiblen Parametrisierung werden keine kombinierten Lichtmodelle wie von LEE und HAO [2006] herangezogen, bei denen durch mehrere lokale Lichtquellen die gesamte Beleuchtung optimiert wird. Stattdessen sind die einzelnen Beleuchtungstechniken separat umgesetzt, um ihren Einsatz gezielter steuern zu können. Die folgenden Parameter werden betrachtet (siehe Abb. 3.2):

Hauptlicht: Unbeschränkte diffuse Beleuchtung, womit auch Aufhellungslicht simuliert wird. Die Richtung des Hauptlichtes ist frei wählbar. Zusätzlich wird die Option eingeführt, geglättete Normalen zur Berechnung zu verwenden, um eine weichere Schattierung zu erhalten.

Plateaulicht: Diffuse Beleuchtung aus Betrachtungsrichtung. Hier gibt es keine Einstellungsmöglichkeiten.

Streifendes Licht: Lokale Beleuchtung nach RUSINKIEWICZ et al. [2006]. Die Lichtrichtung sollte gleich der des Hauptlichtes sein. Es bleiben die Variablen Überzeichnungsfaktor und Detailfrequenz.

Oberflächenform: Hervorhebung von Bergen und Tälern nach KINDLMANN et al. [2003]. Dieser Parameter benötigt keine weiteren Einstellungen.

Krümmungsstärke: Visualisierung der Krümmungsstärke nach HAMEL [2000]. Auch hier sind keine variablen Einstellungen vorhanden.

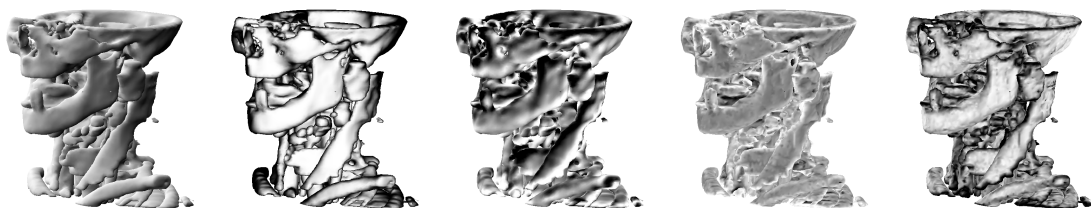


Abb. 3.2: Parameter der Oberflächengestaltung (v.l.n.r.): Hauptlicht, Plateaulicht, Streifendes Licht, Oberflächenform, Krümmungsstärke

Wie schon erwähnt, ist realistischer Schattenwurf in Illustrationen eher selten. Daher wird er nicht weiter betrachtet, würde aber bei Bedarf ebenfalls an dieser Stelle einsortiert werden.

3.2.2 Szeneninformationen

Parameter, die zusätzliche Informationen über Objekte und Szene kodieren, sind hier zusammengefasst. Sie sind im Bereich der Informationsdarstellung von Bedeutung, werden also bei Bedarf zusätzlich zur Objektdarstellung eingesetzt. Es wäre auch möglich, sie ausschließlich zur Illustration zu verwenden, wodurch aber die Objekte an sich nicht mehr erkennbar wären. Hier ist weniger die technische Umsetzung als die Kodierung in Form von Schattierung interessant, worauf in Abschnitt 3.4.1 näher eingegangen wird. Es werden die folgenden Parameter verwendet (siehe Abb. 3.3):

Wichtigkeit: Die Wichtigkeit von Objekten ist anwendungsspezifisch. Sie wird üblicherweise beim Laden der Geometrie festgelegt und kann auch während der Benutzung interaktiv angepasst werden. Da es sich um einen einzelnen Wert handelt, ist es möglich, diesen direkt als Grauwert zu kodieren.

Abstand zum Betrachter: Verwendung des z -Buffers. Es sind keine weiteren Einstellungen nötig.

Abstand zu einem Objekt: Abstand kann wie Tiefe als Grauwert kodiert werden, wobei ein variabler Maximalwert zur Normalisierung nötig ist. Das Objekt, vom dem aus der Abstand gemessen wird, wählt der Benutzer interaktiv aus.

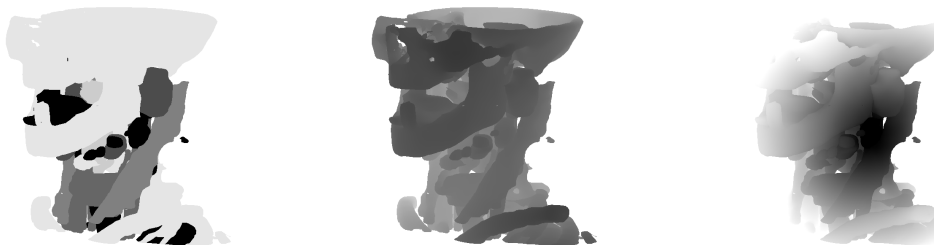


Abb. 3.3: Parameter der Szeneninformationen (v.l.n.r.): Wichtigkeit, Abstand zum Betrachter, Abstand zu einem Objekt

3.2.3 Bildoptimierung

In diese Kategorie fallen alle weiteren Parameter, welche die Illustration übersichtlicher gestalten und einen ästhetisch wertvollen Eindruck vermitteln. Der Unterschied zu den Parametern der Oberflächengestaltung besteht darin, dass sie für eine verständliche Objektdarstellung nicht notwendig sind, aber diese optisch aufwerten können. Im Gegensatz dazu sind sie für die Informationsdarstellung eher

ungeeignet. Trotz Überschneidungen mit den Parametern der Oberflächengestaltung werden sie hier zusammengefasst, da sich ihr Beitrag zur Gesamtschattierung von diesen unterscheidet, wie in Abschnitt 3.4.1 ausgeführt wird. In diese Kategorie fallen (siehe Abb. 3.4):

Kantenlicht: Unschärfemaskierung des z -Buffers nach LUFT et al. [2006], wobei nur positive Werte betrachtet werden. Die Breite des Unschärfefilters ist variabel.

Überdeckungsschatten: Wie Kantenlicht, wobei hier nur die negativen Werte beibehalten werden.

Atmosphärische Perspektive: Manipulation des z -Buffers nach FOLEY et al. [1995] sowie EBERT und RHEINGANS [2000]. Variablen sind minimale und maximale Tiefe, Anstiegsexponent sowie Hintergrundintensität.

Silhouetten/Merkmalenlinien: Kantenfilterung von Objekt- und z -Buffer nach SAITO und TAKAHASHI [1990]. Die Breite der Linien kann durch Art und Dimension des Filters bestimmt werden.

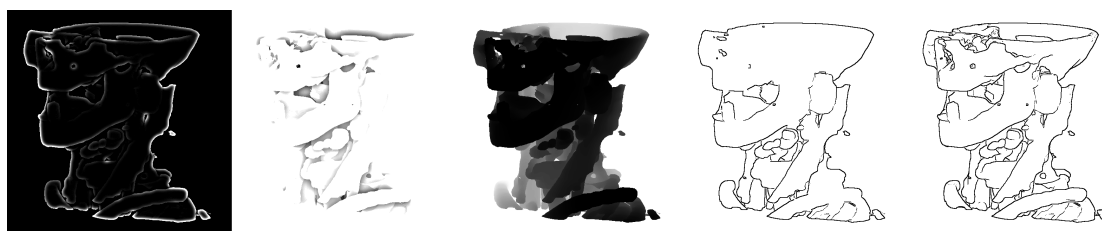


Abb. 3.4: Parameter der Bildoptimierung (v.l.n.r.): Kantenlicht (Hintergrund abgedunkelt), Überdeckungsschatten (um 1 erhöht, damit negative Werte darstellbar sind), atmosphärische Perspektive, Silhouetten, Merkmalslinien (beide invertiert)

3.3 Verschiedene Ansätze zur Parametrisierung

Wie in der Anforderungsanalyse ermittelt wurde, ist es für eine hinreichend variable Gestaltung der Schattierung notwendig, das Ergebnis beeinflussen zu können. Neben den zuvor genannten Variablen kann dafür auch die Auswirkung jedes Parameters auf die Schattierung reguliert werden, was in Form von Gewichtungsfaktoren geschieht. Um aus allen Parametern eine gesamte Schattierung zu erzeugen, müssen sämtliche Variablen, also die Einstellungsmöglichkeiten der einzelnen Parameter sowie ihre Gewichtungsfaktoren, aufeinander abgestimmt werden. Daher ist als nächstes der Frage nachzugehen, wie eine sinnvolle Parametrisierung möglich ist

und welchen Einfluss der Anwender darauf hat. Er soll einerseits die Freiheit haben, die Darstellung unterschiedlich auszurichten, andererseits nicht von der Menge an Einstellungen überfordert werden.

Weiter ist zu unterscheiden, wie detailliert die Parametrisierung einer Szene vorgenommen wird. Es bieten sich zwei Möglichkeiten an: eine einheitliche Parametrisierung der gesamten Szene (Abb. 3.5(a)) und eine nach Strukturtyp differenzierte (Abb. 3.5(b)). Während eine szenenweite Parametrisierung zu einer einheitlichen Darstellung führt, würde es die Einstellung pro Strukturtyp beispielsweise erlauben, Organe detaillierter und einbettendes Gewebe glatter und unauffälliger darzustellen. Es sind auch andere Gruppierungen möglich, beispielsweise in Fokus- und Kontextobjekte. Das Prinzip der Parametrisierung von Objektgruppen bleibt dabei jedoch gleich, daher werden sie nicht gesondert behandelt.

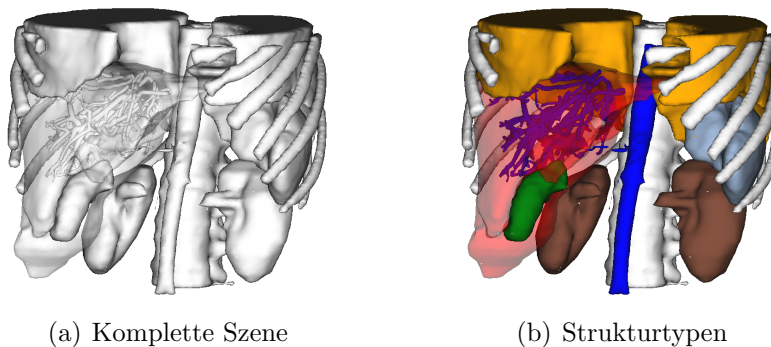


Abb. 3.5: Ebenen der Parametrisierung: Variablen können spezifiziert werden für (a) die gesamte Szene oder (b) Gruppen von Strukturtypen (Lungenflügel orange, Nieren braun, Gefäße blau usw.)

Zusammen mit den gestalterischen Möglichkeiten steigt auch die Komplexität der Parametrisierung. Bei einer szenenweiten Einstellung wird jede Variable nur einmal benötigt. Bei der separaten Schattierung einzelner Strukturtypen erhöht sich die Anzahl der Variablen beträchtlich, da jede Variable für jeden Typ festgelegt werden muss. Für die folgende Diskussion wird zunächst von einer manuellen, statischen Parametrisierung ausgegangen, d. h. sie wird einmal vorgenommen und bleibt über die Dauer der Anwendung konstant.

Die Einstellung sämtlicher Variablen von Hand ist aufgrund der Menge an Möglichkeiten nicht sehr effizient. Der Benutzer sollte in der Lage sein, die Darstellung zielgerichtet zu verändern. Es wird nun näher untersucht, wie die Variablen auf wenige und einfache Einstellungen reduziert werden können, die jeweils eine direkt nachvollziehbare Wirkung haben.

3.3.1 Standardwerte

Zunächst ist festzustellen, dass für einige Variablen sinnvolle Standardwerte existieren. Diese beruhen auf Konventionen oder sind empirisch ermittelt. Einige Variablen können daher aus der Parametrisierung eliminiert werden, indem sie Standardwerte erhalten, die der Benutzer nicht verändern kann. Dabei ist auf eine angemessene Anzahl zu achten: Gibt man zu wenig Standardwerte vor, ist die Parametrisierung für den Benutzer nicht effektiv vereinfacht. Zu viele Standardwerte dagegen schränken die Flexibilität der Darstellung stark ein.

Durch informelle Evaluierung und Diskussion wurden folgende Variablen ermittelt, die sich als Vorgaben eignen. Soweit nicht anders vermerkt, werden die genannten Werte in der weiteren Arbeit als Standardwerte angenommen:

Richtung des Hauptlichtes: Wie schon erwähnt, ist es in Illustrationen üblich, ein von schräg oben einfallendes Hauptlicht anzunehmen. Der Vektor der Lichtrichtung wird daher von links oben nach rechts unten ausgerichtet. Er steht senkrecht zur Blickrichtung, wodurch der Term der unbeschränkten diffusen Beleuchtung optimal ausgenutzt wird.

Gewichtung des streifenden Lichtes: Das Ausmaß, mit dem streifendes Licht Details hervorhebt, ist sowohl durch seine Gewichtung als auch durch den Überzeichnungsfaktor bestimmt. Um den Eindruck einer Hauptlichtquelle beizubehalten, sollte die Gewichtung etwa im Verhältnis 1:2 zu Hauptlicht und Plateaulicht stehen, ein höheres Verhältnis ergibt unbefriedigende Resultate. Wird dieses Verhältnis konstant beibehalten, kann die Detailstärke alleine durch den Überzeichnungsfaktor reguliert werden, was die Parametrisierung weiter vereinfacht.

Hintergrundintensität: Entsprechend dem realen Vorbild wird weiß als Hintergrundintensität der atmosphärischen Perspektive gewählt. Andere Grauwerte wirken in Schwarz-Weiß-Darstellungen ungünstig, da der Hintergrund durch sie mehr optisches Gewicht erhält und damit eher hervorgehoben statt abgeschwächt wird (siehe Abb. 3.6).

Maximale Tiefe: Eine maximale Tiefe d_f von 0.8 ist unabhängig von minimaler Tiefe und Anstiegsexponent angemessen, um den Hintergrund mittels atmosphärischer Perspektive effektiv abzuschwächen.

Gewichtung der Merkmalslinien: Merkmalslinien sollten nicht zu kräftig dargestellt werden, damit sie einen illustrativen Charakter erhalten und sich besser in die Darstellung einfügen (siehe Abb. 3.7). Ein günstiger Wert ist 80% der vollen Stärke.

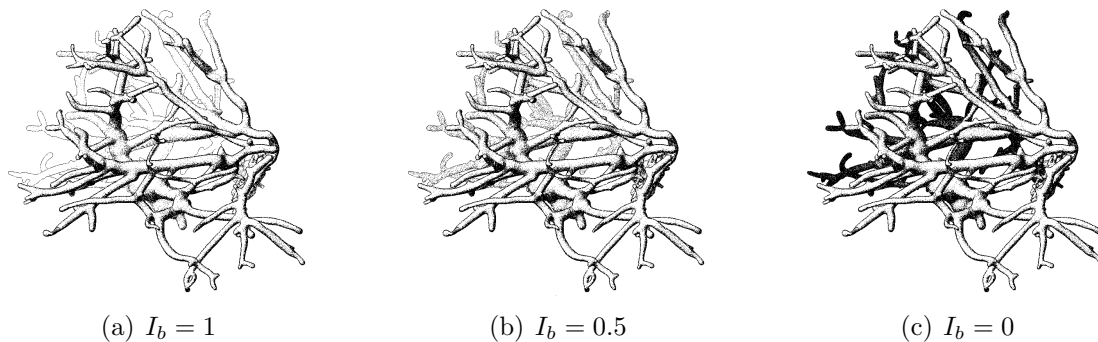


Abb. 3.6: Darstellung von Blutgefäßen der Leber mit atmosphärischer Perspektive und verschiedenen Hintergrundintensitäten I_b : Die beste Abschwächung des Hintergrundes findet in (a) statt.

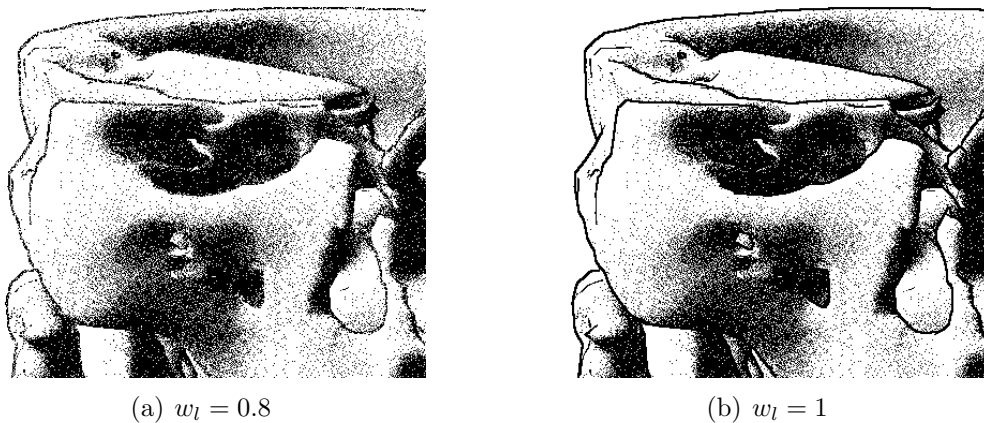


Abb. 3.7: Gewichtung w_l der Merkmalslinien im Detail: Leicht abgeschwächte Linien fügen sich besser in die Illustration ein.

3.3.2 Kombinierte Variablen

Eine weitere Form der Reduktion besteht darin, Variablen nach übergreifenden Gesichtspunkten zusammenzufassen. Durch geschickte Kombination kann der Benutzer anschauliche Eigenschaften der Darstellung mit einer Einstellung verändern, ohne sich um die Belegung der einzelnen Variablen zu kümmern. Dies erlaubt eine weitaus intuitivere Parametrisierung. Beispiele für kombinierte Variablen sind:

Randbetonung: Außer der Breite der Merkmalslinien bestimmt auch das Verhältnis zwischen Hauptlicht und Plateaulicht, wie stark sich die Ränder eines Objektes abzeichnen. Daher können beide Gewichtungsfaktoren zusammen mit der Linienbreite zur Randbetonung zusammengefasst werden, deren Wirkung in Abb. 3.8 dargestellt ist.

Detailamplitude: Details werden durch den Überzeichnungsfaktor des streifenförmigen Lichtes und die Gewichtung der Oberflächenform hervorgehoben, durch Glättung des Hauptlichtes dagegen abgeschwächt. Diese Variablen werden übergreifend zur Detailamplitude kombiniert (siehe Abb. 3.9), die im Gegensatz zur Detailfrequenz nicht die Größe, sondern die Stärke von Details reguliert. Dabei entspricht eine mittlere Einstellung dem originalen Modell, die Extreme der zusätzlichen Glättung bzw. Überzeichnung.

Räumliche Wirkung: Kantenlicht, Überdeckungsschatten und atmosphärische Perspektive tragen alle zur räumlichen Wirkung einer Szene bei (siehe Abb. 3.10). Ihre Gewichtungsfaktoren können entsprechend zu einer übergreifenden Einstellung kombiniert werden.

Ausdehnung des Vordergrundes: Sowohl minimale Tiefe als auch Anstiegsexponent bestimmen den Bereich, der bei Verwendung der atmosphärischen Perspektive als Vordergrund hervorgehoben wird. Sie können demnach zusammengelegt werden, um die Ausdehnung des Vordergrundes zu regulieren (siehe Abb. 3.11).

Folgende Überlegungen führen auch bei den Variablen der Szeneninformationen zu einer Vereinfachung: Die gleichzeitige Visualisierung mehrerer Informationen liefert in der Regel kein gutes Ergebnis, da sich diese in ihrer Wirkung gegenseitig überlagern. Ebenso ist es angebracht, sie bei Bedarf mit voller Gewichtung in die Schattierung einfließen zu lassen, damit die Information genau vermittelt wird. Daher kann ihre Parametrisierung darauf beschränkt werden, eine der vorhandenen Informationen zur Darstellung auszuwählen, die maximale Gewichtung erhält.

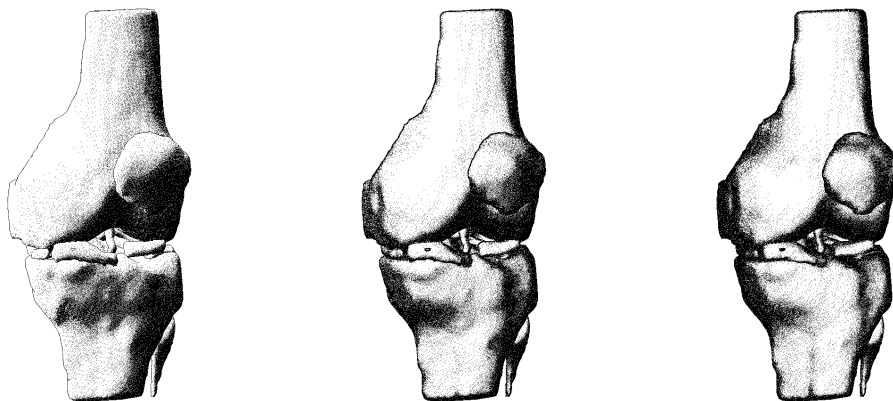


Abb. 3.8: Veränderung der Randbetonung: Der Anteil des Hauptlichtes nimmt von links nach rechts ab, der des Plateaulichtes zu. Die Breite der Merkmalslinien steigt ebenfalls an.

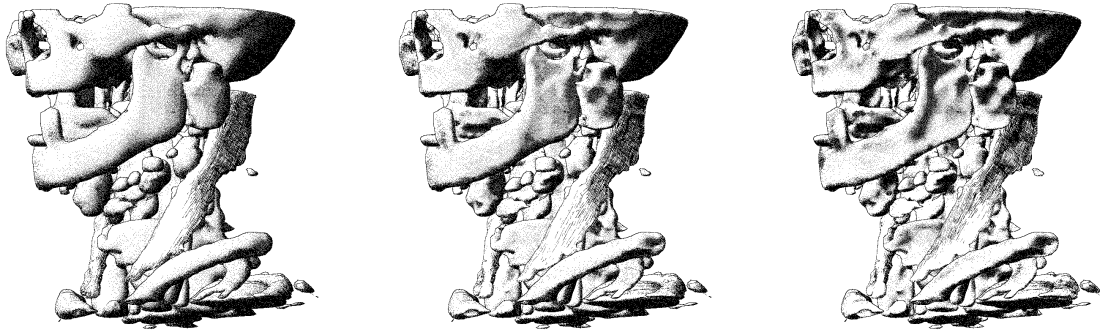


Abb. 3.9: Veränderung der Detailamplitude: Die Glättung des Hauptlichtes nimmt von links nach rechts ab, Überzeichnungsfaktor und Gewichtung der Oberflächenform nehmen zu.

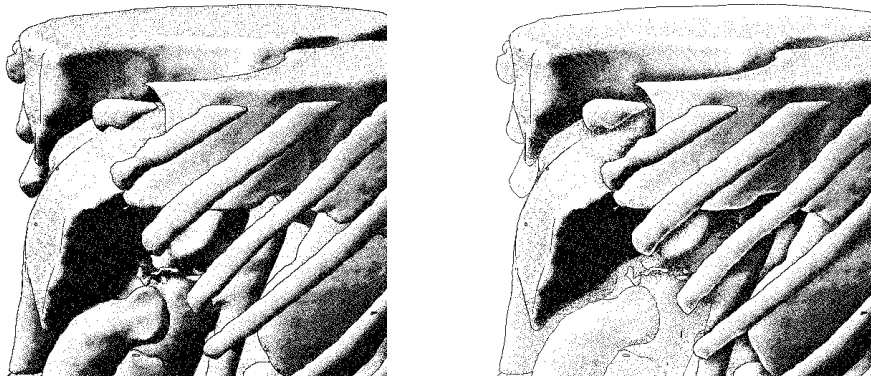


Abb. 3.10: Räumliche Wirkung: Keine Verstärkung (links). Verstärkung durch Kantenlicht, Überdeckungsschatten und atmosphärische Perspektive (rechts).



Abb. 3.11: Ausdehnung des Vordergrundes: Durch Verringerung der minimalen Tiefe und des Anstiegsexponenten wird die Aufmerksamkeit stärker auf den Vordergrund konzentriert.

3.3.3 Verwendung von Templates

Während durch Standardwerte und Kombinationen die Anzahl an Variablen reduziert wird, muss der Benutzer für die übrigen immer noch passende Werte finden. Diese Aufgabe kann durch *Templates* vereinfacht werden. Sie enthalten jeweils eine komplette Parametrisierung, die einem bestimmten Darstellungsziel entspricht. Der Benutzer wählt nur das gewünschte Template aus und weist es der Szene oder den einzelnen Strukturtypen zu. Es bietet sich an, Templates durch Abbildungen zu repräsentieren, welche den jeweiligen Stil anhand einer einfachen Szene darstellen. Dieses Prinzip wird ähnlich von BRUCKNER und GRÖLLER [2007] für ihre *Style Transfer Functions* eingesetzt und findet auch in Materialbibliotheken gängiger 3D-Modellierungssoftware Verwendung. Denkbar wäre auch, die aktuelle Szene selbst unter Anwendung verschiedener Templates darzustellen, wodurch eine noch intuitivere Auswahl möglich ist. Templates können grundlegende und kombinierte Variablen enthalten; da der Benutzer nicht direkt auf sie zugreift, ist ihre Reduzierung jedoch nicht notwendig. Auch Standardwerte müssen so nicht absolut sein, sondern können für verschiedene Visualisierungsstile verändert werden.

Für beliebige Szenen lassen sich offensichtlich nur sehr allgemeine Empfehlungen angeben. Als generelle Vorgabe kann gelten: Das Hauptlicht sollte überwiegen, das Plateaulicht maximal die halbe Gewichtung erhalten und schwache Merkmalslinien verwendet werden, um eine leichte Randbetonung zu erzeugen. Der Übersichtlichkeit halber ist verstärkte räumliche Wirkung sinnvoll, wobei der Vordergrund nicht zu stark beschränkt werden sollte. Abb. 3.13 zeigt drei Szenen mit gleicher Parametrisierung, die der obigen Beschreibung entspricht.

Auflösung und Qualität der Modelle verlangen besondere Beachtung. Bei starken Artefakten, die wie erwähnt bei der Aufnahme oder Segmentierung entstehen können, sollte eine geringe Detailamplitude oder starke Glättung gewählt werden (siehe Abb. 3.12). Sind die Modelle dagegen hinreichend genau und detailliert, können diese Details durch eine hohe Amplitude hervorgehoben werden.

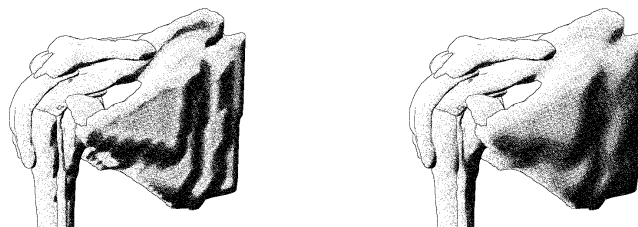


Abb. 3.12: Das Modell des Schultermuskels weist starke Segmentierungsartefakte auf (links). Durch Glättung werden diese deutlich abgemildert, wobei auch andere Details verloren gehen können (rechts).

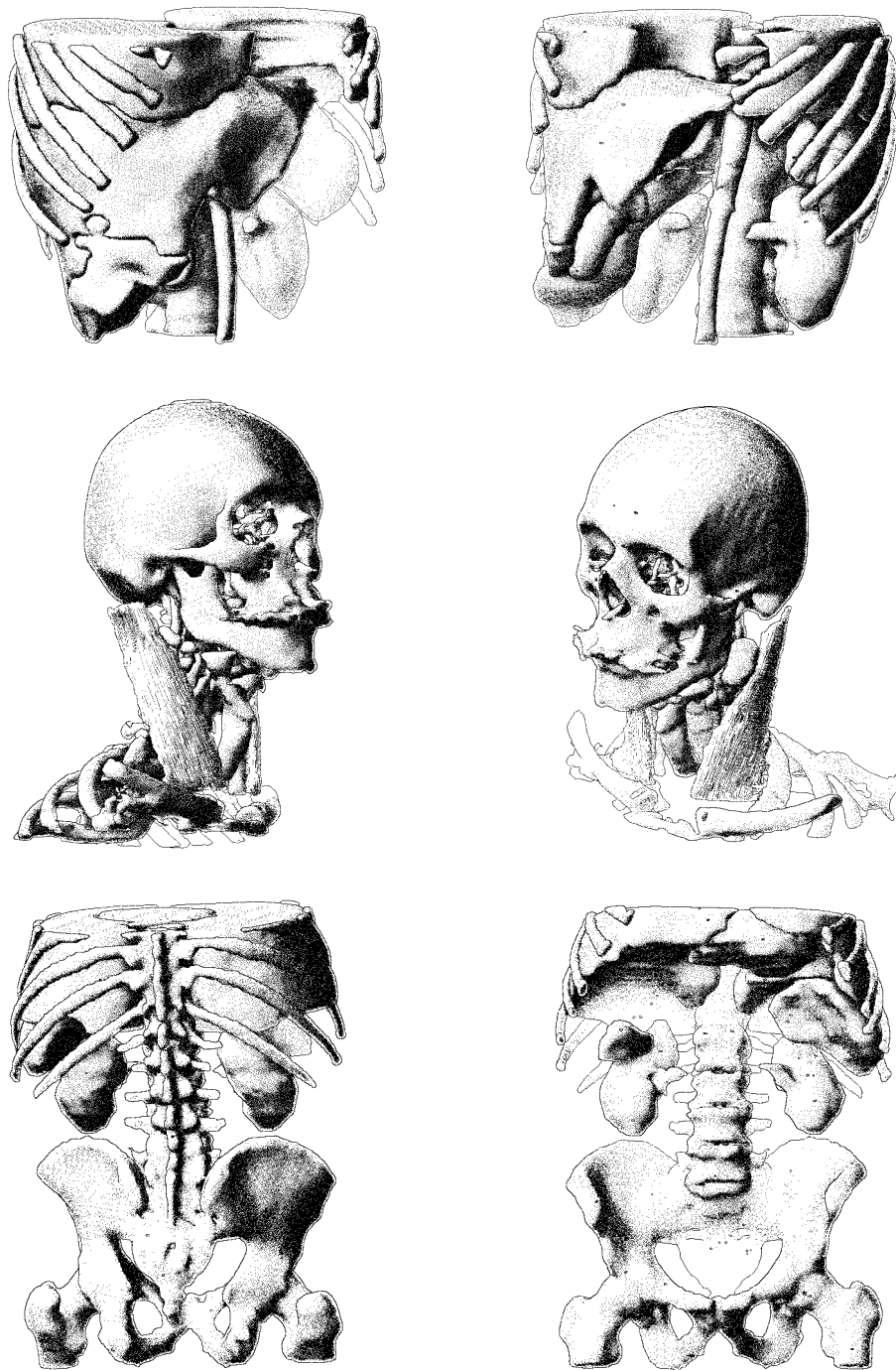


Abb. 3.13: Alle Ansichten wurden mit der gleichen Parametrisierung erstellt. Durch starkes Hauptlicht und schwaches Plateaulicht entsteht eine leichte Randbetonung. Die Modelle sind zusätzlich geglättet. Räumliche Wirkung wird durch breiten Überdeckungsschatten, schmales Kantenlicht und atmosphärische Perspektive verstärkt, der Vordergrundbereich ist weit ausgedehnt.

Eine bessere Schattierung ergibt sich durch die Parametrisierung einzelner Strukturtypen. Entsprechende Templates könnten der Geometrie automatisch zugewiesen werden, um die Benutzung weiter zu vereinfachen. Der Benutzer müsste nur in den Prozess eingreifen, falls die automatische Auswahl nicht optimal ist. Die folgende Aufzählung gibt geeignete Einstellungen für verschiedene Strukturen an:

Blutgefäße: Bei röhrenförmigen Strukturen bietet es sich grundsätzlich an, den Rand zu betonen, da so auch die Rundung hervorgehoben wird. Alle Gefäße sollten daher eine höhere Randbetonung erhalten, vor allem durch einen hohen Anteil an Plateaulicht. Kantenlicht kann zur weiteren Betonung der Form verwendet werden (siehe Abb. 3.14). Beide Techniken sind auch in Handzeichnungen üblich. Weiter ist Glättung vorteilhaft, da Oberflächendetails bei Gefäßen keine große Rolle spielen und bei kleinen Strukturen eher störend wirken. Bei komplexen Gefäßbäumen sollte der Vordergrund stark hervorgehoben werden, da sie sonst sehr unübersichtlich erscheinen (siehe Abb. 3.14(b)).

Knochen: Auch Knochen sollten generell glatt dargestellt werden. Es ist ihrer Form nach zu unterscheiden: Bei röhrenförmigen Knochen ist wie bei Gefäßen die Betonung des Randes angebracht (siehe Abb. 3.15(a)). Größere Knochenstrukturen wie Schädel und Becken dagegen profitieren von stärkerem Hauptlicht (siehe Abb. 3.15(b)). Auch hier ist es bei ausgedehnten Strukturen wie dem Brustkorb hilfreich, die räumliche Wirkung zu verstärken und die Aufmerksamkeit auf den Vordergrund zu konzentrieren.

Organe/Muskeln: Wie bei Knochenstrukturen ist bei Organen und Muskeln ein starkes Hauptlicht angebracht. Da einzelne Organe meist eine kompakte Form besitzen, ist keine Betonung der Raumwirkung notwendig. Die Detailstärke hängt von der konkreten Struktur ab: Handelt es sich um die Außenseiten von Organen wie in Abb. 3.16, ist Glättung sinnvoll. Strukturen mit ausgeprägter Oberflächenform (z. B. Magenwand, Gehirn, geäderte Herzoberfläche) können dagegen mit hoher Detailamplitude dargestellt werden. Hier bietet sich auch die Visualisierung von Oberflächenform oder Krümmungsstärke an.

Diese Templates sind für die allgemeine Illustration geeignet, für andere Anwendungen müssen sie eventuell angepasst werden. So kann bei allen Strukturen auch die Betonung von Details wichtig sein, um pathologische Veränderungen erkennbar zu machen, beispielsweise Aneurysmen an Gefäßen oder Frakturen an Knochen. Dies setzt hinreichend genaue Modelle voraus und hängt auch vom Größenverhältnis der interessanten Bereiche zu ungewollten Artefakten ab. Die generellen Templates sind daher weiter differenzierbar, um verschiedene Anwendungsfälle, Modellqualitäten, Detailstufen oder Strukturgrößen abzudecken. Eine genauere Untersuchung verlangt jedoch ein breites Spektrum an Modellen sowie eine ausführliche Evaluierung mit potentiellen Anwendern, was den Rahmen dieser Arbeit übersteigt.

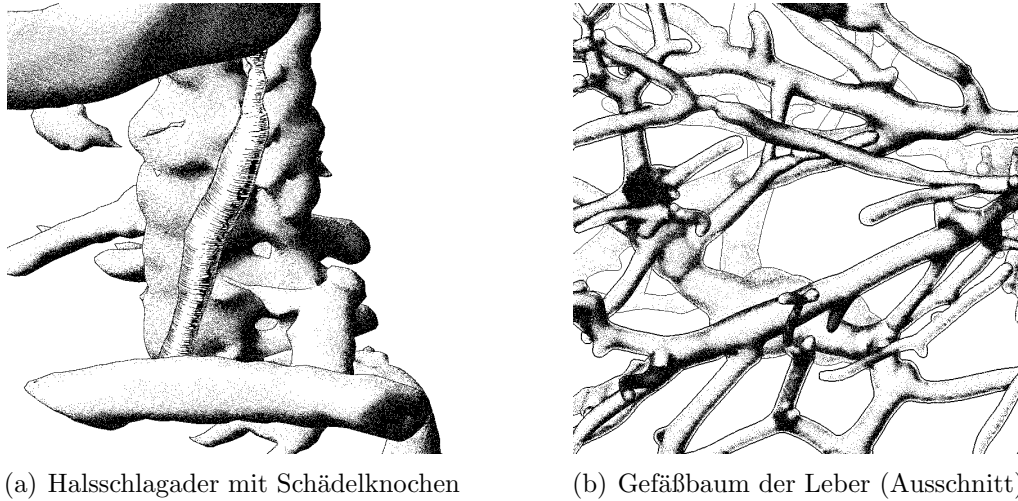


Abb. 3.14: Parametrisierung von Blutgefäßen: Durch Plateau- und Kantenlicht wird die runde Form betont. Rechts ist der Hintergrund zusätzlich stark abgeschwächt, um die Illustration übersichtlicher zu gestalten.

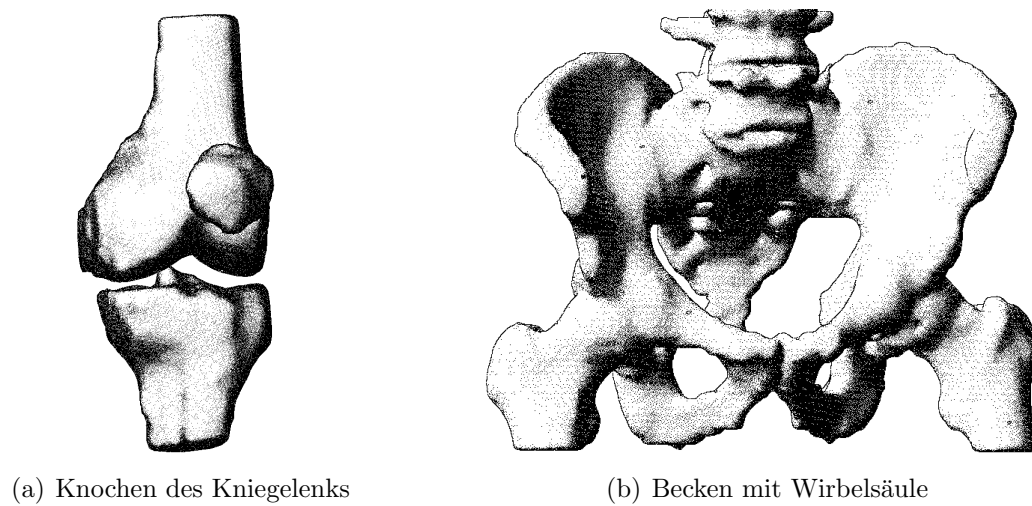
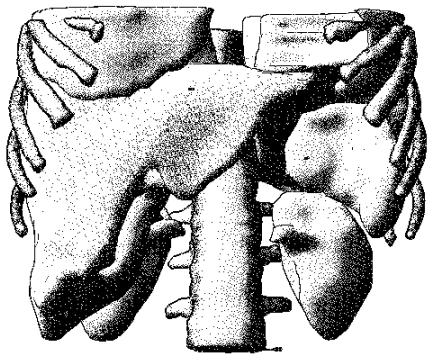
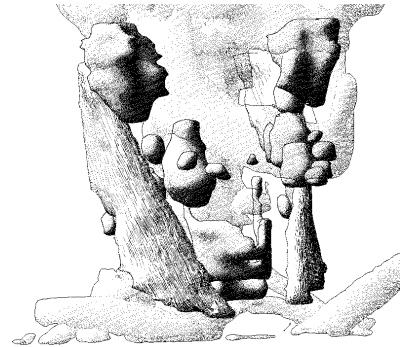


Abb. 3.15: Parametrisierung von Knochen: Bei röhrenförmigen Knochen (a) ist eine starke, bei ausgedehnten Strukturen (b) eine schwache Randbetonung geeignet.



(a) Lunge, Leber, Milz und Nieren. Die Rippen sind mit stärkerer Randbetonung illustriert.



(b) Halsmuskeln, Lymphknoten und Drüsen. Knochen sind transparent dargestellt.

Abb. 3.16: Parametrisierung von Organen und Muskeln: Detailarme Oberflächen sind geglättet und vorwiegend mit Hauptlicht illustriert. Für einzelne Organe ist räumliche Wirkung nicht nötig.

Des Weiteren eignen sich Templates auch für die Informationsdarstellung. Bei der Visualisierung zusätzlicher Informationen muss die Objektdarstellung einfach gehalten werden, damit Schattierung, die der optischen Aufwertung dient, nicht mit den zu vermittelnden Informationen verwechselt wird. Das bedeutet vor allem, dass keine Verstärkung der räumlichen Wirkung stattfinden sollte. In Abb. 3.17 ist dies für Wichtigkeit veranschaulicht: Bei der dargestellten Szene sind Tumor und Lymphknoten besonders wichtig, die Knochenstruktur ist Kontextobjekt. Die Strukturen werden mit abnehmender Wichtigkeit schwächer dargestellt, um den Blick auf die Fokusobjekte zu lenken. Abb. 3.18 zeigt eine ähnliche Visualisierung des Abstandes von einem der Lymphknoten. Die Grundschattierung der Strukturen ist einfach und einheitlich, damit die visualisierte Information besser hervortritt.

Bei Visualisierung der Entfernung zum Betrachter ist es am besten, die Objektdarstellung noch einfacher zu halten, d. h. reines Haupt- oder Plateaulicht zu verwenden und keine Details zu zeigen (siehe Abb. 3.19). So kann die tatsächliche Entfernung aus der Illustration abgeschätzt werden, im Gegensatz zur groben Trennung in Vorder- und Hintergrund bei Verwendung der atmosphärischen Perspektive. Dies geht jedoch auf Kosten einer ansprechenden Objektdarstellung, da die gestalterischen Möglichkeiten der Schattierung nicht genutzt werden können.

Es wäre auch möglich, Templates aus wenigen Variablen zu bilden, die nur bestimmte Eigenschaften verändern. Sie werden zu einer schon vorhandenen Parametrisierung addiert, indem sie die enthaltenen Variablen überschreiben und alle anderen nicht verändern. So könnte ein Template detailreiche Objekte durch starke Glättung unauffälliger machen oder durch breite Silhouetten und extremes Kantenlicht einzelne Objekten hervorheben (siehe Abb. 3.20).

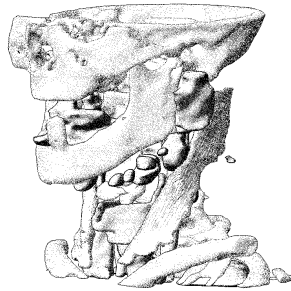


Abb. 3.17: Tumor und Lymphknoten sind mit normaler Schattierung dargestellt, Strukturen mit geringer Wichtigkeit werden leicht (links) bzw. stark (rechts) ausgeblendet.

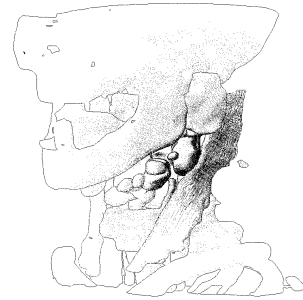
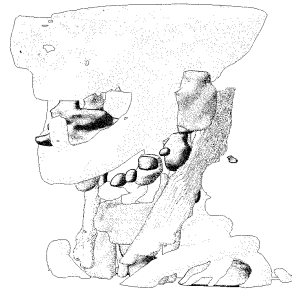


Abb. 3.18: Strukturen werden mit wachsendem Abstand zum zentral liegenden Lymphknoten abgeschwächt.

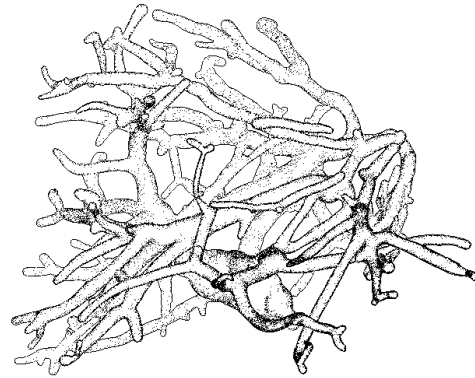
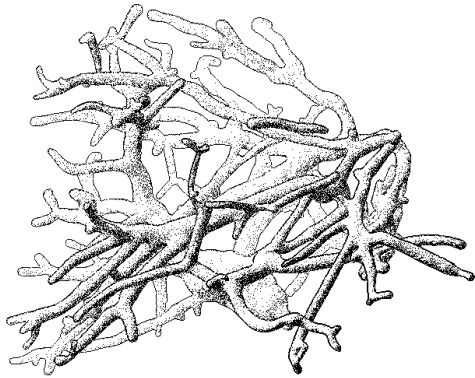


Abb. 3.19: Der Gefäßbaum ist nur mit Hauptlicht (links) bzw. Plateaulicht (rechts) schattiert, damit die Entfernung der Äste zum Betrachter und zueinander besser erkennbar ist.

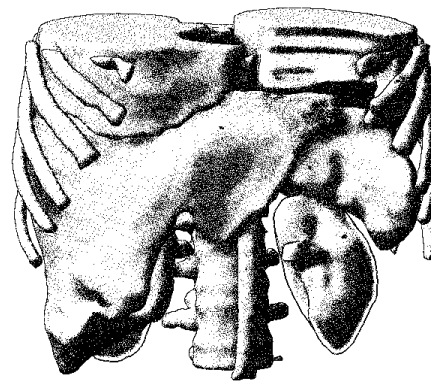
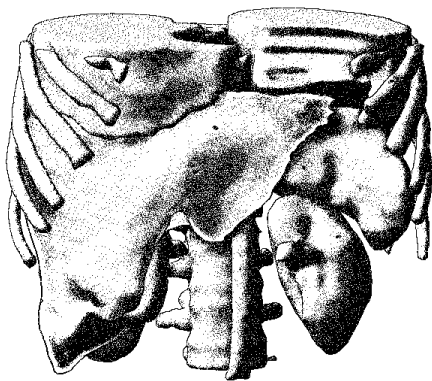


Abb. 3.20: Durch starkes Kantenlicht werden die Leber (links) bzw. die Nieren (rechts) hervorgehoben.

3.3.4 Aufwand der manuellen Parametrisierung

Die vorgestellten Ansätze haben das Ziel, die Parametrisierung für den Benutzer zu vereinfachen. Daher stellt sich die Frage, welche Zeit- und Aufwandsersparnis sie tatsächlich erbringen. Es kann nicht generell angegeben werden, wie lange ein Anwender braucht, um für ein gegebenes Modell eine gute Schattierung zu erreichen: Zum einen ist die Beurteilung der Qualität einer Schattierung nicht nur praktisch, sondern auch ästhetisch bedingt und damit sehr individuell. Zum anderen ist die Vorstellung des Anwenders nicht notwendigerweise exakt umsetzbar. Daher wird stattdessen betrachtet, wie effizient die Parametrisierung durchführbar ist, d. h. wie zielgerichtet der Benutzer die Schattierung nach Wunsch verändern kann.

Da jede Veränderung in Echtzeit am Modell sichtbar wird, sind die Auswirkung einzelner Einstellungen direkt beobachtbar, was die Parametrisierung stark vereinfacht. Der Benutzer kann also prinzipiell durch Ausprobieren die Auswirkungen sämtlicher Variablen erfahren. Ohne jede Vereinfachung stehen dem Benutzer jedoch 24 Variablen zur Verfügung, deren Auswirkungen teils voneinander abhängen, sich überschneiden oder direkt entgegengesetzt sind. Ein Anwender ohne Vorwissen brauchte so recht lange, um die Parametrisierung sinnvoll zu betreiben, da er zunächst die Zusammenhänge verstehen müsste. Hier bietet schon die Reduktion durch Standardwerte einen großen Vorteil, reicht aber noch nicht aus.

Mit dem Modell der kombinierten Variablen kann die Parametrisierung auf sechs Einstellungen reduziert werden, deren Auswirkungen weitestgehend unabhängig sind:

- Randbetonung
- Detailamplitude
- Detailfrequenz
- Räumliche Wirkung
- Ausdehnung des Vordergrundes
- Auswahl der darzustellenden Szeneninformation

Auch ein unerfahrener Anwender sollte durch die anschauliche Veränderung der übergreifenden Einstellungen recht schnell ihre Wirkung begreifen können. Der Vorgang ist durch folgende Strategie weiter zu erleichtern: Da rein diffuse Schattierung zunächst immer ein passables Ergebnis liefert, kann mit einer einfachen geglätteten Beleuchtung durch Hauptlicht begonnen werden. Details, räumliche Wirkung usw. werden nun nach Bedarf in der genannten Reihenfolge hinzugefügt, um die Illustration zu verbessern, wobei ihre jeweilige Wirkung direkt sichtbar wird. So ist es

möglich, die anfängliche Hauptlicht-Beleuchtung in wenigen Minuten zielgerichtet anzupassen.

Die Verwendung von Templates würde idealerweise nur minimalen Aufwand für den Benutzer bedeuten, indem sie schon automatisch den einzelnen Strukturtypen zugewiesen würden. Falls jedoch eine weitere Anpassung erwünscht ist, ist auch diese nach der genannten Strategie sehr einfach möglich. So kann statt eines einfachen Hauptlichtes ein geeignetes Template als Ausgangslage dienen, die der Benutzer nach Bedarf anpasst.

3.3.5 Abgestufte Einstellungen

Um dem Benutzer je nach Erfahrung Einfachheit oder Flexibilität zu bieten, werden Einstellungen in Programmen häufig in mehrere Stufen aufgeteilt. Ein unerfahrener Benutzer erhält nur Zugriff auf die wichtigsten Einstellungen, ein Experte dagegen kann sämtliche Möglichkeiten nutzen, um das Programm für sich zu optimieren. Ein solcher Ansatz, in dem alle oben genannten Strategien verwendet werden, ist auch für die Parametrisierung denkbar. Auf der obersten Stufe stehen dem Benutzer nur feste Templates zur Auswahl. In der nächsten Stufe kann er auf die kombinierten und grundlegenden Variablen zugreifen, aus denen die Templates bestehen, um sie besser an die gewünschte Darstellung anzupassen. Ein noch erfahrenerer Anwender erhält auch Zugriff auf die einzelnen Komponenten der kombinierten Variablen. Zusätzlich können einige Variablen mit Standardwerten belegt werden, die erst in einer weiteren Stufe veränderbar sind.

3.3.6 Automatische Parametrisierung

Die genannten Strategien zur Vereinfachung gehen davon aus, dass für jede Szene eine bestimmte Parametrisierung gefunden werden kann, die sie aus allen Richtungen hinreichend gut schattiert. Offensichtlich ist dies noch keine optimale Lösung, da eine komplexe Szene sehr unterschiedliche geometrische Verhältnisse aufweisen kann. Um die bestmögliche Darstellung zu erhalten, sollten die Variablen mit wechselnder Ansicht leicht angepasst werden, ohne das übergeordnete Darstellungsziel zu verändern. Auf manuellem Wege ist dies nicht praktikabel zu erledigen. Daher wird nun die Möglichkeit der automatischen Parametrisierung betrachtet.

Für eine automatische Einstellung wird eine Funktion benötigt, die die Qualität der Schattierung beschreibt und durch Änderung der Variablen optimierbar ist. Wie in Abschnitt 2.2 erwähnt, stellen SHACKED und LISCHINSKI [2001] eine solche Funktion der Wahrnehmungsqualität für statische Ansichten auf. Während es

prinzipiell denkbar ist, mit den hier vorliegenden Variablen ähnlich zu verfahren, müssen einige wichtige Unterschiede beachtet werden.

Die Berechnung der Wahrnehmungsqualität von SHACKED und LISCHINSKI [2001] beruht auf sehr allgemeinen Faktoren der optischen Wahrnehmung. Um die unterschiedlichen Ausrichtungen zu nutzen, die der hier beschriebene Schattierungsansatz bietet, müssten zusätzlich speziellere Kriterien für eine optimale Schattierung erarbeitet werden. Die übergreifenden Gesichtspunkte der kombinierten Variablen könnten entsprechend abgewandelt werden, indem sie nicht direkt grundlegende Variablen steuern, sondern als Faktoren in die Funktion einfließen. Die konkreten Werte der Variablen ergeben sich nun aus der Optimierung der Funktion und hängen damit zusätzlich von den geometrischen Bedingungen und der aktuellen Ansicht ab. Da sehr viele unterschiedliche Variablen das Ergebnis beeinflussen, ist ein entsprechend komplexer und dennoch echtzeitfähiger Optimierungsprozess zu entwerfen. Bei SHACKED und LISCHINSKI [2001] dienen Intensität und Position von ein bis zwei Lichtquellen als freie Variablen, ihr Ansatz ist also nicht übertragbar. Um die Komplexität der Optimierung zu verringern, können einige Variablen wiederum mit Standardwerten belegt werden.

Aus der Interaktivität folgt als weiterer kritischer Punkt die Anforderung nach einer frame-kohärenten Darstellung. Falls die gewählte Funktion dies nicht leistet, muss der Optimierungsprozess entsprechend angepasst werden. Ein Ansatz dafür wäre es, die Parametrisierung nicht für jeden Frame komplett neu zu berechnen, sondern die des letzten Frames als Ausgangspunkt für den nächsten heranzuziehen. Dadurch sollte es möglich sein, die Veränderungen der Variablen so gering zu halten, dass sich ein fließender Übergang zwischen den einzelnen Frames ergibt. Dies würde auch der Echtzeitfähigkeit zugute kommen.

3.3.7 Zusammenfassung

Es wurden mehrere Ansätze vorgestellt, mit deren Hilfe die Parametrisierung für den Benutzer vereinfacht werden kann. Dies geschieht auf zwei verschiedene Arten: Standardwerte und kombinierte Variablen verringern die Menge an Einstellungen, wodurch der Benutzer die Darstellung effizienter parametrisieren kann. Dies muss er jedoch noch selbst vornehmen. Mit Templates dagegen erhält er komplette Parametrisierungen für bestimmte Strukturtypen und Anwendungsziele, hier muss er also nur die gewünschte Darstellungsart auswählen. Sollen die Templates noch angepasst werden, ist dafür wiederum der erste Ansatz anwendbar. Bei der automatischen Parametrisierung sind die Faktoren der Optimierungsfunktion festzulegen, was ebenfalls auf beide genannten Weisen vereinfacht werden kann.

In Tabelle 3.1 sind die vorgestellten Templates für mehrere Strukturtypen und komplette Szenen noch einmal zusammengefasst. Für die Beschreibung der Einstellungen werden die kombinierten Variablen verwendet, die in Abschnitt 3.3.2 vorgestellt wurden. Obwohl die Detailfrequenz auch eine wichtige Einstellung ist, wird sie hier nicht aufgeführt, da ihr Wert stark von Qualität und Auflösung der Modelle abhängt und so keine allgemeine Empfehlung möglich ist.

Template	Kombinierte Variablen			
	Randbetonung	Detailamplitude	Räumliche Wirkung	Ausdehnung des Vordergrundes
Gesamte Szene	0	*)	+	+
Blutgefäße	+	-	-	*)
Gefäßbäume	+	-	+	-
Einzelne Knochen	+	-	-	*)
Knochenstrukturen	-	-	+	0
Organe/Muskeln	-	*)	-	*)
Szeneninformationen	0	-	-	*)

Tabelle 3.1: Geeignete Einstellungen der kombinierten Variablen für verschiedene Templates. Kleine, mittlere und große Werte sind mit -, 0 und + angedeutet. *) Wert hängt vom Modell ab. *) Wert ist nicht relevant, da keine Verstärkung der räumlichen Wirkung stattfindet.

3.4 Entwurf des Systems

Neben der Umsetzung einer effizienten Parametrisierung ist auch der Frage nachzugehen, wie die Parameter zu einer gemeinsamen Schattierung kombiniert werden können. Die Algorithmen zur Berechnung der Parameter weisen teils sehr unterschiedliche rechnerische Eigenschaften auf. Daraus ergeben sich einige technische Anforderungen an das gesuchte System, die nun erläutert werden:

Lokale/globale Berechnung: Beleuchtung kann lokal an jedem Punkt der Oberfläche berechnet werden, Histogrammoperationen dagegen sind nicht lokal durchzuführen. Daraus folgt, dass die Helligkeit eines Oberflächenpunktes abhängig von seiner Umgebung ist und nicht wie bei einer üblichen Renderpipeline erst lokal im Renderer ermittelt werden kann. Daraus ergibt sich die Grundidee des Systems, die Helligkeitsverteilung in einem separaten Prozess zu berechnen und dem Renderer zur Verfügung zu stellen.

Objekt-/Bildraum: Die Parameter müssen teils im Objektraum (Beleuchtung, Krümmung) und teils im Bildraum (Kontrastverbesserung, Merkmalslinien) berechnet werden. Zur Kombination ist es notwendig, sie in einen einheitlichen Arbeitsraum zu transferieren. Dafür bietet sich der Bildraum an, da sich jeder Objektraum-Parameter nach dem Prinzip der G-Buffer von SAITO und TAKAHASHI [1990] in den Bildraum projizieren lässt. Die Rücktransformation von Bildraumdaten auf die Objektgeometrie dagegen ist aufgrund des massiven Rechenaufwandes keine günstige Alternative.

Konstante/variable Werte: Einige Werte wie die Oberflächenkrümmung bleiben über die Dauer der Anwendung konstant und können vorberechnet werden. Da andere Parameter wie die Tiefe jedoch abhängig von der Blickrichtung und damit variabel sind, muss die Berechnung der Schattierung in jedem Frame erfolgen.

Die obigen Betrachtungen führen zu der Forderung nach einem Hybrid-System: Parameter werden sowohl im Objektraum als auch im Bildraum berechnet, aber nur im Bildraum kombiniert. Alle statischen Werte, die im Objektraum ermittelbar sind, sollten dabei in einem Vorverarbeitungsschritt berechnet werden, um den Rechenaufwand pro Frame so gering wie möglich zu halten. Aus der Kombination der Parameter im Bildraum ergibt sich die Schattierungskarte: eine Ansicht der Szene mit Kameraeinstellung und Auflösung des Renderers, in der jeder Pixel die gewünschte Helligkeit anzeigt. Aus ihr kann der Renderer die Helligkeit pixelweise auslesen und zur eigentlichen Darstellung mittels Stippling oder Hatching verwenden (siehe Abb. 3.21). Auf diese Weise kann auch Helligkeit oder Farbsättigung eines normalen Rendervorgangs gesteuert werden.

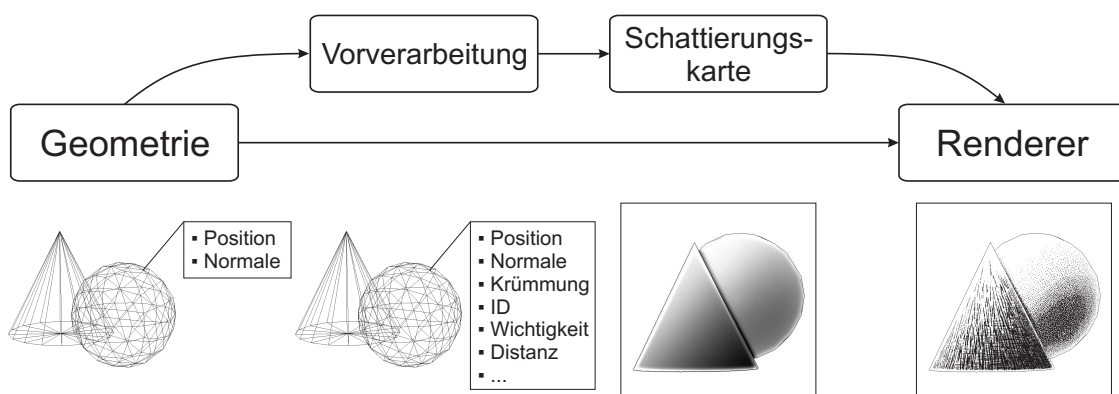


Abb. 3.21: Prinzip der erweiterten Renderpipeline: Zunächst wird die Geometrie mit zusätzlichen Werten versehen, die zur Berechnung der Schattierungskarte notwendig sind. Diese dient anschließend dem Renderer als Referenz für die Helligkeit.

Analog zu den oben erwähnten G-Buffern erfolgt die Erstellung der Schattierungskarte auf der Basis von Parameter-Buffern. Jeder Buffer stellt die Verteilung eines Parameters in der Szene dar und wird entweder direkt als Helligkeit interpretiert oder dient dazu, diese auf andere Weise zu verändern. Viele der Parameter-Buffer können direkt beim Rendern erzeugt werden, andere entstehen erst durch Bildverarbeitung aus anderen Buffern. Die Erzeugung der Schattierungskarte lässt sich demnach in drei Schritte unterteilen (siehe Abb. 3.22):

- Rendern von Parameter-Buffern aus der Szenengeometrie
- Berechnung weiterer Buffer auf Grundlage der gerenderten
- Kombination aller Buffer zur Schattierungskarte

Es ist anzumerken, dass der gesamte Vorgang für den Benutzer „unsichtbar“ abläuft, die Buffer und die Schattierungskarte also nicht auf dem Bildschirm zu sehen sind. Einige Aspekte dieses System werden nun näher betrachtet.

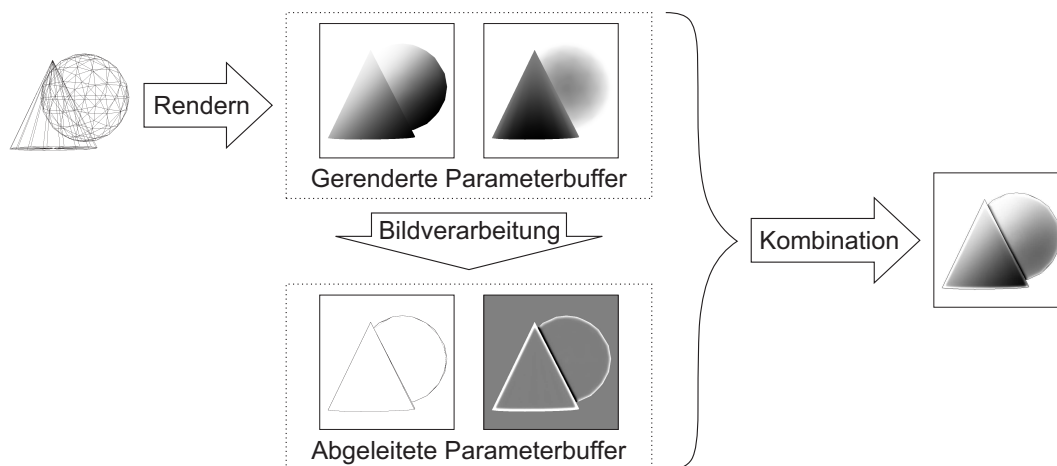


Abb. 3.22: Erzeugung der Schattierungskarte: Aus der Geometrie werden Parameter-Buffer erzeugt, weiterverarbeitet und zur Schattierungskarte kombiniert.

3.4.1 Kombination der Buffer

Die Zusammenführung aller Buffer zur Schattierungskarte hat einen wesentlichen Einfluss auf das Ergebnis. Im Folgenden wird eine Reihenfolge von Kombinationen vorgestellt, die sowohl durch Überlegung als auch empirisch als sinnvoll ermittelt wurde. Die Struktur der vorgeschlagenen Lösung orientiert sich an den Parameter-Kategorien aus Abschnitt 3.2 und wird entsprechend aufgeteilt.

In den folgenden Gleichungen werden Parameter-Buffer P und Schattierungskarte M als zweidimensionale Arrays von reellen Zahlen betrachtet, die dazugehörigen

Gewichtungsfaktoren w sind reelle Zahlen im Bereich $[0, 1]$. Der Einfachheit halber wird von einer einheitlichen Parametrisierung der Szene ausgegangen. Die Addition zweier Buffer $P = P_i + P_j$ wird elementweise durchgeführt:

$$\forall x, y : P(x, y) = P_i(x, y) + P_j(x, y) \quad (3.1)$$

Die Multiplikation $P = w \cdot P_i$ bedeutet entsprechend:

$$\forall x, y : P(x, y) = w \cdot P_i(x, y) \quad (3.2)$$

Falls eine differenzierte Parametrisierung erwünscht ist, kann statt eines konstanten Gewichtungsfaktors ebenso ein je nach Objekt wechselnder Wert angenommen werden. Da sich die Operationen auf Pixelebene abspielen, ist dies für die Gültigkeit der Gleichungen nicht relevant.

Oberflächengestaltung

Die Parameter-Buffer der Oberflächengestaltung repräsentieren direkt die Verteilung von Helligkeit. Wie auch bei HAMEL [2000] und AKERS et al. [2003] werden sie gewichtet aufaddiert, was generell in Beleuchtungsmodellen üblich ist:

$$M = \frac{1}{\sum w_i} \cdot \sum w_i \cdot P_i \quad (3.3)$$

P_i sind alle Parameter-Buffer dieser Kategorie, w_i ihre Gewichte. Die resultierende Schattierungskarte M ist normalisiert und wird im Anschluss einer Histogramm-äqualisation unterzogen, um den Kontrast zu verstärken (siehe Abb. 3.23). Dieser Schritt muss hier erfolgen, da alle folgenden Parameter die Helligkeit nur lokal verändern. Fügt man sie vor der Histogrammoperation hinzu, wirken sich diese lokalen Änderungen auf die gesamte Schattierungskarte aus: Die Erhellung des Hintergrundes durch atmosphärische Perspektive führt z. B. dazu, dass gleichzeitig der Vordergrund abgedunkelt wird, was jedoch nicht erwünscht ist.

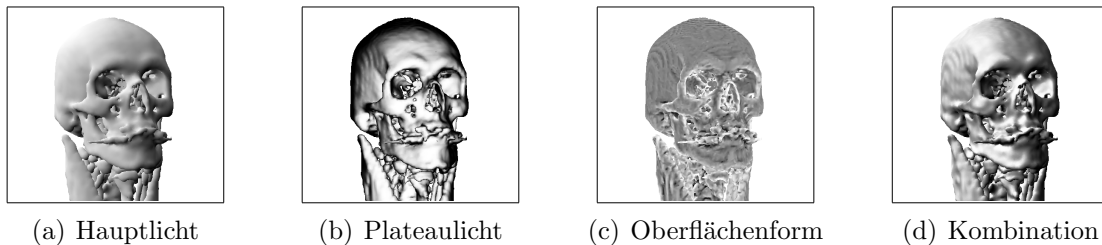


Abb. 3.23: Kombination der Parameter der Oberflächengestaltung: (d) ist die gewichtete Summe von (a), (b) und (c), zusätzlich wurde mittels Histogrammäqualisation der Kontrast verbessert.

Szeneninformationen

Die Parameter-Buffer der Szeneninformationen werden anders behandelt als die vorhergehenden, da sie nicht selbst Helligkeit repräsentieren, sondern abstrakte Information vermitteln. Sie dienen der Modifikation der vorläufigen Schattierungskarte. Zur Erläuterung wird der Parameter-Buffer der Wichtigkeit P_w betrachtet, das Prinzip ist aber ebenso auf die anderen Parameter anwendbar. P_w besteht aus den Wichtigkeitswerten, die für jedes Objekt definiert sind. Der einfachste Weg, sie in die Schattierungskarte zu integrieren, wäre der Einbezug als Helligkeit in die gewichtete Summe der Oberflächengestaltung. Da die Wichtigkeit für jedes Objekt konstant ist, entspricht ihr Beitrag jedoch dem Einsatz von ambienter Beleuchtung und führt dazu, dass mit zunehmender Gewichtung des Buffers der Kontrast sämtlicher Objekte abnimmt (siehe Abb. 3.24). Im Interesse der Objektdarstellung ist dies eher kontraproduktiv, des Weiteren ist die Identifikation von hellen und dunklen Objekten als wichtig bzw. unwichtig nicht zwangsläufig verständlich.

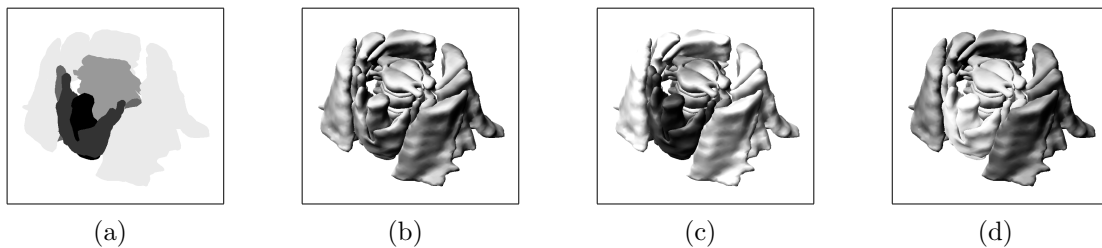


Abb. 3.24: Einbezug der Wichtigkeit in die gewichtete Summe: (a) Ursprüngliche Schattierungskarte. (b) Wichtigkeits-Buffer P_w . (c) und (d) P_w als Teil der gewichteten Summe. In (d) wurde P_w invertiert.

Eine weitere Möglichkeit besteht darin, den Objekten der Wichtigkeit zufolge eine unterschiedliche Grundhelligkeit zu geben, die mit dem Inhalt der Schattierungskarte moduliert wird. Der Gewichtungsfaktor bestimmt dabei die Stärke der Modulierung (siehe Abb. 3.25):

$$M' = (1 - w_w) \cdot M + w_w \cdot P_w \quad (3.4)$$

Dadurch erscheinen die einzelnen Objekte insgesamt unterschiedlich hell, weisen aber mehr Schattierung auf, um ihre Form erkenntlich zu machen. Das Ergebnis ist ausgeglichener als die gewichtete Addition, weist aber auch Nachteile auf. Bei zu kleinem w_w bleiben feine Abstufungen der Wichtigkeit unbemerkbar, da die Oberfläche an sich zu starke Helligkeitsunterschiede aufweist. Bei zu großem w_w dagegen nimmt die Erkennbarkeit der Objekt Oberfläche stark ab, ein Wert von 1 ergibt den Parameter-Buffer selbst. Zudem wird auch hier die Information nicht sehr intuitiv vermittelt.

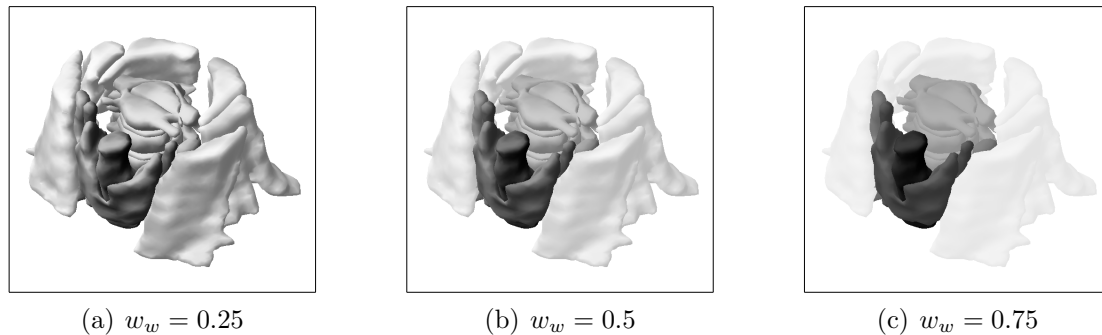


Abb. 3.25: Modulation des Wichtigkeits-Buffers mit der Schattierungskarte nach Gleichung 3.4

Die letzte und sinnvollste Möglichkeit besteht in der in Abschnitt 2.3 erwähnten Kontrastverringering nach Wichtigkeit. Dazu werden die Originalwerte der Karte abhängig vom Wichtigkeits-Buffer gegen 1 interpoliert. Dies ist auch die grundlegende Funktionsweise der atmosphärischen Perspektive, daher entspricht die Gleichung im Prinzip derjenigen, die von FOLEY et al. [1995] verwendet wird:

$$M' = (1 - w_w \cdot P_w) \cdot M + w_w \cdot P_w \quad (3.5)$$

So wird der Effekt des Ausblendens unwichtiger Objekte erreicht, der in Abschnitt 2.3.1 beschrieben wurde. Objekte mit hoher Wichtigkeit dagegen behalten ihre ausdrucksstarke Schattierung bei (siehe Abb. 3.26(a)). Mit dem Einsatz von Merkmalslinien wie in Abb. 3.26(b) ergibt sich eine ähnliche Abstufung wie bei TIETJEN et al. [2005], bei der Kontextobjekte nur durch ihre Silhouette, Fokusobjekte zusätzlich durch Schattierung der Oberfläche illustriert werden. Darüber hinaus kann auch die Breite der Merkmalslinien verändert werden, um den ausblendenden Effekt noch zu verstärken. Dies wird im nächsten Abschnitt behandelt.

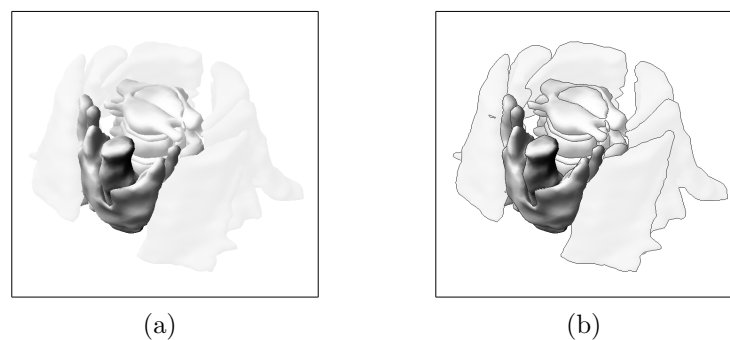


Abb. 3.26: Kontrastverringering der Schattierungskarte nach Wichtigkeit. (b) zusätzlich mit Merkmalslinien

Die zuletzt genannte Methode der Einbindung ist für alle Szeneninformationen geeignet. Bei Wichtigkeit und Objektabstand bietet sich auch der Einsatz von Transparenz anstelle von oder parallel zur Kontrastverringering an. Der Einbezug von Transparenz in die Pipeline wird in Abschnitt 3.4.3 näher betrachtet. Aus der Überschneidung dieser Methoden ergibt sich, dass nicht alle drei Parameter gleichzeitig mittels Schattierung dargestellt werden können, ohne die einzelnen Informationen unerkennbar zu machen. Ohne Zuhilfenahme weiterer Visualisierungsmittel kann also nur jeweils einer der genannten Parameter visualisiert werden.

Bildoptimierung

Die Parameter-Buffer der Bildoptimierung verlangen aufgrund ihrer sehr unterschiedlichen Ausprägung individuelle Betrachtung. Die Modifikationen sind dabei in der beschriebenen Reihenfolge durchzuführen, d. h. das Ergebnis M' einer Gleichung ist die Variable M der folgenden.

Kantenlicht und Überdeckungsschatten werden gesondert behandelt, obwohl sie Beleuchtung repräsentieren, da ihre Wirkung nur auf Objektränder beschränkt sein soll. Würden die Buffer in die gewichtete Summe aufgenommen, ergäbe dies eine globale Kontrastverringering wie bei den Parametern der Szeneninformationen. Stattdessen werden sie so angewandt, dass sie die Helligkeit lokal verändern (siehe Abb. 3.27). Das Kantenlicht wird direkt auf die vorläufige Schattierungskarte aufaddiert und das Ergebnis auf den Wertebereich $[0, 1]$ beschnitten:

$$M' = \underset{[0,1]}{\text{clamp}}(M + w_{kl} \cdot P_{kl}) \quad (3.6)$$

Der Buffer des Überdeckungsschattens wird nach der Gewichtung mit 1 addiert, so dass alle unbetroffenen Pixel den Wert 1 und negative Werte im Bereich des Schattens geringe positive Werte erhalten. Diese resultierenden Werte werden mit der Schattierungskarte multipliziert. Dadurch verschiebt sich die Helligkeit im Schattenbereich gegen 0, Pixel außerhalb des Schattens dagegen bleiben durch die Multiplikation mit 1 unverändert:

$$M' = (1 + w_{us} \cdot P_{us}) \cdot M \quad (3.7)$$

Die atmosphärische Perspektive wird mittels Kontrastverringering nach Gleichung 2.6 und damit ähnlich wie die Szeneninformationen realisiert (siehe Abb. 3.28). Hier ist zusätzlich die Hintergrundintensität $I_b \in [0, 1]$ frei wählbar:

$$M' = (1 - w_{ap} \cdot P_{ap}) \cdot M + w_{ap} \cdot P_{ap} \cdot I_b \quad (3.8)$$

Die Einbindung der Merkmalslinien erfolgt zuletzt, da sie von keinem anderen Effekt beeinflusst werden sollen. Dadurch wird garantiert, dass jedes Objekt wenigstens minimal durch sie repräsentiert ist. Auch sie können durch einen Gewichtungsfaktor in ihrer Stärke variiert werden, wodurch sie in der endgültigen Stippling-/Hatching-Darstellung einen lockeren, zeichnerischen Charakter erhalten. Da sie nach der Kantenfilterung zunächst weiß auf schwarz im Buffer auftreten, müssen sie im Anschluss an die Gewichtung invertiert werden. Sie werden über eine Minimum-Operation hinzugefügt, damit sie auf einer helleren Oberfläche dargestellt werden, eine dunklere dagegen nicht aufhellen:

$$M' = \min(1 - w_l \cdot P_l, M) \quad (3.9)$$

Weiter ist es möglich, atmosphärische Perspektive auch über die Breite der Merkmalslinien auszudrücken. Durch die Auswahl des Kantenfilters können Merkmalslinien mit unterschiedlicher Breite erzeugt werden. Zwischen diesen wird über den Wert der atmosphärischen Perspektive interpoliert (siehe Abb. 3.29(a)). Ihre Einbindung in die Schattierungskarte bei gleichzeitiger Interpolation ergibt sich dann wie folgt:

$$M' = \min\left((1 - w_{ap} \cdot P_{ap}) \cdot (1 - w_l \cdot P_l^s) + w_{ap} \cdot P_{ap} \cdot (1 - w_l \cdot P_l^t), M\right) \quad (3.10)$$

Dabei sind P_l^s und P_l^t Merkmalslinien-Buffer mit starken bzw. schwachen Linien. Sollen die Merkmalslinien auch mit einer der Szeneninformationen verändert werden, wird analog verfahren. Zusätzlich kann zwischen Merkmalslinien und Silhouetten interpoliert werden. Dadurch werden mit abnehmender Wichtigkeit auch die inneren Kanten der Objekte schwächer, so dass sie im Extremfall nur noch durch ihre Silhouette repräsentiert sind (siehe Abb. 3.29(b), Abb. 3.29(c)).

3.4.2 Vorverarbeitung

Wie erwähnt enthalten einige Techniken Berechnungen, die direkt auf der Objektgeometrie durchgeführt werden. Sofern die Ergebnisse statisch sind, können sie in einem Vorverarbeitungsschritt ermittelt werden, um den Rechenaufwand pro Frame zu minimieren. Dabei handelt es sich um die geglätteten Normalen und die Hauptkrümmungen κ_1 und κ_2 . Diese Werte müssen jedem Vertex mitgegeben werden, damit sie beim Rendern der Parameter-Buffer zur Verfügung stehen. Auf diese Weise sind auch andere Werte zu übermitteln, die nicht zu weiteren Berechnungen dienen, sondern in einen Parameter-Buffer gendert werden. Dazu gehören eine eindeutige, statische Objekt-ID sowie variable Werte, die nur selten verändert werden. Dies sind Wichtigkeit und Objektabstand: Beide werden nur aktualisiert, wenn der Benutzer neue Wichtigkeitswerte festlegt oder ein neues Referenzobjekt für die Abstandsberechnung auswählt.

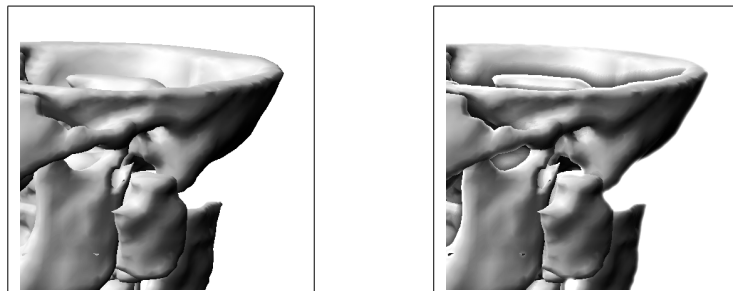
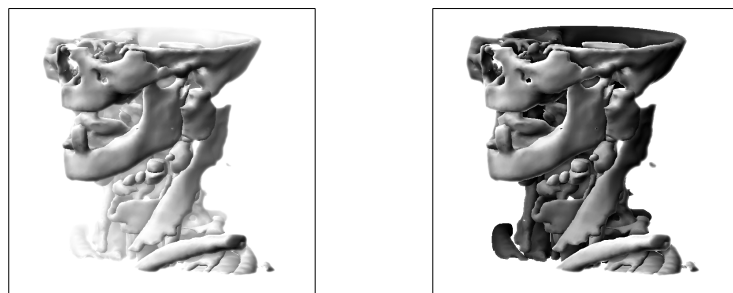


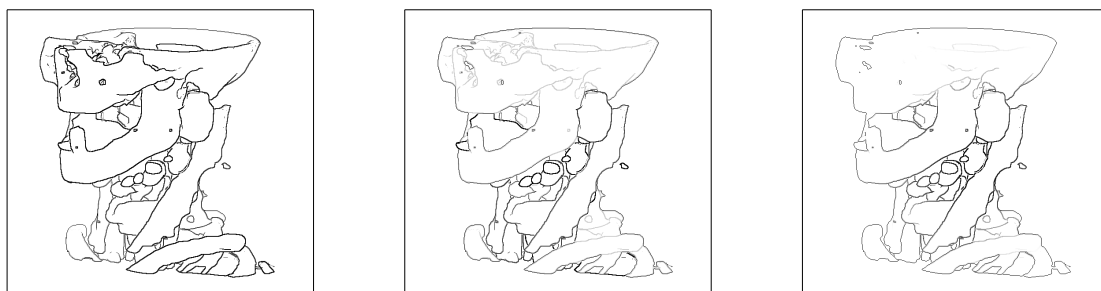
Abb. 3.27: Einbindung von Kantenlicht und Überdeckungsschatten. Links: Ursprüngliche Schattierungskarte. Rechts: Mit Kantenlicht und Überdeckungsschatten nach Gleichung 3.6 und 3.7



(a) $I_b = 1$

(b) $I_b = 0$

Abb. 3.28: Einbindung der atmosphärischen Perspektive nach Gleichung 3.8 mit unterschiedlicher Hintergrundintensität I_b



(a)

(b)

(c)

Abb. 3.29: Variation der Breite von Merkmalslinien nach verschiedenen Kriterien: (a) Atmosphärische Perspektive. (b) Wichtigkeit. (c) Abstand vom zentralen Lymphknoten. Bei (b) und (c) wird zusätzlich zwischen Merkmalslinien und Silhouetten überblendet.

3.4.3 Transparenz

Vor allem für die Operationsplanung ist es wichtig, Objekte transparent darstellen zu können. Für die Umsetzung im vorgeschlagenen System müssen Schattierungsinformationen auch für alle Objekte vorliegen, die hinter transparenten Objekten liegen. Dies ist durch eine Erweiterung der Pipeline möglich, die nun erläutert wird.

In der Shader-Programmierung existiert zur korrekten Darstellung transparenter Objekte die Technik des *Depth Peeling*, bei der eine Szene mit transparenten Objekten in opake Sichtbarkeits Ebenen zerlegt wird [EVERITT, 2001]. Hintere Ebenen enthalten diejenigen Objektteile, die in der jeweils vorderen Ebene verdeckt wurden. Anschließend werden die Ebenen wieder von hinten nach vorne unter Beachtung der Transparenz kombiniert (siehe Abb. 3.30).

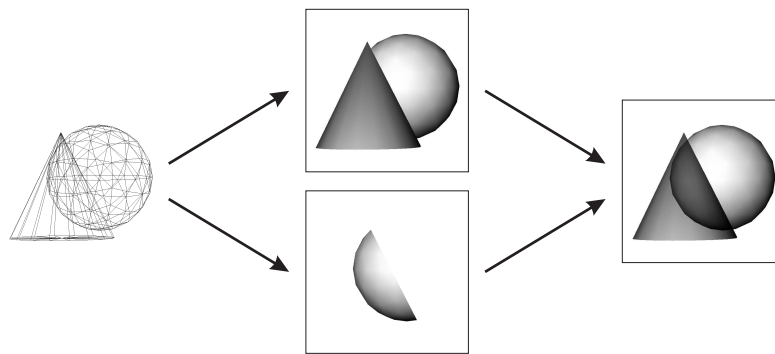


Abb. 3.30: Prinzip des Depth Peeling: Eine Szene wird in zwei Sichtbarkeits Ebenen zerlegt. Die zweite Ebene enthält nur Teile der Geometrie, die in der ersten Ebene verdeckt sind. Anschließend werden beide Ebenen wieder zusammengefügt, wobei das vordere Objekt korrekt transparent dargestellt werden kann.

Diese Technik kann auch auf die Schattierungskarte angewandt werden. Dazu wird Depth Peeling beim Rendern der Parameter-Buffer durchgeführt, so dass sämtliche Buffer einmal für jede Sichtbarkeits Ebene vorliegen. Sie werden nun parallel zu jeweils einer Schattierungskarte kombiniert (siehe Abb. 3.31). Der Stippling- oder Hatching-Renderer führt ebenfalls Depth Peeling durch und greift beim Schattieren der einzelnen Ebenen auf die entsprechende Schattierungskarte zu. Abb. 3.32 zeigt ein Beispiel der resultierenden Darstellung.

Da alle Ebenen außer der vordersten auch Teilobjekte enthalten können, entstehen möglicherweise falsche Kanten in den Buffern. Diese verfälschen wiederum die Helligkeitsverteilung, was bei der Verarbeitung beachtet werden muss. Auch ist es wichtig, bei der Histogrammäqualisation für alle Ebenen dieselbe Transformation zu benutzen, da sonst Diskontinuitäten in der Helligkeit eines halbverdeckten Objektes auftreten können. Sinnvollerweise sollte nur das Histogramm der vordersten Ebene ermittelt und zur Transformation aller Ebenen herangezogen werden.

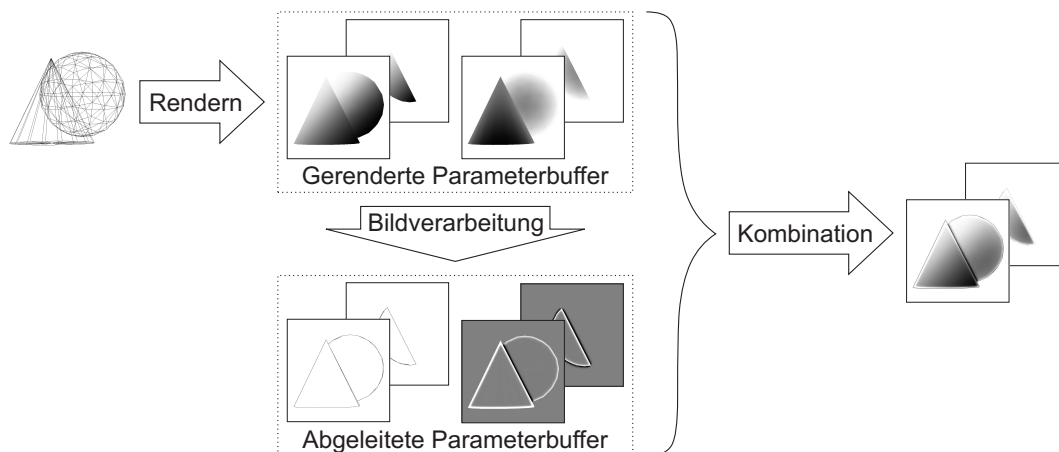


Abb. 3.31: Erzeugung von Schattierungskarten bei transparenten Objekten: Für jede Sichtbarkeitsebene werden separate Parameter-Buffer erzeugt, verarbeitet und kombiniert.

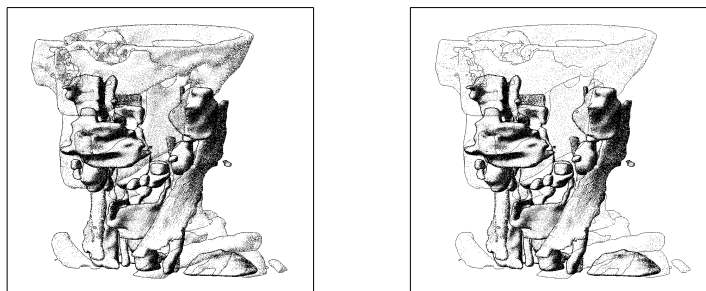


Abb. 3.32: Die Schädelknochen sind transparent dargestellt, zusätzlich werden sie durch einen geringen Wichtigkeitswert leicht (links) bzw. stark (rechts) abgeschwächt.

3.5 Zusammenfassung

In diesem Kapitel wurde das Konzept für ein System vorgestellt, welches eine komplexe illustrative Schattierung für medizinische 3D-Modelle ermöglicht. Die Schattierung setzt sich aus einzelnen Parametern zusammen, die jeweils einen Aspekt der Schattierung beinhalten und daher kombiniert werden müssen. Dazu wurden die Anforderungen an ein solches System analysiert, die aus den potentiellen Anwendungsbereichen Lehre und Operationsplanung hervorgehen. Um unterschiedliche Ausrichtungen der Anwendung zu ermöglichen, muss die Schattierung parametrisierbar sein, was durch diverse Variablen möglich ist. Dabei handelt es sich sowohl um Variablen, die die Umsetzung der Parameter steuern, als auch um die Gewichtung ihres Anteils an der Schattierung.

Aus der großen Menge an Einstellungsmöglichkeiten ergibt sich die Notwendigkeit, dem Benutzer die Aufgabe der Parametrisierung zu erleichtern. Zum einen kann die Menge an Variablen auf wenige einfache Einstellungen reduziert werden, was dem Benutzer eine effiziente und zielgerichtete Parametrisierung ermöglicht. Zum anderen können Templates vordefiniert werden, welche dem Benutzer Parametrisierungen für unterschiedliche Darstellungsarten vorgeben und bei Bedarf auf einfache Weise anpassbar sind. Es wurden geeignete Templates für verschiedene Strukturtypen und andere Kriterien vorgestellt und ihre Tauglichkeit an zahlreichen Abbildungen demonstriert. Als Testszenen standen Modelle aus segmentierten CT- und MRT-Daten zur Verfügung.

Des Weiteren wurde ein Verfahren für die Kombination aller Parameter entwickelt, das auf einer Verarbeitung im Bildraum beruht. Die resultierende Schattierung wird dabei in Form einer Schattierungskarte in die Renderpipeline eingefügt. Unter Betrachtung ihrer jeweiligen Eigenschaften wurde erläutert, wie die Parameter mathematisch zur Schattierungskarte kombiniert werden. Abschließend wurde betrachtet, wie die Rechenintensität durch Vorberechnungen reduziert und das System auf Szenen mit transparenten Objekten erweitert werden kann.

4 Implementierung

Mit zunehmender Rechenleistung von GPUs wird es immer gebräuchlicher, neben dem Rendern von 3D-Geometrie auch einfache 2D-Bildverarbeitung auf der Grafikkarte zu betreiben. Komposition von Bildern sowie zahlreiche Algorithmen der Bildverarbeitung können mit Shader-Programmierung umgesetzt werden. Dank hoher Parallelität der Rechenoperationen ergibt sich ein deutlicher Geschwindigkeitsvorteil gegenüber der Bearbeitung auf der CPU. Die in Abschnitt 3.4 beschriebene Erstellung der Schattierungskarte wird daher mit den Mitteln programmierbarer Grafik-Hardware umgesetzt.

Zunächst wird ein Überblick über die Soft- und Hardware gegeben, die zur Implementierung des konzipierten Systems verwendet wurde. Anschließend werden die Vorverarbeitung der Geometrie, die Erstellung der Schattierungskarte auf der GPU sowie die Integration mit einem illustrativen Renderer zu einem lauffähigen Gesamtsystem beschrieben.

4.1 Entwicklungsumgebung

Als Laufzeitumgebung dient MEVISLAB 1.5.1, eine Prototyping-Plattform des Bremer Forschungszentrums MeVis Research, die die grafische Entwicklung von Anwendungen der medizinischen Visualisierung und Bildverarbeitung ermöglicht [MEVIS RESEARCH, 2007]. Das Konzept der Plattform besteht im Aufbau von Netzwerken aus einfachen Modulen, die jeweils einzelne Aktionen wie Laden, Manipulieren und Anzeigen medizinischer Datensätze ausführen. Es sind eine Vielzahl von Algorithmen und Darstellungstechniken vorhanden, was die Umsetzung der entworfenen Renderpipeline stark vereinfacht. Bei Bedarf können weitere Module programmiert und direkt eingebunden werden. Zudem ist es möglich, ganze Netzwerke zu Makromodulen zusammenzufassen. Zur Darstellung und Exploration kompletter medizinischer Szenen dient das MEDICAL EXPLORATION TOOLKIT (METK), eine Gruppe von Makromodulen, die unter anderem die Auswahl verschiedener Rendermethoden erlauben [TIETJEN et al., 2008]. Das Konzept der vorliegenden Arbeit wurde mittels mehrerer Module umgesetzt und in das METK integriert.

MEVISLAB basiert auf mehreren Bibliotheken. Für die Anzeige, Verarbeitung und Segmentierung von Bilddaten stellt die *MeVis Image Processing Library* (ML) umfangreiche Methoden bereit. Zur Bearbeitung und Anzeige von 3D-Daten dagegen dient *Open Inventor*, eine Grafikkbibliothek auf Grundlage von OpenGL. Open Inventor stellt Funktionalitäten zur Verfügung, mit der 3D-Geometrie in Form eines Szenengraphen organisiert und gerendert werden kann. Beide Bibliotheken sind in C++ umgesetzt. Die Programmierung von Modulen erfolgt daher sowohl auf den abstrakteren Ebenen von ML und Open Inventor als auch in C++ und OpenGL.

Mit der MLWEM-Bibliothek steht eine weitere geometrische Datenstruktur zur Verfügung: das *Winged Edge Mesh* (WEM), das auf BAUMGART [1972] zurückgeht. Dabei handelt es sich um eine Polygonstruktur, die den schnellen Zugriff von allen geometrischen Primitiven auf deren Nachbarn erlaubt. Dies ist für bestimmte Algorithmen erforderlich, z. B. bei der Berechnung der Oberflächenkrümmung. Die Implementierung von WEMs in MEVISLAB ermöglicht das Speichern von zusätzlichen Werten pro Vertex, den sogenannten *UserNodeValues*, die für weitere Berechnungen verwendet werden können.

Für die Shader-Programmierung wird die *OpenGL Shading Language* (GLSL) verwendet, die als Hochsprache für die Benutzung mit OpenGL entwickelt wurde. Informationen können auf zwei Arten von OpenGL an die Shader-Programme übermittelt werden: durch Angabe von Werten pro Vertex (*Attribute*) und pro Rendervorgang (*Uniform-Variablen*). Attribute sind beispielsweise Position und Normalen von Vertices. OpenGL-Zustände wie Transformationsmatrizen, die für die Dauer eines Rendervorgangs konstant sind, werden dagegen als Uniform-Variablen übertragen. Eine spezielle Form von Uniform-Variablen sind Textursampler, über die ein Shader Lesezugriff auf Texturen bekommt.

Eine Erweiterung von OpenGL sind Framebuffer Objects (FBO), die von neueren Grafikkarten unterstützt werden [JULIANO und SANDMEL, 2007]. Sie stellen ein Interface dar, mit dem die Ausgabe der Renderpipeline in eigens definierte Buffer geleitet werden kann, was eine einfache Form des Offscreen-Renderns ermöglicht. Insbesondere erlauben sie das Rendern in Texturen, die daraufhin in weiteren Rendervorgängen verwendbar sind. Durch Verwendung von Gleitkomma-Texturen mit 16- oder 32-Bit-Komponenten sind genauere Berechnungen ohne Beschränkung auf den Wertebereich $[0, 1]$ möglich. Jedoch werden hardwareabhängig nicht alle Operationen auf allen Texturformaten unterstützt, was zu numerischen Fehlern führen kann. Dies wird später genauer ausgeführt.

Die Implementierung erfolgte auf einem INTEL PENTIUM 4 Prozessor mit 3,2 GHz, 1 GB Arbeitsspeicher und einer NVIDIA GEFORCE 7800 GS Grafikkarte mit 512 MB Speicher, die das Shader-Model 3.0 unterstützt. Laufzeit-Tests wurden auf einer NVIDIA GEFORCE 7900 GS unter ansonsten gleichen Systemvoraussetzungen durchgeführt.

4.2 Vorverarbeitung

Die Geometrie der Strukturen, die mit dem METK dargestellt werden, liegt im Open-Inventor-Format vor. Es enthält Position, Normale, Farbe und Texturkoordinaten der als Dreiecksnetz organisierten Vertices. Wie in Abschnitt 3.4.2 erläutert wurde, werden darüber hinaus beim Rendern der Parameter-Buffer weitere Werte pro Vertex benötigt. Zur Berechnung von Krümmung und Normalenglättung sind jedoch Informationen über die Nachbarschaft von Vertices erforderlich, die in einfachen Dreiecksnetzen nicht vorliegen und eine andere Datenstruktur verlangen.

Hierfür werden WEMs herangezogen, welche durch die enthaltenen topologischen Informationen diese Berechnungen ermöglichen. Die Krümmungsberechnung wurde von SCHULZ [2005] mit dem Modul `WEMCurvature` umgesetzt. Zur Berechnung von Distanzen existiert das Modul `WEMSurfaceDistance`, das für alle Vertices eines Objektes den minimalen Abstand zu einem zweiten Objekt ermittelt. Damit ist die Szeneninformation über Abstand von einer Struktur umsetzbar. WEMs eignen sich daher für die Vorverarbeitung der Geometrie und Speicherung der erhaltenen Werte als `UserNodeValues`. Dieser Schritt wurde mit dem Modul `WEMInfoGenerator` umgesetzt. Auch Wichtigkeit und Objekt-ID, die nicht berechnet werden müssen, werden hier für jeden Vertex gespeichert. Zur Konvertierung von Open-Inventor-Geometrie in ein WEM liegt das Modul `UMDWEMConvertInventor` vor. In MEVISLAB sind WEMs jedoch nicht direkt darstellbar und müssen zum Rendern wieder in Open-Inventor-Geometrie umgewandelt werden.

Die `UserNodeValues` werden zusammen mit der Geometrie in der Renderpipeline benötigt und sollten daher mit OpenGL als Shader-Attribute übergeben werden. Da keine Datenstruktur in MEVISLAB die Angabe von Attributen ermöglicht, wurde die neue Datenstruktur *Attributed Indexed Triangle Strip Set* (AITSS) implementiert. Sie basiert auf der Open-Inventor-Klasse `SoIndexedTriangleStripSet`: Die Vertices sind in indizierten Listen gespeichert und in Reihen aneinanderhängender Dreiecke organisiert, wodurch sie sehr effizient mit OpenGL gerendert werden können. Die Struktur kann ebenfalls `UserNodeValues` für jeden Vertex speichern, die beim Rendern als Shader-Attribute übergeben werden. Für die Konvertierung von WEM in AITSS wurde das Modul `SoWEM2AttribITSS` implementiert, das auch alle `UserNodeValues` überträgt.

Es erscheint zunächst sehr aufwändig, die Open-Inventor-Geometrie in ein WEM und dieses wiederum in ein AITSS zu konvertieren. Da beide Schritte jedoch nur beim Laden der Geometrie auszuführen sind, handelt es sich um einen einmaligen Rechenaufwand, der keinen negativen Einfluss auf die interaktive Darstellung hat. Falls sich Wichtigkeit oder Objektabstand durch Benutzerinteraktion im `WEMInfoGenerator` ändern, sorgt `SoWEM2AttribITSS` dafür, die neuen Werte im zugehörigen AITSS ohne erneute Konvertierung zu aktualisieren.

Die Vorverarbeitung eines geometrischen Objektes besteht demnach aus folgenden Arbeitsschritten, die in Abb. 4.1 an einem einfachen Netzwerk aus MEVISLAB-Modulen dargestellt sind:

1. Laden der Geometrie aus Open-Inventor-Datei
2. Konvertierung in WEM
3. Berechnung und Speicherung der UserNodeValues
4. Konvertierung in AITSS
5. Rendern des AITSS mit zusätzlichen Shader-Attributen

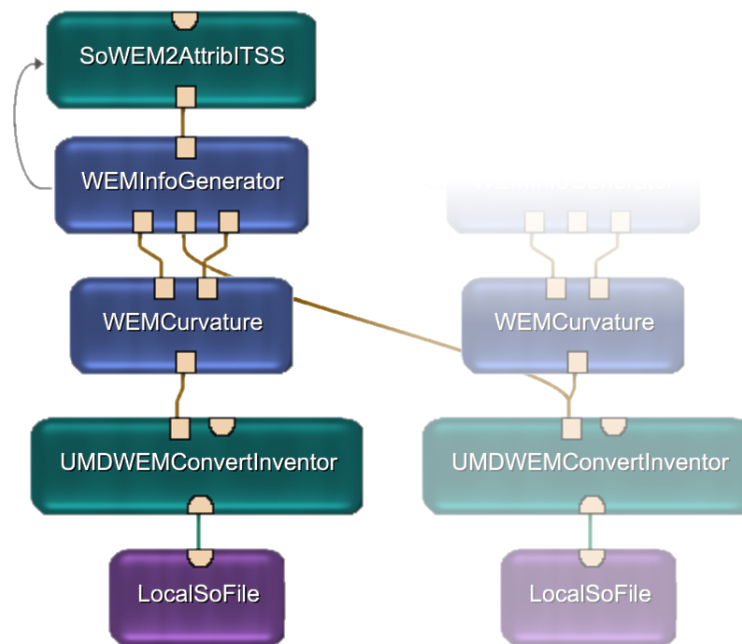


Abb. 4.1: MEVISLAB-Netzwerk zur Vorverarbeitung der Geometrie (der Informationsfluss verläuft von unten nach oben): Das Modul `LocalSoFile` lädt Geometrie aus einer Open-Inventor-Datei. `UMDWEMConvertInventor` wandelt die Geometrie in ein WEM um. `WEMCurvature` berechnet die Krümmung der Oberfläche, die in `WEMInfoGenerator` den einzelnen Vertices als Wert zugewiesen wird. Dort werden auch alle weiteren Werte berechnet und als `UserNodeValues` gespeichert. Die rechte Geometrie dient als Referenzobjekt für die Abstandsmessung. `SoWEM2AttribITSS` wandelt das WEM wieder in Open-Inventor-Geometrie um, die beim Rendern die zusätzlichen Werte als Shader-Attribute übergibt.

4.3 Programmierung der Grafik-Hardware

Nach der Vorverarbeitung wird aus der Geometrie die Schattierungskarte erzeugt. Mit den Mitteln der Shader-Programmierung kann dies auf der Grafikkarte durchgeführt werden. Die vorberechneten Werte werden mit der Geometrie als Shader-Attribute übermittelt. Das Offscreen-Rendern der Parameter-Buffer erfolgt über FBOs. Durch sie werden die Parameter in Texturen abgebildet, die über Textursampler in einem weiteren Rendervorgang verarbeitet werden. Es ist also eine Abfolge von Rendervorgängen realisierbar, bei der das Ergebnis des vorigen Rendervorgangs im nächsten weiterverarbeitet wird (siehe Abb. 4.2). Neuere Grafikkarten erlauben das gleichzeitige Rendern in mehrere Texturen, was die Pipeline weiter beschleunigt, da so nicht für jeden Parameter-Buffer ein eigener Rendervorgang nötig ist. Alle relevanten Variablen für die Parameterberechnung sowie ihre Gewichtungsfaktoren können den Shadern als Uniform-Variablen übergeben werden.

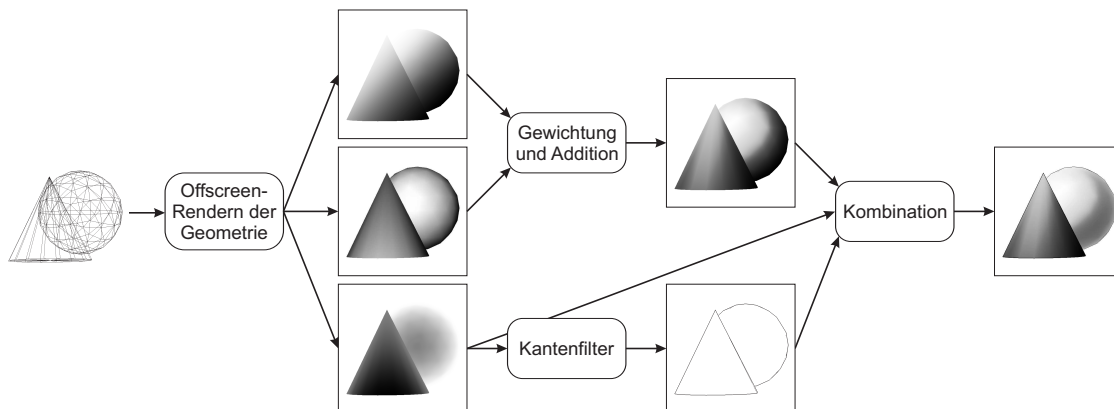


Abb. 4.2: Jeder Rendervorgang speichert seine Ausgabe in eine oder mehrere Texturen, die wiederum in Form von Textursamplern als Eingabe für weitere Rendervorgänge dienen.

Da die verwendeten Stippling-/Hatching-Renderer ebenfalls mit Shadern realisiert sind, können sie direkt auf die Schattierungskarte zugreifen. Die Verarbeitung auf der Grafikkarte bringt nicht nur wegen der hohen Geschwindigkeit der GPU Vorteile. Der Datentransfer von der Grafikkarte zur CPU verläuft sehr langsam, daher hätte schon das Auslesen aller Buffer durch die CPU starke Geschwindigkeitseinbußen zur Folge. Mit einer rein GPU-basierten Pipeline entfällt dieser Engpass.

4.3.1 Effiziente Nutzung von Texturen

Im ersten Rendervorgang wird die Geometrie der Szene in Parameter-Buffer gerendert. Mit der verwendeten Grafikkarte ist das Rendern in vier Texturen gleichzeitig möglich, es müssen jedoch neun Parameter-Buffer erzeugt werden. Um dies in nur

einem Vorgang durchzuführen, werden mehrere Buffer in einer Textur zusammengelegt. Jeder Parameter definiert nur einen Grauwert pro Pixel, daher können alle vier Kanäle einer Farbtextur (Rot, Grün, Blau und Alpha) für je einen Parameter verwendet werden. (siehe Abb. 4.3). Dadurch ist es möglich, bis zu 16 Parameter-Buffer parallel zu erstellen. Eine geschickte Bündelung der Parameter in Texturen ist auch für die Laufzeit der Shader von Vorteil: Benötigt ein Shader die Werte von vier Parametern und sind diese in einer Textur zusammengefasst, können sie im Shader mit nur einem Texturzugriff ausgelesen werden.

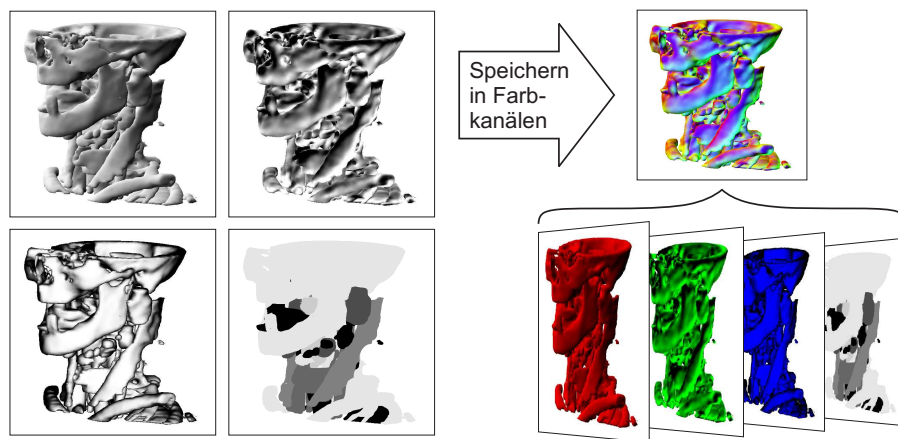


Abb. 4.3: Ausnutzung der Kanäle einer RGBA-Textur: In jedem Kanal wird ein Parameter gespeichert, um die Renderpipeline effizienter zu gestalten.

4.3.2 Bildverarbeitung

Bildverarbeitung ist auf der Grafikkarte sehr einfach zu realisieren. Durch Rendern eines Rechtecks über die gesamte Viewportgröße wird genau ein Fragment pro Pixel der Ausgabertextur erzeugt und jeder Pixel genau einmal beschrieben. Über Textursampler können Werte aus anderen Texturen verwendet werden, um den Ausgabewert zu berechnen. Haben Ein- und Ausgabertextur dieselbe Dimension, kann nun für jeden gerenderten Pixel der entsprechende Pixel der Eingabetextur gelesen und manipuliert werden, was eine grundlegende Bildverarbeitung ermöglicht. Ebenso können mehrere Texturen kombiniert werden, wodurch die gewichtete Aufsummierung und weitere Manipulationen möglich sind.

Bildfilterung ist ebenso möglich, indem zur Berechnung des Ausgabewertes auch auf andere Pixel der Eingabetextur zugegriffen wird. Theoretisch sind so beliebig große Filter denkbar. Da ein Texturzugriff jedoch relativ viel Rechenzeit verbraucht, sind sie nur bis zu einer bestimmten Größe rentabel. Für die hier benötigten Glättungs- und Kantenfilter ist dies ausreichend.

Glättungsfilter

Die Glättung des z -Buffers erfolgt wie bei LUFT et al. [2006] mit einem Gauß-Filter. Für einen sichtbaren Effekt wird ein relativ großer Kernel benötigt. Bei einem zweidimensionalen Filterkernel führt dies zu einer hohen Anzahl an Texturzugriffen, die die Echtzeitfähigkeit des Systems beeinträchtigen. Da der Gauß-Filter separabel ist, kann er jedoch in zwei eindimensionale Kernel zerlegt werden. Die Glättung erfolgt daher in zwei Rendervorgängen, in denen der Filter jeweils in x - bzw. y -Richtung angewandt wird. Dies verringert deutlich die benötigten Texturzugriffe und beschleunigt die Berechnung, ohne das Ergebnis zu verändern.

Die Breite des Glättungsfilters ist variabel, da er die Ausdehnung von Kantenlicht und Überdeckungsschatten festlegt. Um beide Parameter getrennt zu steuern, werden gleichzeitig zwei Filter mit unterschiedlicher Breite angewandt. Beide Filterkernel werden dynamisch berechnet und dem Shader zusammen mit ihren Breiten als Uniform-Variablen übergeben. Die Filterung verläuft wie bei üblicher CPU-Programmierung mit einer Schleife. Die verwendete Grafikkarte erlaubt jedoch keine variablen Schleifendurchläufe, daher muss eine konstante Zahl an Iterationen durchgeführt werden. Beide Filter werden in der Schleife nur berechnet, solange die Iterationsvariable kleiner als die jeweilige Breite ist. Da also immer die maximale Anzahl an Iterationen ausgeführt wird, sollte diese nicht zu hoch angesetzt werden. In der vorliegenden Implementierung wurde eine Obergrenze von 60 gewählt, womit noch ausreichende Frame-Raten erreicht werden. Bei zukünftiger Grafik-Hardware stellt diese Einschränkung eventuell kein Problem mehr dar.

Um die Tiefenmaske zu erhalten, muss die Differenz zwischen originalem und geglättetem z -Buffer gebildet werden. Dies wird in einer späteren Stufe der Verarbeitung erledigt, die Ausgabetextur des Glättungsfilters enthält daher sowohl den originalen z -Buffer als auch beide geglätteten Versionen in den ersten drei Kanälen.

Kantenfilter

Für die Kantenfilterung wurde der Laplace-Filter gewählt. Er wird gleichzeitig auf ID- und z -Buffer angewandt und berechnet dadurch sowohl Silhouetten als auch Merkmalslinien. Durch einen Schwellenwert werden die Linien eindeutig auf den Wert 1 gesetzt. Da in den Merkmalslinien auch die Silhouetten enthalten sind, können beide in der Ausgabetextur zusammengelegt werden, indem die inneren Kanten mit Wert 0.5 gespeichert werden. Um sie wieder zu trennen, genügt eine Schwellenwertoperation, wobei ein Schwellenwert > 0.5 die Silhouetten bzw. < 0.5 die Merkmalslinien ergibt.

Zur Erzeugung unterschiedlicher Linienstärken werden die folgenden vier Filter-Kernel verwendet, die parallel in einer Schleife angewandt werden können. Die Resultate werden in den vier Kanälen einer Textur gespeichert, die somit Merkmalslinien- und Silhouetten-Buffer mit je vier Linienstärken enthält. Wegen der geringen Anzahl an Texturzugriffen stellt der Filter kein Laufzeit-Problem dar.

$$\begin{aligned}
 L_1 &= \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, & L_2 &= \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\
 L_3 &= \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & -12 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, & L_4 &= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -24 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad (4.1)
 \end{aligned}$$

Der Laplace-Filter erzeugt auf beiden Seiten einer Kante jeweils ein Maximum mit entgegengesetztem Vorzeichen. Als Merkmalslinie ist nur die Kante auf dem jeweils vorderen Objekt erwünscht. Mit den oben angegebenen Filtern ist bei der Filterung des z -Buffers immer der positive Wert als Kante zu betrachten, da vordere Objekte einen geringeren Tiefenwert besitzen. Bei der Filterung des ID-Buffers dagegen ist dies nicht eindeutig: Vordere Objekte können eine höhere oder niedrigere ID besitzen als hintere Objekte. Entsprechend wäre jeweils zu entscheiden, ob die positive oder die negative Kante die gesuchte ist (siehe Abb. 4.4).

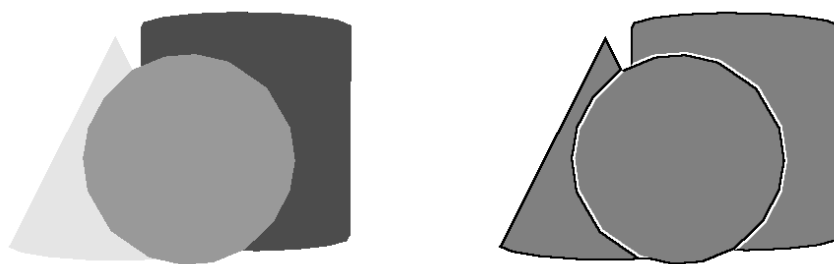


Abb. 4.4: Problem bei der Filterung des ID-Buffers: Das vordere Objekt liegt vor Objekten mit höherer und niedrigerer ID (links). Dadurch ist einmal die positive, einmal die negative Kante korrekt (rechts). Im rechten Bild entspricht grau einem Wert von 0, positive und negative Werte sind mit weiß bzw. schwarz gekennzeichnet.

Das Problem lässt sich beheben, indem für jede Filteroperation die IDs temporär so verändert werden, dass sie mit der Tiefe der Szene ansteigen (siehe Abb. 4.5). Dadurch entstehen an jedem Pixel optimale Bedingungen für den Filter und es kann ebenfalls immer die positive Kante als Silhouette gewählt werden.

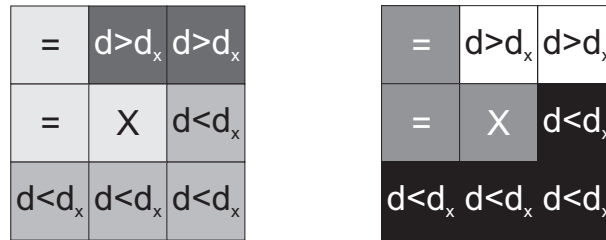


Abb. 4.5: Modifizierung des ID-Buffers: Links ist der aktuell zu filternde Ausschnitt des ID-Buffers zu sehen. Alle Pixel mit derselben ID wie X werden für die Berechnung auf 0.5 gesetzt. Das obere Objekt ist weiter entfernt und erhält die ID 1, das untere Objekt liegt näher und erhält ID 0. Rechts sind die ID-Werte dargestellt, die temporär für die Filteroperation verwendet werden.

4.3.3 Histogrammerzeugung

Für die Erzeugung des Histogramms auf der Grafikkarte wird das Verfahren von SCHEUERMANN und HENSLEY [2007] verwendet, das in Abschnitt 2.2.4 vorgestellt wurde. Es wird auf die Textur angewandt, die die gewichtete Summe der Oberflächengestaltungs-Parameter enthält, im Folgenden als Summentextur bezeichnet. Für die Histogrammäqualisation wird auch das kumulative Histogramm benötigt. Auf der Grafikkarte ist dies nur mit Hilfe mehrerer Rendervorgänge umzusetzen, weshalb diese Berechnung auf der CPU durchgeführt wird. Dafür muss die Histogramm-Textur von der Grafikkarte zur CPU transferiert werden. Da es sich jedoch im Gegensatz zu einem kompletten Frame um eine sehr geringe Datenmenge handelt, hat dies keinen nachteiligen Einfluss auf die Laufzeit.

Wie bereits erwähnt, wird für die Histogrammberechnung die Eingabetextur als Vertexliste interpretiert. Die Farbwerte (r, g, b, a) jedes Pixels entsprechen also den homogenen Koordinaten (x, y, z, w) eines Vertex. Der Vertexshader muss diese homogenen Koordinaten in Clip-Koordinaten (x_c, y_c, z_c, w_c) umrechnen. In der Summentextur ist nur ein Farbkanal für die Übermittlung des Grauwertes nötig, wofür der rote Kanal gewählt wird. Da somit x dem Grauwert des Pixels entspricht, werden die Vertices mit folgender Gleichung nach zunehmender Helligkeit sortiert und auf alle Pixel der Histogrammtextur verteilt:

$$(x_c, y_c, z_c, w_c) = (2 \cdot x - 1, 0, 0, 1) \quad (4.2)$$

Da das Verfahren die gesamte Summentextur einbezieht, verfälscht der Hintergrund der Szene das Histogramm und muss herausgefiltert werden. Vertices können im Vertexshader entfernt werden, indem w_c auf 0 gesetzt wird: Nach dem Vertexshader werden die Clip-Koordinaten in die euklidischen Koordinaten $\left(\frac{x_c}{w_c}, \frac{y_c}{w_c}, \frac{z_c}{w_c}\right)$ umgerechnet. Mit $w_c = 0$ erhält der Vertex keine sinnvollen Koordinaten und ist damit effektiv aus der Pipeline eliminiert. Um den Hintergrund zu entfernen, muss beim Offscreen-Rendern der Geometrie ein Stencil-Buffer angelegt werden, der alle

Objekte weiß und den Hintergrund schwarz anzeigt. Dessen Werte werden bei der Gewichtung in den Alpha-Kanal der Summentextur kopiert und im Histogramm-Vertexshader an w_c übergeben. So erhalten alle Vertices, die Hintergrundpixeln entsprechen, für w_c den Wert 0 und fließen nicht in das Histogramm ein.

Da nur die ungefähre Helligkeitsverteilung benötigt wird, reicht es aus, eine Version der Summentextur mit geringerer Auflösung für das Histogramm heranzuziehen. Dies kann auf dieselbe Weise wie die Filterung des Hintergrundes realisiert werden. Der grüne Kanal der Summentextur erhält bei der Gewichtung den Wert 1, wenn Zeile und Spalte des Pixels ungerade sind, ansonsten 0. Durch folgende Koordinatenzuweisung werden zusätzlich zum Hintergrund drei Viertel aller Vordergrundpixel beseitigt, es entsteht das Histogramm der Textur mit halber Auflösung:

$$(x_c, y_c, z_c, w_c) = (2x - 1, 0, 0, y \cdot w) \quad (4.3)$$

Ein weiteres Problem stellt die Genauigkeit der Histogrammtextur dar. Um den Überlauf von Histogrammklassen zu verhindern, sollte eine möglichst hohe Farbtiefe für die Textur gewählt werden. Bei der verwendeten Grafikkarte ist Blending auf Gleitkomma-Texturen mit maximal 16 Bit pro Kanal möglich. Bei 16-Bit-Gleitkommazahlen ergibt die Addition von 2048 und 1 wieder 2048. Somit können durch wiederholtes Addieren von 1 keine größeren Werte in der Textur dargestellt werden, was das Histogramm eventuell stark verzerrt. Es gibt zwei Ansätze zur Abhilfe: Zum einen kann die Textur mit einem Wert von -2047 initialisiert werden, was das Aufsummieren von 2047 zusätzlichen Vertices erlaubt. Zum anderen können die Vertices auf die vier Kanäle der Textur verteilt und bei der Auswertung auf der CPU addiert werden, was auch von SCHEUERMANN und HENSLEY [2007] vorgeschlagen wird. Bei perfekter Gleichverteilung erhöht sich der maximal darstellbare Wert pro Histogrammklasse damit auf $(2048 + 2047) \cdot 4 = 16380$, wodurch der Überlauf einer Klasse weitestgehend vermieden wird. Die Verteilung der Vertices muss ebenfalls bei der Gewichtung in der Summentextur festgelegt werden, da im Vertexshader keine Möglichkeit besteht, die schon bearbeiteten Vertices zu zählen. Dafür wird der blaue Kanal verwendet, dessen Wert b sich für den Pixel mit Spaltennummer x_p wie folgt berechnet:

$$b = \left\lfloor \frac{x_p}{2} \right\rfloor \bmod 4 \quad (4.4)$$

Dadurch erhält je ein Viertel aller Spalten der Textur einen der Ganzzahlwerte im Intervall $[0, 3]$. Dieser entscheidet, mit welcher Farbe der zugehörige Vertex in der Histogrammtextur gespeichert werden soll, wodurch deren Farbkanäle annähernd gleichmäßig genutzt werden. Da jeweils zwei nebeneinander liegende Spalten mit gerader und ungerader Nummer denselben Wert erhalten, funktioniert die Verteilung unabhängig davon, ob das Histogramm von der kompletten Textur oder der Version mit halber Auflösung erzeugt wird.

4.3.4 Framework in OpenGL

Die benötigten Shader, FBOs und Texturen werden mit OpenGL-Befehlen erzeugt und kontrolliert, was in dem neu implementierten Modul `SoShadingMap` geschieht. Um dessen Programmierung zu vereinfachen, wurde die C++-Klasse `OffscreenRenderPackage` entworfen, die das Ausführen eines anpassbaren Offscreen-Rendervorgangs erleichtert. Eine Instanz dieser Klasse (im Folgenden ORP genannt) enthält folgende Membervariablen:

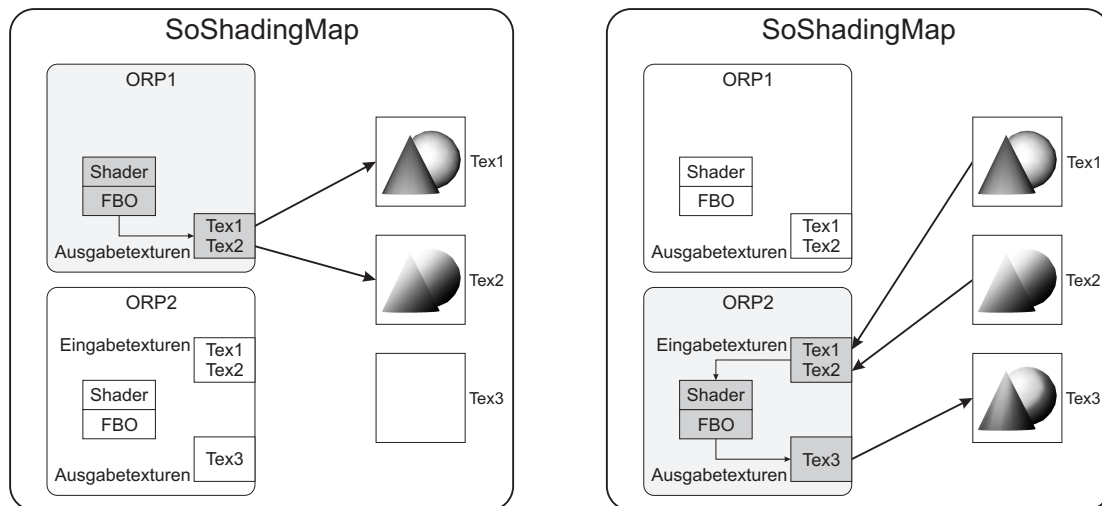
- Framebuffer Object
- Shader-Programm
- Liste mit Zeigern zu Eingabetexturen
- Liste mit Zeigern zu Ausgabertexturen
- Liste mit Werten von Uniform-Variablen

Texturen dienen gleichzeitig als Ein- und Ausgabe für verschiedene ORPs, daher sind sie keine Membervariablen von `OffscreenRenderPackage`, sondern werden im Hauptprogramm von `SoShadingMap` erzeugt und über Zeiger referenziert. Bei der Initialisierung einer ORP-Instanz muss mindestens eine Ausgabetextur angegeben werden, da sonst kein Ziel für den Rendervorgang existiert, Eingabetexturen dagegen sind optional. Des Weiteren können bei der Initialisierung Dateien mit Quellcode für Vertex- und Fragmentshader übergeben werden, ansonsten wird die Standard-Renderpipeline verwendet. Ebenfalls optional ist die Angabe der Werte von Uniform-Variablen, die das Shader-Programm benötigt.

Durch den Aufruf der Klassenfunktion `createFramebuffer(x,y)` werden das FBO erzeugt und alle Ausgabertexturen auf die Größe (x,y) gesetzt. `createShader()` kompiliert die übergebenen Quellcodes und setzt sie zu einem Shader-Programm zusammen. Damit ist das ORP bereit zum Rendern. Der Rendervorgang selbst, der in Abb. 4.6 schematisch dargestellt ist, wird in jedem Frame mit folgenden Funktionen ausgeführt:

1. `bindPackage()` aktiviert FBO und Shader, bindet die Ein- und Ausgabertexturen und aktualisiert die Werte der Uniform-Variablen
2. `render()` erstellt ein Rechteck in Größe des aktuellen Viewports und rendert es in die angegebenen Texturen
3. `unbindPackage()` setzt den Framebuffer und die Renderpipeline wieder auf den System-Standard zurück

Mit dieser Abfolge wird ein wie in Abschnitt 4.3.2 beschriebener Bildverarbeitungsschritt durchgeführt (siehe Abb. 4.6(b)). Statt `render()` kann auch ein anderer Vorgang aufgerufen werden, z. B. das Rendern von Geometrie mit einer Open-Inventor-Funktion (siehe Abb. 4.6(a)).



(a) ORP1 ist aktiv und rendert Geometrie in die Texturen `Tex1` und `Tex2`.

(b) ORP2 ist aktiv und rendert in die Textur `Tex3`. Die vorher gerenderten Texturen `Tex1` und `Tex2` dienen als Eingabe.

Abb. 4.6: Funktionsweise der Klasse `OffscreenRenderPackage`. Das Modul `SoShadingMap` enthält zwei ORP-Instanzen und drei Texturen. Beide ORPs führen nacheinander einen Rendervorgang aus und speichern das jeweilige Ergebnis in den Texturen.

4.3.5 Berechnung der Schattierungskarte

Mit Hilfe der genannten Techniken und Verfahren der Shader-Programmierung kann die Berechnung der Schattierungskarte erfolgen. Die gesamte Pipeline ist schematisch in Abb. 4.7 dargestellt und wird nun erläutert.

Im ersten Schritt wird die gesamte Szenengeometrie in neun Buffer gerendert, die in drei Texturen gebündelt sind. Der zugehörige Vertexshader berechnet aus der Lichtrichtung und den Normalen aller Glättungsstufen die Buffer der Beleuchtung. Krümmung, Szeneninformationen, ID- und Stencil-Buffer sind einfache Projektionen der jeweiligen Werte. Für korrektes Backface Culling muss ein separater z -Buffer vorhanden sein, der automatisch berechnet wird und ebenfalls als Textur weiterverwendbar ist.

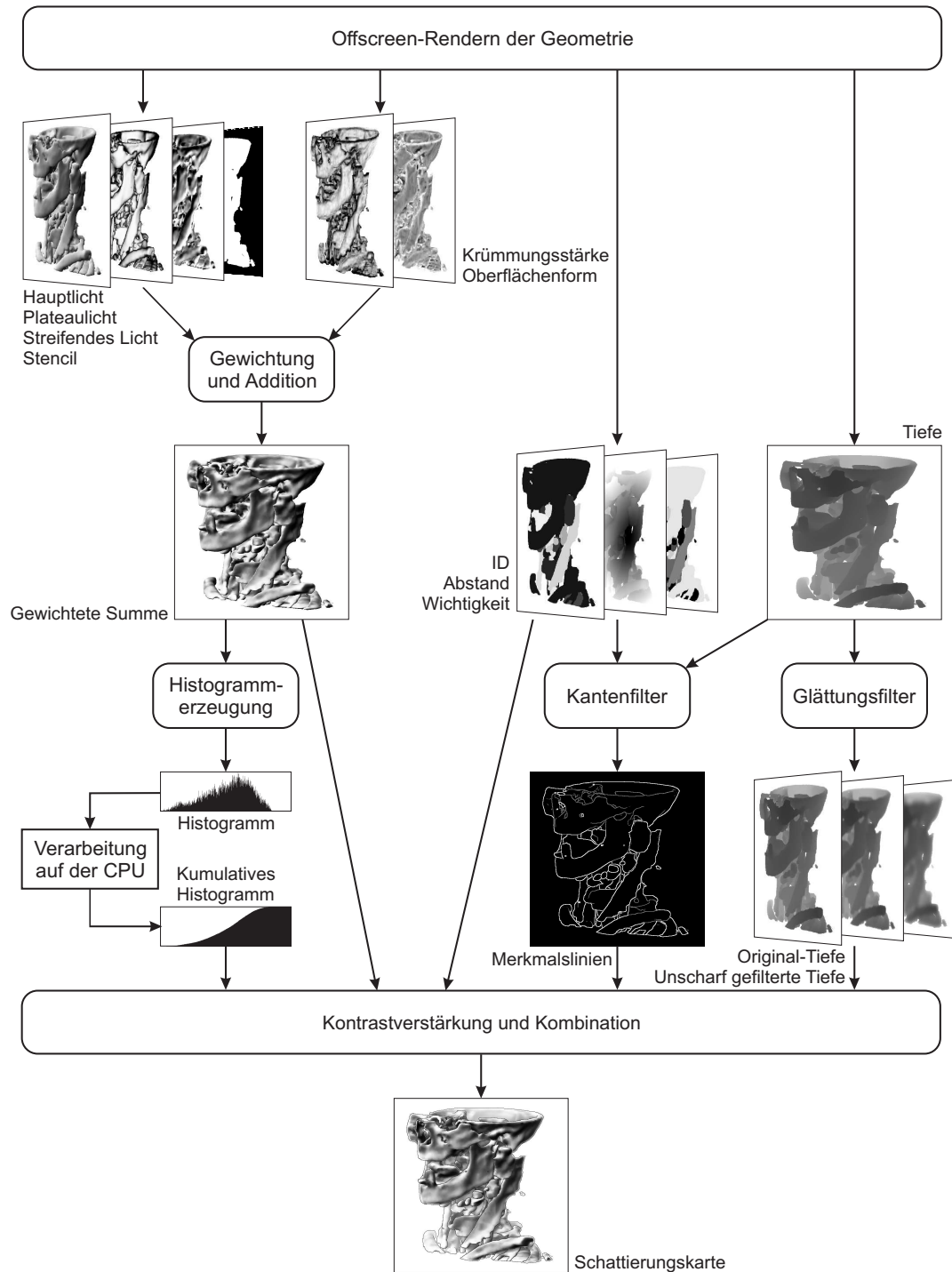


Abb. 4.7: Gesamte Pipeline zur Erstellung der Schattierungskarte mittels Grafik-Hardware. Die Glättungsfilterung wird zur Vereinfachung als ein Rendervorgang dargestellt. Ebenso sind die Kanäle der Summen- und der Merkmalslinien-Textur nicht einzeln dargestellt.

Beleuchtungs- und Krümmungs-Buffer werden im nächsten Schritt gewichtet und addiert. Das Ergebnis ist die Summentextur, deren Kanäle wie in Abschnitt 4.3.3 beschrieben belegt sind. Sie wird anschließend der Histogrammberechnung unterzogen. Auf der CPU entsteht das kumulative Histogramm, das wiederum als Textur auf die Grafikkarte transferiert wird. Parallel dazu werden Kanten- und Glättungsfilter auf ID- und z -Buffer angewandt.

Im letzten Schritt wird zunächst die Histogrammäqualisation der Summentextur durchgeführt. Der jeweilige Grauwert der Summentextur dient als Index für die Textur des kumulativen Histogramms, aus der der neue Grauwert ausgelesen wird. Dieser wird im Anschluss allen Modifikationen mit den Buffern der Szeneninformationen und der Bildoptimierung unterworfen, die in Abschnitt 3.4.1 beschrieben sind. Hier wird auch die Differenz aus originalem und gefiltertem z -Buffer gebildet sowie die atmosphärische Perspektive aus dem z -Buffer berechnet. Die resultierende Ausgabertextur enthält die Schattierungskarte und kann an den illustrativen Renderer übergeben werden.

4.4 Umsetzung des illustrativen Renderns

Die Darstellung der Szenen erfolgt mit den illustrativen Rendermethoden Stippling und Hatching. Es existieren zahlreiche Konzepte zur Umsetzung beider Stile, die in dieser Arbeit aus Zeitgründen jedoch nicht weiter analysiert wurden. Stattdessen wurde auf vorhandene MEVISLAB-Module zurückgegriffen, die aus den Arbeiten von GASTEIGER [2007] und BAER [2005] hervorgingen.

Wegen der implizierten Richtung der Schraffurlinien eignet sich Hatching vor allem für bestimmte anatomische Strukturen wie Muskeln und Blutgefäße, die traditionell entlang der Zugrichtung bzw. radial schraffiert werden. Zur Generierung entsprechender Texturkoordinaten wurde der Ansatz von GASTEIGER [2007] verwendet, der eine automatische Texturierung dieser Strukturen ermöglicht. Die Darstellung des Hatchings erfolgt durch TAM-Texturen nach dem Verfahren von PRAUN et al. [2001]. Dafür wurde ein in MEVISLAB vorhandenes Netzwerk leicht angepasst und zu dem Makromodul `SoHatching` zusammengefasst. Zur Zeit arbeitet es jedoch nur mit einer Texturgröße, daher erfolgt keine Skalierung der Textur.

Alle anderen Strukturen wurden mittels Stippling dargestellt, da die ungerichteten Punkte keine Ausrichtung implizieren. Die Darstellung geschieht ebenfalls mit TAM-Texturen nach demselben Prinzip. Von BAER [2005] wurde ein Verfahren für Stippling entwickelt, das die Szene in unterschiedlich große Einheitswürfel zerlegt und diese als Grundlage für Cube Mapping der Textur verwendet. Dies erlaubt

eine frame-kohärente Darstellung der Stippling-Punkte bei unterschiedlicher Betrachtungsrichtung. Da die Texturierung anhand der Würfel erfolgt, werden keine Textur-Koordinaten für die Objekte benötigt, wodurch beliebige Geometrie ohne weitere Vorverarbeitung verwendbar ist. Das Verfahren ist in dem Modul **SoStippling** implementiert, für die Anwendung mit dem METK existiert das Makromodul **METKStippling**.

Die Module zum illustrativen Rendern basieren teils auf Shader-Programmierung, daher können die jeweiligen Shader mittels Textursampler auf die Schattierungskarte zugreifen. Die MEVISLAB-Bibliothek **SoShader** stellt eine Reihe von Modulen zur Verfügung, mit denen Vertex- und Fragmentshader, Uniform-Variablen und Sampler in Open-Inventor-Szenengraphen integriert werden können. Ein Sampler-Modul bindet beim Rendern des Szenengraphen eine Textur an eine Textureinheit und teilt dem aktiven Shader deren Nummer mit, wodurch dieser Zugriff erhält. Das Modul **SoShadingMap**, das die Schattierungskarte erzeugt, ist daher von der abstrakten Oberklasse **SoSampler** abgeleitet. Im Interface des Moduls kann der Benutzer die Einstellung sämtlicher Variablen der Parametrisierung vornehmen sowie die Textureinheit angeben, um Überschneidungen mit anderen Samplern zu vermeiden. Das komplette Netzwerk vom Laden der Geometrie bis zur illustrativ schattierten Darstellung mittels Stippling ist in Abb. 4.8 abgebildet. Zur Verwendung im METK wurde das Makromodul **METKNPRShading** erstellt, das die Vorverarbeitung und Berechnung der Schattierungskarte für ganze Szenen übernimmt. Es ermöglicht die Darstellung mittels Stippling, Hatching und einfacher Schattierung und erlaubt die Auswahl der Darstellungsart für jedes einzelne Objekt.

4.5 Zusammenfassung

Die Implementierung des entwickelten Konzeptes erfolgte mit der Entwicklungsplattform MEVISLAB. Es wurde die Open-Inventor-basierte Datenstruktur AITSS entworfen und implementiert, mit der `UserNodeValues` an die Renderpipeline übergeben werden können. Für die Umsetzung in MEVISLAB wurden drei Module entwickelt:

WEMInfoGenerator: Dieses Modul berechnet alle benötigten Werte wie Krümmung und Normalenglättung auf einem WEM und speichert diese als `UserNodeValues`.

SoWEM2AttribITSS: Hiermit wird ein WEM in die AITSS-Struktur konvertiert, wobei auch die `UserNodeValues` übernommen werden. Gegebenenfalls aktualisiert das Modul Wichtigkeit oder Objektabstand, die im **WEMInfoGenerator** geändert werden.

SoShadingMap: In diesem Modul wird aus der AITSS-Geometrie die Schattierungskarte der gesamten Szene berechnet und an ein Modul übergeben, welches das illustrative Rendern auf Grundlage der Karte übernimmt.

Die einzelnen Module wurden zur Visualisierung kompletter medizinischer Szenen zu Makromodulen kombiniert und in das METK-Framework eingebunden. Die Berechnung der Schattierungskarte wird auf der Grafik-Hardware durchgeführt. Dafür wurde die Klasse `OffscreenRenderPackage` entwickelt, die das einfache Offscreen-Rendern und die Bildverarbeitung mittels Shadern ermöglicht. Mehrere Instanzen dieser Klasse können zu einer Pipeline verbunden werden, die komplexe Manipulation und Komposition von Texturen durchführt.

Durch den Einsatz von Grafik-Hardware konnten ausreichend hohe Frame-Raten für eine interaktive Visualisierung erreicht werden. In Tabelle 4.1 sind für alle in der Arbeit gezeigten Modelle die Frame-Raten angegeben, die auf dem Testsystem erreicht wurden. Bei der Messung wurden die Modelle nur schattiert, die Darstellung mit Stippling und Hatching erzeugt eine Verringerung der Frame-Raten von etwa 5%. Die Parametrisierung hat keinen Einfluss auf die benötigte Rechenzeit.








Viewportgröße							
256 ²	29	31	19	15	13	10	10
512 ²	20	15	14	12	10	7.5	7.5
1024 ²	6	6	5	5	4.5	4	4
Dreiecke	25k	39k	94k	138k	147k	225k	236k

Tabelle 4.1: Frames pro Sekunde für unterschiedliche Szenen und Viewportgrößen. Die letzte Zeile zeigt die Anzahl der Dreiecke pro Szene (Angabe in Tausend).

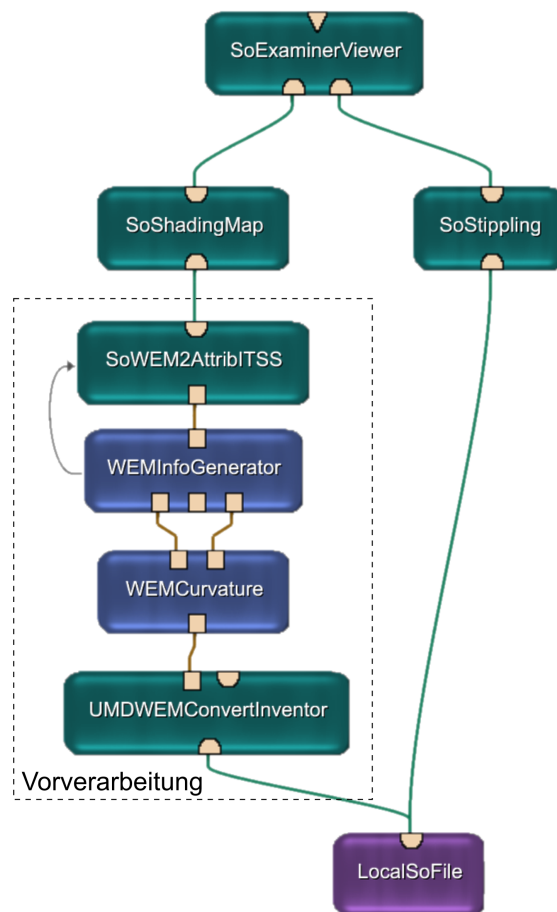


Abb. 4.8: Stippling mit illustrativer Schattierung in MEVISLAB: Die Objektgeometrie durchläuft im linken Zweig die in Abb. 4.1 dargestellte Vorverarbeitung. Aus dem resultierenden AITSS wird in `SoShadingMap` die Schattierungskarte erzeugt und dem Fragmentshader von `SoStippling` übergeben. Die visualisierte Geometrie kann mit dem Modul `SoExaminerViewer` betrachtet werden. Für die Darstellung mit Hatching oder sonstigen Rendermethoden wird `SoStippling` gegen das entsprechende Modul ausgetauscht.

5 Zusammenfassung

Ziel dieser Arbeit war die Entwicklung und Implementierung eines Systems, das die illustrative Schattierung medizinischer 3D-Modelle ermöglicht. Die Schattierung sollte interaktiv parametrisierbar sein, um sie auf unterschiedliche Anwendungsfälle und Modelle anzupassen. Ferner sollten Ansätze für die praktische Handhabung der Parametrisierung aufgezeigt werden.

Durch Analyse medizinischer Illustrationen und Visualisierungen wurden diverse Schattierungs-Parameter ermittelt und in eine Renderpipeline integriert. Die Darstellung der Modelle erfolgt mittels Stippling und Hatching, es können aber auch andere Methoden angewandt werden, bei denen Schattierung eine Rolle spielt. Die Berechnung der Schattierung geschieht durch Kombination aller Parameter zu einer Schattierungskarte, die als Referenz für die illustrative Darstellung dient.

Die Pipeline nutzt das Potential programmierbarer Grafik-Hardware, indem ein Großteil der Berechnungen auf der Grafikkarte ausgeführt wird. Durch die hohe Rechenleistung der GPU und geringen Datentransfer zur CPU ergeben sich ausreichende hohe Frame-Raten, um eine interaktive Betrachtung der schattierten Modelle zu ermöglichen. Die Parameter werden auf der Grafikkarte mittels Framebuffer Objects in Texturen gerendert, verarbeitet und als Schattierungskarte an den Stippling-/Hatching-Renderer übermittelt. Die Bildfilterung der Parameter-Buffer sowie die Histogrammerzeugung erfolgen mit Shader-Programmen. Lediglich die Auswertung des Histogramms geschieht auf der CPU, was die Echtzeitfähigkeit des Systems jedoch nicht beeinträchtigt.

Da zahlreiche Variablen Einfluss auf die Schattierung haben, wurden Konzepte zur Vereinfachung der Parametrisierung entwickelt. Diese zielen darauf ab, die Parametrisierung einerseits durch Reduzierung der Variablen zu erleichtern, andererseits durch die Vorgabe von Templates ganz zu vermeiden. Es wurde anhand von Beispielen gezeigt, dass für verschiedene Strukturtypen sinnvolle Voreinstellungen existieren, die bei Bedarf mit sechs Variablen in kurzer Zeit zielgerichtet angepasst werden können. Da Veränderungen in Echtzeit am Modell sichtbar sind, kann die Parametrisierung explorativ betrieben werden.

Zusammenfassend wurde gezeigt, dass durch die ermittelten Parameter eine sowohl zweckgebundene als auch ästhetische Schattierung medizinischer 3D-Modelle

möglich ist, die über einfache Beleuchtungsmodelle hinausgeht. Die Schattierung kann effizient und zielgerichtet vom Benutzer parametrisiert werden, um verschiedene Modelle, Strukturtypen und Anwendungsziele abzudecken. Die Betrachtung der Modelle und die Parametrisierung der Schattierung sind in Echtzeit möglich. Damit wurde das Ziel der Arbeit in allen Aspekten erfüllt.

5.1 Ausblick

Es wird nun betrachtet, wie das erreichte System zu verbessern und zu erweitern ist. Zunächst wäre die Implementierung der Teilaspekte zu nennen, die im Konzept erarbeitet, aber aus Zeitmangel nicht umgesetzt wurden. Dies wäre vor allem die Parametrisierung einzelner Objektgruppen, die relativ einfach zu realisieren ist. Mehr Aufwand verlangen der Einbezug von transparenten Objekten und die automatische Parametrisierung. Beide Bereiche müssen genauer betrachtet und die aufgezeigten Problemfelder adressiert werden, um sie konkret umzusetzen.

In der vorliegenden Implementierung ist fest vorgegeben, wie die Parameter zur Schattierungskarte kombiniert werden. Es wäre auch denkbar, dies dem Benutzer zu überlassen. In ihrer Hatching-Pipeline geben ZANDER et al. [2004] dem Anwender die Möglichkeit, durch Eingabe von Gleichungen die Gestalt der Hatching-Linien zu bestimmen; Änderungen werden sofort in die Pipeline integriert. Eine ähnliche Parametrisierung kann auch hier implementiert werden, indem der Benutzer die gewünschte Parameterkombination als mathematischen Ausdruck oder mittels einer Skriptsprache angibt. Um den Vorgang anschaulicher zu gestalten, kann auch der Ansatz einer grafischen Oberfläche zum Entwurf von Renderpipelines Anwendung finden, der von HALPER et al. [2002] vorgestellt wird. Dem Benutzer werden Abbildungen der einzelnen Parameter-Buffer präsentiert, die er durch Ziehen von Verbindungen zu einem Netzwerk kombinieren kann. Die Verbindungen entsprechen Operationen, mit denen die Buffer verändert oder verknüpft werden. Die Resultate der Verarbeitungsschritte werden ebenfalls grafisch dargestellt, wodurch die gewünschte Kombination sehr intuitiv erstellt werden kann.

Eine informelle Nutzerstudie ergab Ansätze für eine bessere Bedienbarkeit der Anwendung. Es wurde vorgeschlagen, neben der gewählten illustrativen Darstellung auch eine einfache schattierte Ansicht des Modells sowie die einzelnen Parameter-Buffer zu zeigen. Dies könnte dabei helfen, den Einfluss der Parameter weiter zu verdeutlichen und auch bei manueller Einstellung sämtlicher Variablen eine gezielte Bearbeitung ermöglichen. Zusätzliches räumliches Verständnis würde durch Schattentwurf der gesamten Szene auf eine Hintergrundebene erreicht. Dafür kann ein korrekter Schatten wie in der Arbeit von RITTER et al. [2001] verwendet oder der Überdeckungsschatten auf den Hintergrund erweitert werden.

Weiter ergaben sich Vorschläge für die Parametrisierung, die über die einfache Angabe von Werten hinausgehen. So könnte beispielsweise der Verlauf der atmosphärischen Perspektive in einem Graphen abgebildet werden. Dieser bietet zum einen eine alternative Visualisierung des Parameters, zum anderen könnten minimale und maximale Tiefe sowie Anstiegsexponent durch Manipulation des Graphen eingestellt werden. Denkbar wäre auch die beliebige Veränderung des Graphen durch Kontrollpunkte. Die Repräsentation als Graph bietet sich auch für Detailfrequenz und -amplitude an und könnte so die Parametrisierung durch weitere visuelle Hinweise erleichtern. Parameter wie das Kantenlicht, die eine deutliche Ausdehnung haben, könnten direkt in der illustrativen 3D-Darstellung verändert werden, indem die Entfernung zwischen dem Rand des Kantenlichtes und der angrenzenden Merkmalslinie festgelegt wird. Überdeckungsschatten und Objektabstand wären ebenfalls auf diese Weise einstellbar.

Zur Erweiterung der Parameterbasis sind die in Abschnitt 2.1.5 genannten Gestaltungsprinzipien wie Komposition und Balance der Illustration zu untersuchen. In ihrer grundlegenden Arbeit über computergenerierte Zeichnungen wenden WINKENBACH und SALESIN [1994] das Prinzip der Andeutung an: Es werden nur ausgewählte Stellen einer Illustration mit Textur versehen, um sie ästhetischer wirken zu lassen. Wenn die bei ihnen noch manuelle Auswahl der Bereiche automatisiert werden könnte, wäre dieser Ansatz als weiterer Parameter zu integrieren. Da solche gestalterischen Parameter wie erwähnt stark von der Betrachtungsrichtung abhängen, könnte hier die interaktive Betrachtung auf günstige Positionen gelenkt werden. Eine Untersuchung der optimalen Kamerapositionierung in medizinischen Szenen wurde von MÜHLER et al. [2007] durchgeführt. Mit diesem Ansatz wäre durch Optimierung der Blickrichtung auch die Schattierung weiter verbesserbar.

Für ein hohes Maß an Kontrolle über die Darstellung könnte auch zu einer stärker manuell ausgerichteten Parametrisierung wie bei WINKENBACH und SALESIN [1994] übergegangen werden. So wäre es beispielsweise möglich, bestimmte Bereiche eines Objektes mehr oder weniger zu detaillieren, was die gleichzeitige Glättung von Artefakten und Hervorhebung erwünschter Details an verschiedenen Stellen ermöglicht. Die Realisierung der Auswahl von Bereichen in einer dreidimensionalen Umgebung stellt eine Herausforderung dar. Das Konzept des 3D-Pinsels, das in gängiger Modellierungssoftware angewandt wird, käme dafür in Betracht.

Zur weiteren Verbesserung des Verständnisses sind Illustrationen oft mit Kommentaren versehen. Vor allem bei internen Beschriftungen der Objekte wie in Abb. 2.9 (S. 10) ist es wichtig, dass diese vor dem Hintergrund des schattierten Objektes gut zu erkennen ist. Von GÖTZELMANN et al. [2006] stammt ein System, mit dem Kommentare in interaktive 3D-Szenen integriert werden können. In einer Kombination beider Arbeiten könnte die optimale Erkennbarkeit solcher Kommentare als weiteres Kriterium für die Schattierung herangezogen werden.

Der bisherige Ansatz der Schattierung ist auf Oberflächenmodelle beschränkt. Direktes Volumen-Rendering wird in der medizinischen Visualisierung auch häufig verwendet, wobei ebenfalls mit illustrativen Techniken gearbeitet werden kann. Unter anderem wurden Konzepte für Stippling [LU et al., 2002] und Hatching [GERL, 2006] realisiert. Daher wäre es interessant, den Ansatz dieser Arbeit auf Volumen-Rendering zu übertragen. Da hierbei Transparenz eine wichtige Rolle spielt und keine eindeutigen Oberflächen existieren, wäre möglicherweise eine Alternative zu der Verarbeitung im Bildraum zu entwickeln.

Die vorgestellten Resultate und vielfältigen Ansätze zur Erweiterung zeigen, dass die interaktive Schattierung ein großes Potenzial bietet, die Methoden der medizinischen Visualisierung zu ergänzen.

Literaturverzeichnis

- [AKERS et al. 2003] AKERS, David ; LOSASSO, Frank ; KLINGNER, Jeff ; AGRAWALA, Maneesh ; RICK, John ; HANRAHAN, Pat: Conveying Shape and Features with Image-Based Relighting. In: *IEEE Visualization*, IEEE Computer Society, 2003, S. 349–354
- [ANDERSON und LEVOY 2002] ANDERSON, Sean E. ; LEVOY, Marc: Unwrapping and Visualizing Cuneiform Tablets. In: *IEEE Computer Graphics and Applications* 22 (2002), Nr. 6, S. 82–88
- [ANDREWS 2006] ANDREWS, Bill: Introduction to Perceptual Principles in Medical Illustrations. In: *Illustrative Visualization for Medicine and Science*, ACM Press, 2006, S. 8–30
- [BAER 2005] BAER, Alexandra: *Hardwaregestütztes Stippling von medizinischen Oberflächenmodellen*, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Diplomarbeit, 2005
- [BAUMGART 1972] BAUMGART, Bruce G.: Winged Edge Polyhedron Representation. Stanford University, 1972. – Forschungsbericht
- [BRUCKNER und GRÖLLER 2007] BRUCKNER, Stefan ; GRÖLLER, Meister E.: Style Transfer Functions for Illustrative Volume Rendering. In: *Computer Graphics Forum* 26 (2007), Nr. 3, S. 715–724
- [CHESELDEN 1733] CHESELDEN, William: *Osteographia, or The Anatomy of the Bones*. London : William Bowyer, 1733
- [CIGNONI et al. 2005] CIGNONI, Paolo ; SCOPIGNO, Roberto ; TARINI, Marco: A Simple Normal Enhancement Technique for Interactive Non-Photorealistic Renderings. In: *Computers & Graphics* 29 (2005), Nr. 1, S. 125–133
- [COHEN et al. 2004] COHEN, Jonathan D. ; DUNCAN, Donald ; SNYDER, Dean ; COOPER, Jerrold ; KUMAR, Subodh ; HAHN, Daniel ; CHEN, Yuan ; PURNOMO, Budirijanto ; GRAETTINGER, John: iClay: Digitizing Cuneiform. In: *The 5th International Symposium on Virtual Reality, Archaeology and Cultural Heritage*, Eurographics Association, 2004, S. 135–143

- [EBERT und RHEINGANS 2000] EBERT, David S. ; RHEINGANS, Penny: Volume Illustration: Non-Photorealistic Rendering of Volume Models. In: *IEEE Visualization*, IEEE Computer Society, 2000, S. 195–202
- [EVERITT 2001] EVERITT, Cass: *Interactive Order-Independent Transparency*. 2001. – URL http://developer.nvidia.com/object/Interactive_Order_Transparency.html. – Link gefunden am 17.12.2007
- [FLUCK et al. 2006] FLUCK, Oliver ; AHARON, Shmuel ; CREMERS, Daniel ; ROUSSON, Mikael: GPU Histogram Computation. In: *SIGGRAPH '06: ACM SIGGRAPH 2006 Research posters*, ACM Press, 2006, S. 53
- [FOLEY et al. 1995] FOLEY, James D. ; DAM, Andries van ; FEINER, Steven K. ; HUGHES, John F.: *Computer Graphics: Principles and Practice*. 2. Reading, Massachusetts : Addison-Wesley, 1995
- [GASTEIGER 2007] GASTEIGER, Rocco: *Krümmungs- und modellbasierte Schraffierung auf patientenspezifischen, anatomischen Oberflächen*, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Diplomarbeit, 2007
- [GERL 2006] GERL, Moritz: *Volume Hatching for Illustrative Visualization*, Technische Universität Wien, Fakultät für Informatik, Diplomarbeit, 2006
- [GOOCH et al. 1998] GOOCH, Amy ; GOOCH, Bruce ; SHIRLEY, Peter ; COHEN, Elaine: A Non-Photorealistic Lighting Model for Automatic Technical Illustration. In: *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM Press, 1998, S. 447–452
- [GOOCH et al. 1999] GOOCH, Bruce ; SLOAN, Peter-Pike J. ; GOOCH, Amy ; SHIRLEY, Peter ; RIESENFELD, Richard F.: Interactive Technical Illustration. In: *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, ACM Press, 1999, S. 31–38
- [GÖTZELMANN et al. 2006] GÖTZELMANN, Timo ; HARTMANN, Knut ; STROTHOTTE, Thomas: Agent-Based Annotation of Interactive 3D Visualizations. In: *Smart Graphics*, Springer Verlag, 2006, S. 24–35
- [GRAY 1918] GRAY, Henry: *Anatomy of the Human Body*. Philadelphia : Lea & Febiger, 1918
- [GREEN 2005] GREEN, Simon: *Image Processing Tricks in OpenGL*. Game Developers Conference 2005: OpenGL Tutorial Day. 2005. – URL http://developer.nvidia.com/object/gdc_2005_presentations.html. – Link gefunden am 17.12.2007
- [GUMHOLD 2002] GUMHOLD, Stefan: Maximum Entropy Light Source Placement. In: *IEEE Visualization*, IEEE Computer Society, 2002, S. 275–282

- [HADWIGER et al. 2005] HADWIGER, Markus ; SIGG, Christian ; SCHARSACH, Henning ; BÜHLER, Katja ; GROSS, Markus: Real-Time Ray-Casting and Advanced Shading of Discrete Isosurfaces. In: *Computer Graphics Forum* 24 (2005), Nr. 3, S. 303–312
- [HALLE und MENG 2003] HALLE, Michael ; MENG, Jeanette: LightKit: A Lighting System for Effective Visualization. In: *IEEE Visualization*, IEEE Computer Society, 2003, S. 363–370
- [HALPER et al. 2002] HALPER, Nick ; SCHLECHTWEG, Stefan ; STROTHOTTE, Thomas: Creating Non-Photorealistic Images the Designer’s Way. In: *NPAR ’02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, ACM Press, 2002, S. 97–104, 162
- [HAMEL 2000] HAMEL, Jörg: *A New Lighting Model for Computer Generated Line Drawings*, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Dissertation, 2000
- [HANSEN 2006] HANSEN, Christian: *Verwendung von Textur in der Gefäßvisualisierung*, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Diplomarbeit, 2006
- [HASENFRATZ et al. 2003] HASENFRATZ, Jean-Marc ; LAPIERRE, Marc ; HOLZSCHUCH, Nicolas ; SILLION, François: A Survey of Real-Time Soft Shadows Algorithms. In: *Computer Graphics Forum* 22 (2003), Nr. 4, S. 753–774
- [HLADUVKA et al. 2000] HLADUVKA, Jiří ; KÖNIG, Andreas ; GRÖLLER, Eduard: Curvature-Based Transfer Functions for Direct Volume Rendering. In: *Spring Conference on Computer Graphics 2000* Bd. 16, Comenius University Press, 2000, S. 58–65
- [HODGES 1989] HODGES, Elaine R. S. (Hrsg.): *The Guild Handbook of Scientific Illustration*. New York : Van Nostrand Reinhold, 1989
- [ISENBERG et al. 2003] ISENBERG, Tobias ; FREUDENBERG, Bert ; HALPER, Nick ; SCHLECHTWEG, Stefan ; STROTHOTTE, Thomas: A Developer’s Guide to Silhouette Algorithms for Polygonal Models. In: *IEEE Computer Graphics and Applications* 23 (2003), Nr. 4, S. 28–37
- [ISENBERG et al. 2006] ISENBERG, Tobias ; NEUMANN, Petra ; CARPENDALE, Sheelagh ; SOUSA, Mario C. ; JORGE, Joaquim A.: Non-Photorealistic Rendering in Context: An Observational Study. In: *NPAR ’06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, ACM Press, 2006, S. 115–126

- [JULIANO und SANDMEL 2007] JULIANO, Jeff ; SANDMEL, Jeremy: *OpenGL Framebuffer Object Extension*. 2007. – URL http://www.opengl.org/registry/specs/EXT/framebuffer_object.txt. – Link gefunden am 17.12.2007
- [KINDLMANN et al. 2003] KINDLMANN, Gordon L. ; WHITAKER, Ross T. ; TASDIZEN, Tolga ; MÖLLER, Torsten: Curvature-Based Transfer Functions for Direct Volume Rendering: Methods and Applications. In: *IEEE Visualization*, IEEE Computer Society, 2003, S. 513–520
- [KOPSCH und RAUBER 1950] KOPSCH, Friedrich ; RAUBER, August: *Lehrbuch und Atlas der Anatomie des Menschen*. Bd. 3. 17. Leipzig : Georg Thieme Verlag, 1950
- [KUMAR et al. 2003] KUMAR, Subodh ; SNYDER, Dean ; DUNCAN, Donald ; COHEN, Jonathan ; COOPER, Jerry: Digital Preservation of Ancient Cuneiform Tablets Using 3D-Scanning. In: *Fourth International Conference on 3-D Digital Imaging and Modeling (2003)*, S. 326–333
- [LEE und HAO 2006] LEE, Chang H. ; HAO, Xuejun: Geometry-Dependent Lighting. In: *IEEE Transactions on Visualization and Computer Graphics* 12 (2006), Nr. 2, S. 197–207
- [LORENSEN und CLINE 1987] LORENSEN, William E. ; CLINE, Harvey E.: Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In: *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ACM Press, 1987, S. 163–169
- [LU et al. 2002] LU, Aidong ; MORRIS, Christopher J. ; EBERT, David S. ; RHEINGANS, Penny ; HANSEN, Charles: Non-Photorealistic Volume Rendering Using Stippling Techniques. In: *IEEE Visualization*, IEEE Computer Society, 2002, S. 211–218
- [LUFT et al. 2006] LUFT, Thomas ; COLDITZ, Carsten ; DEUSSEN, Oliver: Image Enhancement by Unsharp Masking the Depth Buffer. In: *ACM Transactions on Graphics* 25 (2006), Nr. 3, S. 1206–1213
- [MEVIS RESEARCH 2007] MEVIS RESEARCH: *MeVisLab Home Page*. 2007. – URL <http://www.mevislab.de>. – Link gefunden am 17.12.2007
- [MILLER 1994] MILLER, Gavin: Efficient Algorithms for Local and Global Accessibility Shading. In: *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM Press, 1994, S. 319–326
- [MITCHELL et al. 2006] MITCHELL, Jason ; MCTAGGART, Gary ; GREEN, Chris: Shading in Valve's Source Engine. In: *SIGGRAPH '06: ACM SIGGRAPH 2006 Courses*, ACM Press, 2006, S. 129–142

- [MÜHLER et al. 2007] MÜHLER, Konrad ; NEUGEBAUER, Mathias ; TIETJEN, Christian ; PREIM, Bernhard: Viewpoint Selection for Intervention Planning. In: *IEEE/Eurographics Symposium on Visualization*, Eurographics Association, 2007, S. 267–274
- [PRAUN et al. 2001] PRAUN, Emil ; HOPPE, Hugues ; WEBB, Matthew ; FINKELSTEIN, Adam: Real-time Hatching. In: *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 2001, S. 581–586
- [PREIM et al. 2005] PREIM, Bernhard ; TIETJEN, Christian ; DÖRGE, Christina: NPR, Focussing and Emphasis in Medical Visualizations. In: *Simulation und Visualisierung 2005*, SCS European Publishing House, 2005, S. 139–152
- [RITTER et al. 2006] RITTER, Felix ; HANSEN, Christian ; DICKEN, Volker ; KONRAD, Olaf ; PREIM, Bernhard ; PEITGEN, Heinz-Otto: Real-Time Illustration of Vascular Structures. In: *IEEE Transactions on Visualization and Computer Graphics* 12 (2006), Nr. 5, S. 877–884
- [RITTER et al. 2001] RITTER, Felix ; STROTHOTTE, Thomas ; DRESDEN, Oliver ; PREIM, Bernhard: Virtual 3D Puzzles: A New Method for Exploring Geometric Models in VR. In: *IEEE Computer Graphics and Applications* 21 (2001), Nr. 5, S. 11–13
- [RUSINKIEWICZ et al. 2006] RUSINKIEWICZ, Szymon ; BURNS, Michael ; DE-CARLO, Douglas: Exaggerated Shading for Depicting Shape and Detail. In: *ACM Transactions on Graphics* 25 (2006), Nr. 3, S. 1199–1205
- [SAITO und TAKAHASHI 1990] SAITO, Takafumi ; TAKAHASHI, Tokiichiro: Comprehensible Rendering of 3-D Shapes. In: *ACM SIGGRAPH Computer Graphics* 24 (1990), Nr. 4, S. 197–206
- [SALAH et al. 2006] SALAH, Zein ; BARTZ, Dirk ; STRASSER, Wolfgang ; TAGIBA, Marcos: Expressive Anatomical Illustrations Based on Scanned Patient Data. In: *GMS CURAC* 1 (2006)
- [SCHEUERMANN und HENSLEY 2007] SCHEUERMANN, Thorsten ; HENSLEY, Justin: Efficient Histogram Generation Using Scattering on GPUs. In: *Proceedings of the 2007 Symposium on Interactive 3D Graphics*, ACM Press, 2007, S. 33–37
- [SCHULZ 2005] SCHULZ, Christian: *Approximation von Krümmungsinformationen zur Umsetzung von Techniken zur illustrativen medizinischen Visualisierung*, Otto-von-Guericke-Universität Magdeburg, Fakultät für Informatik, Diplomarbeit, 2005

- [SHACKED und LISCHINSKI 2001] SHACKED, Ram ; LISCHINSKI, Dani: Automatic Lighting Design Using a Perceptual Quality Metric. In: *Computer Graphics Forum* 20 (2001), Nr. 3, S. 215–227
- [STEINBRECHER 1993] STEINBRECHER, Rainer: *Bildverarbeitung in der Praxis*. München/Wien : Oldenbourg, 1993
- [STONE und STONE 2000] STONE, Robert J. ; STONE, Judith A.: *Atlas of Skeletal Muscles*. 3. New York : McGraw-Hill Higher Education, 2000
- [TIETJEN et al. 2005] TIETJEN, Christian ; ISENBERG, Tobias ; PREIM, Bernhard: Combining Silhouettes, Shading, and Volume Rendering for Surgery Education and Planning. In: *Proceedings of the Eurographics*, Eurographics Association, 2005 (Eurographics Workshop Series), S. 303–310, 335
- [TIETJEN et al. 2008] TIETJEN, Christian ; MÜHLER, Konrad ; RITTER, Felix ; KONRAD, Olaf ; HINDENNACH, Milo ; PREIM, Bernhard: METK – The Medical Exploration Toolkit. In: *Bildverarbeitung für die Medizin*, Springer Verlag, 2008, S. noch nicht bekannt
- [TITTEL 1994] TITTEL, Kurt: *Beschreibende und funktionelle Anatomie des Menschen*. 12. Jena/Stuttgart : Gustav Fischer Verlag, 1994
- [VESALIUS 1543] VESALIUS, Andreas: *De Humani Corporis Fabrica*. Basel : National Library of Medicine, 1543
- [VIOLA et al. 2004] VIOLA, Ivan ; KANITSAR, Armin ; GRÖLLER, Meister E.: Importance-Driven Volume Rendering. In: *IEEE Visualization*, IEEE Computer Society, 2004, S. 139–145
- [WINKENBACH und SALESIN 1994] WINKENBACH, Georges ; SALESIN, David: Computer-Generated Pen-and-Ink Illustration. In: *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM Press, 1994, S. 91–100
- [XIE et al. 2007] XIE, Xuexiang ; HE, Ying ; TIAN, Feng ; SEAH, Hock-Soon ; GU, Xianfeng ; QIN, Hong: An Effective Illustrative Visualization Framework Based on Photic Extremum Lines (PELs). In: *IEEE Transactions on Visualization and Computer Graphics* 13 (2007), Nr. 6, S. 1328–1335
- [YOUNG 2007] YOUNG, Christine (Hrsg.): *Medical Illustration Source Book*. 20. Santa Barbara : Serbin Communications, Inc., 2007
- [ZANDER et al. 2004] ZANDER, Johannes ; ISENBERG, Tobias ; SCHLECHTWEG, Stefan ; STROTHOTTE, Thomas: High Quality Hatching. In: *Computer Graphics Forum* 23 (2004), Nr. 3, S. 421–430

- [ZHOU et al. 2004] ZHOU, Jianlong ; DÖRING, Andreas ; TÖNNIES, Klaus D.: Distance Based Enhancement for Focal Region Based Volume Rendering. In: *Bildverarbeitung für die Medizin*, Springer Verlag, 2004, S. 199–203