Rheinisch Westfählische Technische Hoschschule
Lehrstuhl VIII – Computergrafik

Prosmeninar: Ausgewählte Kapitel der Computergrafik

Lokale Beleuchtungsberechnung

Ausarbeitung von Stefan Moeller

17. Mai 2002

Vorwort

Diese Ausarbeitung ist während des Proseminars „*Ausgewählte Kapitel der Computergrafik*“ zum Thema „***Lokale Beleuchtungsberechnung***“ im Sommersemster 2002 an der RWTH-Aachen (Lehrstuhl 8) entstanden.

Das Dokument darf frei weitergegeben und kopiert werden, dies bezieht sich sowohl auf die Postscript-Datei als auch auf Ausdrucke. Veränderungen an Inhalten sind nur nach ausdrücklicher Genehmigung des Autors erlaubt. Eine kommerzielle Verwertung ist untersagt. Natürlich kann ich für Beschädigungen o.ä., die durch dieses Dokument – auch nur indirekt – hervorgerufen werden, keine Haftung übernehmen.

Ich bitte alle, die dieses Dokument benutzen, freundlichst über evtl. Tippfehler hinwegzusehen, mir aber ggf. eine E-Mail zu schreiben. Auch bei größter Sorgfalt sind inhaltliche Irrtümer und Fehler nicht ausgeschlossen. Diese Ausarbeitung ist unter der Web-Adresse www.sloppi.de/downloads zum Download verfügbar.

Aachen, Mai 2002 *Stefan Moeller*

Letzte Änderung: 5. Juni 2002

Internet: www.smfx.de
E-Mail: sm@smfx.de

Inhaltsverzeichnis

Vorbemerkung

In dieser Ausarbeitung werden verschiedene Beleuchtungsmodelle und Ansätze zu ihrer Implementierung vorgestellt. Beleuchtungsmodelle dienen der Beschreibung der Wechselwirkung zwischen Lichtquelle und Oberfläche einer Szene und liefern Informationen darüber, wie eine Oberfläche unter Berücksichtigung ihrer Beschaffenheit und ihrer Lage bezüglich Betrachter und Lichtquelle einzufärben ist. Beleuchtungsmodelle werden in der Computergrafik in zwei Kategorien eingeteilt: *lokale* und *globale* Beleuchtungsmodelle. Bilderzeugungsverfahren berechnen, welche Oberflächen bzw. Teile von Oberflächen vom Betrachter aus sichtbar sind. In jedem sichtbaren Objektflächenpunkt wird dann ein lokales Beleuchtungsmodell zur Bestimmung der Intensität ausgewertet. Bei lokalen Beleuchtungsmodellen wird für die Berechnung nur die direkte Beleuchtung einer Oberfläche durch Lichtquellen herangezogen. Die Beleuchtung oder Verschattung durch andere Objekte wird nicht berücksichtigt und üblicherweise durch einen konstanten ambienten Term approximiert.

Möchte man den Einfluss anderer Objekte der Szene in die Einfärbung eines Punktes mit einbeziehen, also die Wechselwirkung der Objekte untereinander berücksichtigen, so benötigt man ein globales Beleuchtungsmodell. Erst danach wird die Darstellung optischer Phänomene – wie beispielsweise die Reflexion oder Brechung von Licht – möglich. Zwei Ansätze zur Realisierung eines solchen globalen Modells sind das *Strahlenverfolgungsverfahren* (engl. Raytracing) und das *Strahlenaustauschverfahren* (engl. Radiosity).

In dieser Ausarbeitung gehe ich ausschließlich auf lokale Beleuchtungsberechnungen ein.

1 Beleuchtungsmodelle

1.1 Umgebendes Licht

Allgemein und stark vereinfacht kann man die resultierende Intensität I eines Bildpunktes durch das Produkt aus der Intensität der Lichtquelle (I_a) und dem *globalen Reflexionskoeffizient* ($0 \leq k_a \leq 1$)

$$I = I_a \cdot k_a \tag{1}$$

berechnen.

1.2 Reflexion

Um die Reflexion des Lichtes in diese Intensitätsberechnung einzubeziehen wird die Abhängigkeit vom Winkel θ zwischen dem Richtungsvektor zur Lichtquelle $\overrightarrow{L}$ und dem Normalenvektor der Oberfläche $\overrightarrow{N}$ angehängt. Damit ergibt sich für die Intensität:

$$I = I_a \cdot k_a \cos\theta \tag{2}$$

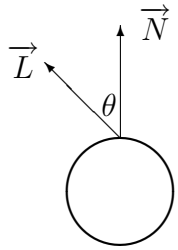

Abbildung 1: einfache Reflexion

Das an der Objektoberfläche reflektierte Licht wird mit größer werdendem Abstand (Objekt $\leftrightarrow$ Lichtquelle) immer schwächer. Um diese *Beleuchtungsabschwächung* zu berücksichtigen, wird der *Abschwächungskoeffizient* f_{att} eingeführt. Dabei ist f_{att} abhängig von der Entfernung der Lichtquelle zur Oberfläche (d_L):

$$f_{att} = \frac{1}{d_L^2} \tag{3}$$

Als besondere Reflexion gibt es die *Lamberti-* oder *Diffuse Reflexion*, wie sie zum Beispiel bei der Lichtreflexion an Kreide auftritt.

Betrachtet man die Reflexion mit farbigen Komponenten (farbiges Licht und/oder farbige Oberflächen) können die Farbkomponenten jeweils einzeln berechnet und schließlich in einem n-Tupel zusammengefasst werden.

Beispiel: Das RGB-Farbmodell wird nach obigen Annahmen in drei Schritten für jede Farbe einzeln berechnet. Die resultierenden Intensitäten werden in einem 3-Tupel zusammengefaßt.

Für die *rote* Komponente ergibt sich:

$$I_{red} = I_{a_{red}} k_a O_{d_{red}} + f_{att} I_{p_{red}} k_p O_{d_{red}} \cdot \left(\overrightarrow{N} \cdot \overrightarrow{L}\right) \tag{4}$$

Analog dazu werden die beiden anderen Komponenten *Blau* und *Grün* berechnet und schließlich in $I_{RGB} = (I_{red}, I_{blue}, I_{green})$ zusammengefaßt.

In der Gleichung (4) stellt der linke Summand die *Licht-* und der rechte Summand die *Farbkomponente* mit $I_{a_{red}}$: Intensität der Lichtquelle, k_a:Globaler Reflexionskoeffizient, $O_{d_{red}}$: Objektfarbe, f_{att}: Abschwächungskoeffizient, $I_{p_{red}}$: primäre Farbintensität (Lichtquelle), k_p: primärer Reflexionskoeffizient (Lichtquelle), $O_{d_{red}}$: Objektfarbe und $\left(\overrightarrow{N} \cdot \overrightarrow{L}\right)$: Winkel zwischen Lichtquellen- und Normalenvektor dar.

1.3 Umgebungs-Abschwächung

Je weiter der Betrachter vom beleuchteten Objekt entfernt ist, desto geringer nimmt er die Objektintensität wahr. Um diese *Tiefenzeichnung* zu simulieren werden die Maßstabsfaktoren s_f und s_b $(0 < s_f, s_b < 1)$ eingeführt, die eine Anhängigkeit zwischen Maßstab, also der Objektentfernung zum Betrachter, und der Beleuchtungsintensität repräsentieren.

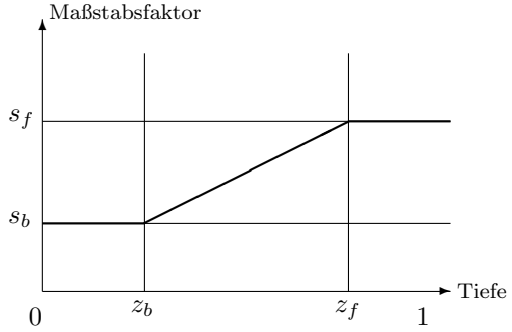

Abbildung 2: Maßstabfaktoren

1.4 Spiegel-Reflexion

Je nach Beschaffenheit der Objektoberfläche sind die Spiegeleigenschaften sehr unterschiedlich (Vergleich: Diffuse Refelxion). Bei glänzenden Oberflächen und starker Beleuchtung verliert die Oberfläche ihre eigene Farbe und nimmt die Farbe der Lichtquelle (ganz oder teilweise) an. Am Beispiel des roten Apfels (Abbildung 3) kann man dieses Phänomen deutlich erkennen.

Um die Spiegel-Reflexion genauer zu beschreiben, erweitern wir das obige Beispiel (Abbildung 1) um den Reflexionsvektor $\overrightarrow{R}$, den Betrachtervektor $\overrightarrow{V}$ und den dazwischen liegenden Winkel α.

Abbildung 3: Beispiel roter Apfel

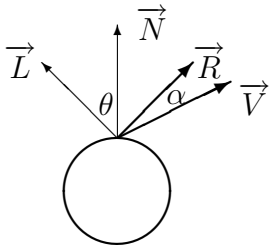

Abbildung 4: Spiegel-Reflexion

Dabei erhält man den Refelxionsvektor durch Spiegelung des Lichtquellenvektors über den Normalenvektor. Die in oben genanntem Apfel-Beispiel beschriebene Intensitätsspitze wird mit diesen Angaben nur dann vom Betrachter wahrgenommen, wenn $\alpha = 0$, also die Vektoren $\overrightarrow{R}$ und $\overrightarrow{V}$ in dieselbe Richtung zeigen. Dieses Modell – *Phong-Modell* – wird für nicht-perfekte Spiegelungen angenommen.

1.5 Verbesserung des Punkt-Lichtquellen-Modell

In den bisherigen Betrachtungen ist man davon ausgegangen, dass die Lichtquellen das Objekt gleichmäßig bestrahlen, was in der Realität aber nicht der Fall ist. Um dies zu simulieren kann man eine *künstliche, gleichmäßige Lichtquelle* erzeugen, indem man eine reale Lichtquelle über einen Spiegel in die Richtung der angenommenen, idealen Lichtquelle reflektiert.

1.6 Mehrere Lichtquellen

Bisher galten alle Berechnungen nur für einzelne Lichtquellen. Diese Berechnungen kann man bei mehreren Lichtquellen analog anwenden. Es werden Berechnungen für jede einzelne Lichtquelle durchgeführt. Diese Einzelergebnisse kann man einfach addieren um eine resultierende Intensität zu erhalten. Für m Lichtquellen gilt:

$$I_{res} = \sum_{1 \leq m < \infty} I_m \tag{5}$$

Wie am Apfel-Beispiel beschrieben, kann ein Bildpunkt maximal mit der Farbe der Lichtquelle gefärbt werden um maximale Reflexion zu simulieren. Dieser Maximalwert kann bei der Summierung überschritten werden. Um diesen Fehler auszugleichen gibt es verschiedene Korrekturmöglichkeiten:

- Die einfachste Korrektur erreicht man durch *Abschneiden des Intensitätswertes*. Dabei wird, wenn der Maximalwert überschritten wird, der mögliche Maximalwert gesetzt.
- Eine weitere Korrekturmöglichkeit ist die Werte jeweils durch den Maximalwert zu teilen, um so eine *„Normierung“* zu erhalten.

1.7 Das Phong-Beleuchtungsmodell

Das Phong-Beleuchtungsmodell fasst den Ambienten, Diffusen und Spekularen Term zusammen zu:

$$I_{res} = I_a + I_d + I_s(+I_m + I_t) \tag{6}$$

Dabei ist I_a der Ambiente Term, der die indirekte Beleuchtung approximiert, I_d der Diffuse (Lambert) Term, der eine gleichmäßige Reflexion beschreibt und I_s der Spekulare Term, der die Spiegelnde Reflexion an rauhen Oberflächen einführt. Die Teile I_m und I_t beschreiben die ideale Spiegelung, bzw. Transmission, die nur mit Ray-Tracing möglich sind und die ich hier nicht weiter beschreiben werde.

Mit O_d als Farbkomponente (die in den einzelnen Term-Betrachtungen nicht berücksichtigt wird) ergibt sich für das Phong-Beleuchtungsmodell:

$$I_{res} = I_a k_a O_d + f_{att} I_p \left[k_d O_d (\overrightarrow{L} \cdot \overrightarrow{N}) + k_s (\overrightarrow{R} \cdot \overrightarrow{V})^{k_e} \right] \tag{7}$$

Ambienter Term

Der Ambiente Term Approximiert die indirekte Beleuchtung. Ist kein Umgebungslicht vorhanden, sind unbeleuchtete Stellen schwarz. Der Ambiente Term sorgt also für eine „Szenenaufhellung“. Es gilt:

$$I_a = k_a I_a \tag{8}$$

Eine genauere Betrachtung liefert das Radiosity-Verfahren.

Diffuser Term

Die Diffuse Beleuchtung ist abhängig vom Winkel zwischen Lichtquellenvektor und der Normalen der Oberfläche, unabhängig aber vom Standpunkt des Betrachters. Dabei gilt nach Lambert: „Je spitzer der Winkel zur Lichtquelle, desto weniger Lichtenergie gelangt

auf die Oberfläche.“ Der Diffuse Term hebt besonders das Profil eines Objektes hervor. Es ergibt sich mit k_d als „Materialeigenschaft“ für jede Lichtquelle:

$$I_d = k_d I_i \cos\theta = k_d I_i (\overrightarrow{L} \cdot \overrightarrow{N}) \tag{9}$$

Spekularer Term

Der Spekulare Term beschreibt die Reflexion an rauhen Oberflächen, man kann ihn also als „Rauheitsparameter“ auffassen, der für Glanzlichter sorgt. Es gilt für jede Lichtquelle:

$$I_s = k_s I_i \cos^{k_e}\theta = k_s I_i (\overrightarrow{R} \cdot \overrightarrow{V})^{k_e} \tag{10}$$

Dabei beschreibt k_s die Intensität (Helligkeit) und „Shininess“ k_e bestimmt die Fokussierung (Größe) der Glanzlichter. Beispiele hierfür liefert Abbildung 7.

2 Shadingmodelle für Polygone

Shadingalgorithmen führen Bildberechnungen in unterschiedlichen Qualitätsstufen durch. Dabei hat man verschiedene Shadingalgorithmen zur Auswahl: Flat-Shading, Gouraud-Shading, Phong-Shading, Ray-Tracing oder Radiosity. Generell gilt: je realistischer eine Szene berechnet werden soll, desto mehr Rechenzeit wird benötigt. Phong-Shading, ein approximatives Renderingverfahren, stellt hierbei einen Kompromiss zwischen sehr realer Darstellung und geringem Rechenaufwand dar.

2.1 Flat-Shading

Beim Flat-Shading (oder: Konstanten Shading) werden Annahmen gemacht, dass die Lichtquelle und der Betrachter jeweils im Unendlichen sind ($\Rightarrow$ der Winkel α ist überall gleich) und *ein* Punkt eine komplette Polygonfläche repräsentiert. Sind diese Bedingungen erfüllt, wird jede Polygonfläche mit der gleichen Farbe und Intensität „gefärbt“(siehe Abbildung 5).

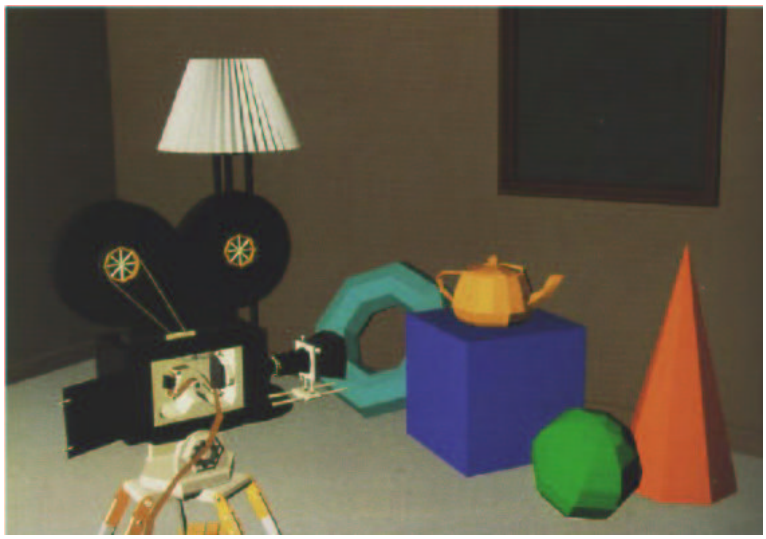

Abbildung 5: Beispiel Flat-Shading

2.2 Interpoliertes Shading

In den bisherigen Modellen wurden (fast) ausschließlich explizite Punktberechnungen ausgeführt; es wurde also jeder Bildpunkt explizit berechnet. Um den Rechenaufwand zu verringern kann man sich der *Interpolation* bedienen, die aus umliegenden Punkten Mittelwerte für einen eingeschlossenen Punkt berechnet.

2.3 Shading verbundener Polygone

Wird die Oberfläche durch ein Netz einzelner Polygone angenähert, können zwei benachbarte Polygone sehr verschieden beleuchtet/schattiert werden, sodass keine einheitliche Oberfläche entsteht, da an jeder Kante Unstetigkeiten der Größe oder Schräge auftreten können. An den Kanten würden diese Bereiche jeweils dunkler oder heller als gewünscht erscheinen. Ein „feineres“ Netz ist dabei teilweise sehr uneffizient, da der Rechenaufwand erheblich steigt.

2.4 Gouraud-Shading

Beim Gouraud-Shading werden die beim Flat-Shading auftretenden Intensitätssprünge an den Polygonkanten eliminiert, indem die Farbintensitäten im Polygoninneren ausgehend von den Farbintensitäten an den Polygonecken interpoliert werden.

Dieses Verfahren ist sehr schnell, da für die Berechnungen nur eine Addition (pro Primärfarbe) benötigt wird. Das ermöglicht eine einfache Integration in Hardware und macht dieses Verfahren schon fast zum Standard für handelsübliche Grafikkarten für den privaten Benutzer.

Der Nachteil von Gouraud-Shading ist, dass es sich nur um eine grobe Approximation der berechneten Objekte handelt und daher störende Effekte, wie fehlende Glanzpunkte und der sogenannte Mach-Band-Effekt (Eigenschaft des menschlichen Auges, sprunghafte Intensitätsunterschiede verstärkt wahrzunehmen), auftreten können (siehe Abbildung 6).

Abbildung 6: Beispiel Gouraud-Shading

2.5 Phong-Shading

Im Gegensatz zum Gouraud-Shading werden beim Phong-Shading die Intensitäten jedes einzelnen Pixels explizit berechnet. Dadurch werden die beim Gouraud-Shading auftretenden Störungen zwar reduziert, der Rechenaufwand allerdings auch erhöht.

Das beiden Algorithmen zugrundeliegende Beleuchtungsmodell setzt sich aus einem ambienten Anteil (Hintergundbeleuchtung), der diffusen und der spiegelnden Reflexion zusammen:

$$I = I_a k_a + I_L \left[k_d(L \cdot N) + k_s(N \cdot H)^n\right] \frac{1}{(r+k)} \tag{11}$$

Dabei ist $I_a k_a$ der ambiente Anteil, I_L die Intensität der Lichtquelle, $k_d(L \cdot N)$ der diffuse Anteil, $k_s(N \cdot H)^n$ der spiegelnde Anteil und $(r + k)$ der Abstand zur Lichtquelle. Besonders die Terme $(L \cdot N)$ und $(N \cdot H)^n$ sind sehr zeitaufwendig zu berechnen, da es sich um Vektorrechnungen handelt, wobei die Vektoren normiert werden müssen.

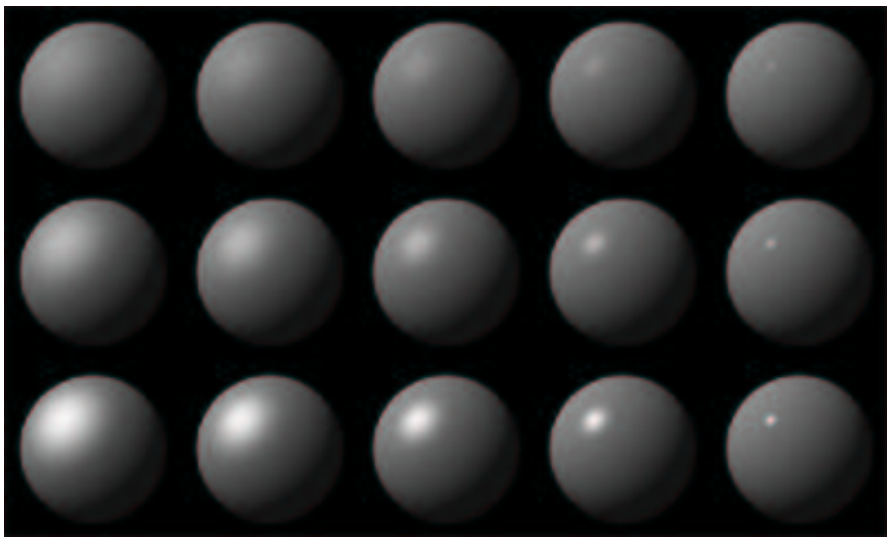

Abbildung 7: Materialeiegnschaften beim Phong-Shading

In Abbildung 7 werden in den Spalten verschiedene Spiegel-Reflexions-Exponenten ($n = 3, 5, 10, 27, 200$) und in den Zeilen verschiedene Reflexionskoeffizienten ($k_s = 0.1, 0.25, 0.5$) dargestellt. Die dadurch entstehenden unterschiedlichen Glanzpunkte und Schattierungen erzeugen beim Betrachter den Eindruck unterschiedlicher Materialien. Ist die Kugel links oben noch eher matt (vergleichbar mit Aluminium) erscheint die Kugel rechts unten schon sehr glänzend (vergleichbar mit Metall/Chrom).

Im Folgenden werden zwei Verfahren vorgestellt, die genau diesen Berechnungsaufwand verringern, indem einmal die Scanline in Segmente der Größe 2^k eingeteilt wird, bzw. Taylorreihen zur Approximation verwendet werden.

Abbildung 8: Beispiel Phong-Shading

In beiden Fällen wird davon ausgegangen, dass die zu berechnenden Polygone als Dreiecke in Bildschirmkoordinaten vorliegen. Zusätzlich wird von einer einzigen Lichtquelle, die sich im Unendlichen befindet (d.h. die Lichtstraheln fallen parallel ein), ausgegangen.

Vektorinterpolation

Man erhält die zur Intensitätsberechnung benötigten Normalenvektoren durch Interpolation der Normalenvektoren in den Eckpunkten.

Die zu bestimmenden Vektoren lassen sich anhand der Formel

$$N_i = \frac{1}{X_n - X_a}\Big[N_a(X_b - X_i) + N_b(X_i - X_a)\Big] \tag{12}$$

ermitteln.

Der somit berechnete Vektor ist im Allgemeinen nicht normiert, was zur Berechnung der Terme $(N \cdot L)$ und $(N \cdot H)$ erforderlich ist. Das heißt, es muss noch durch die Länge des Vektors geteilt werden.

$$N_i = \frac{N_a(X_b - X_i) + N_b(X_i - X_a)}{\sqrt{(X_b - X_i)^2 + (X_i - X_a)^2 + 2N_a \cdot N_b(X_b - X_i)(X_i - X_a)}} \tag{13}$$

Da die explizite Berechnung jedes einzelnen Vektors anhand dieser Formel zu aufwendig wäre, werden die Normalen gewöhnlich inkrementell berechnet, was möglich ist, da die Interpolation linear erfolgt.

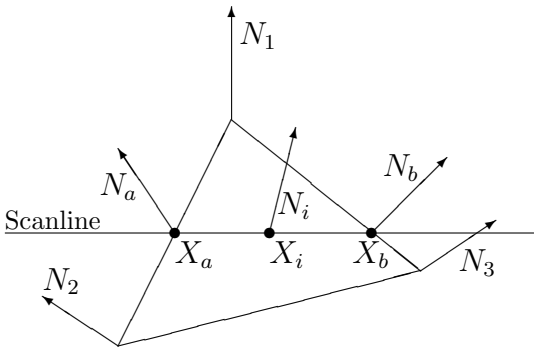

Abbildung 9: Vektorinterpolation entlang der Kanten und der Scanline

Binäre Vektorinterpolation

Im Folgenden wird gezeigt, wie sich der Rechenaufwand reduzieren läßt, wenn die Länge der Scanline, also der Abstand von X_a zu $X - B$ in der Größenordnung 2^k liegt.

Ist X_i der Mittelpunkt des Intervalls $[X_a, B_b]$ – im Folgenden als $X_{1/2}$ bezeichnet – so sind die beiden Terme $(X_b - X_i)$ und $(X_i - X_a)$ identisch und lassen sich kürzen. Für die Normale in diesem Punkt ergibt sich:

$$N_{1/2} = \frac{N_a + N_b}{\sqrt{2 + 2N_a \cdot N_b}} \tag{14}$$

und für das Skalarprodukt $(N \cdot H)_{1/2}$:

$$N_{1/2} = \frac{(N \cdot H)_a + (N \cdot H)_b}{\sqrt{2 + 2N_a \cdot N_b}} \tag{15}$$

Analog lässt sich die Normale auch wieder für die Mittelpunkte der Teilstücke der Scanline berechnen.

Schließlich kann man alle Vektorrechnungen eliminieren, sodass die gesamte Rechnung skalar wird.

Ist der Abstand von den Punkten X_a und X_b gleich einer Zweierpotenz 2^k, so lässt sich dieses Verfahren rekursiv auf alle berechneten Normalen anwenden.

Im Allgemeinen trifft es aber nicht zu, dass die Scanline die erforderliche Länge von 2^k hat. Hier kann man eine der folgeneden Möglichkeiten anwenden:

- Es lässt sich durch Extrapolation ein Vektor (N_i) berechnen, um die Länge der Scanline auf $(M+i) = 2^k$ zu erweitern, worauf dann wieder die binäre Interpolation angewendet werden kann. (Abbildung 11)
- Das Intervall $[X_a, X_b]$ kann in Teilintervalle der Größenordnung 2^k geteilt werden.

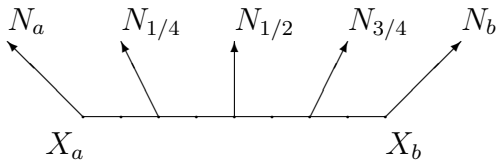

Abbildung 10: Binäre Vektorinterpolation

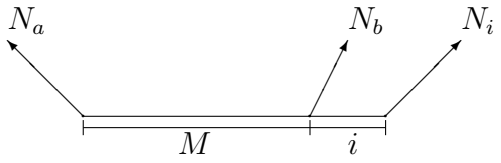

Abbildung 11: Vektorextrapolation

Phong-Shading mit Taylor-Reihen

Auch bei diesem Ansatz wird das oben beschriebene Beleuchtungsmodell mit einem ambienten, diffusen und spiegelnden Anteil verwendet (Vgl. Gleichung (6)). Die Lichtquelle befindet sich auch hier im Unendlichen.

Diffuser Lichtanteil:

$$I_d = I_L k_d (L \cdot N) \tag{16}$$

Die Normalen eines Polygons lassen sich durch die Normalen in den Eckpunkten auf folgende Weise interpolieren:

$$N(x, y) = Ax + By + C \tag{17}$$

Diese Vektoren sind nicht zwingend normiert. Darum muss noch durch deren Länge geteilt werden. Eingesetzt in obige Gleichung ergibt sich:

$$I_d = I_L k_d \left(\frac{L}{|L|} \cdot \frac{Ax + By + C}{|Ax + By + C|} \right) = I_L k_d \left(\frac{L \cdot Ax + L \cdot By + L \cdot C}{|L| \cdot |Ax + By + C|} \right) \tag{18}$$

Diese Gleichung lässt sich umformen zu:

$$I_d = I_L k_d \left(\frac{ax + by + c}{\sqrt{dx^2 + exy + fy^2 + gx + hy + i}} \right) \tag{19}$$

mit

$$a = \frac{L \cdot A}{|L|},\ b = \frac{L \cdot B}{|L|},\ c = \frac{L \cdot C}{|L|} \tag{20}$$

$$d = A \cdot A,\ e = 2A \cdot B,\ f = B \cdot B,\ g = 2A \cdot C,\ h = 2B \cdot C,\ i = C \cdot C \tag{21}$$

Mit diesem Ansatz und der Differenzmethode (s.u.) verringert sich der Aufwand auf drei Additionen, eine Division und eine Quadratwurzel pro Pixel, was eine erhebliche Verbesserung des ursprünglichen Berechnungsaufwandes darstellt. Allerdings sind die Division und die Wurzel immer noch zu aufwendig, um in einem Echtzeitsystem verwendet werden zu können.

Mit der *Differenzmethode* lassen sich Polynome beliebigen Grades so modifizieren, dass keine Multiplikationen mehr benötigt werden.

Auf diesem Ansatz aufbauend wurde ein anderes Verfahren, unter Verwendung der Taylor-Formel, benutzt. Die eindimensionale Form der Taylor-Formel wird dazu benutzt, um beliebige stetige und differenzierbare Funktionen mit Polynomen zu approximieren. Auch in diesem zweidimensionalen Fall liefert die Formel nur eine Approximation. Die Taylor-Formel liefert umso genauere Approximationen, je höher der Grad des verwendeten Polynoms gewählt wird. Voraussetzung dafür ist, dass sich die ursprüngliche Funktion entsprechend oft differenzieren lässt. In diesem Fall begnügt man sich mit einem Polynom vom Grad zwei, was ein Kompromiss zwischen Genauigkeit der Approximation und der benötigten Rechenzeit ist.

Die Taylorformel zweiter Ordnung für eine Funktion mit zwei Variablen ist:

$$f(x+h, y+k) \tag{22}$$

$$= f(x,y) + \frac{1}{1!}\left(\frac{df(x,y)}{dx}h + \frac{df(x,y)}{dy}k\right) + \frac{1}{2!}\left(\frac{d^2f(x,y)}{dx^2}h^2 + \frac{d^2f(x,y)}{dxdy}hk + \frac{d^2f(x,y)}{dy^2}k^2\right) \tag{23}$$

Verschiebt man das Dreieck so, dass sich der Punkt $(0,0)$ in dessen Zentrum befindet, ergibt sich:

$$I_d(x,y) = T_5x^2 + T_4xy + T_3x^2 + T_2x + T_1y + T_0 \text{ mit} \tag{24}$$

$$T_5 = \frac{3ig^2 - 4cdi - 4agi}{8i^2\sqrt{i}} \tag{25}$$

$$T_4 = \frac{3cgh - 2cei - 2bgi - 2ahi}{4i^2\sqrt{i}} \tag{26}$$

$$T_3 = \frac{3ch^2 - 4cfi - 4bhi}{8i^2\sqrt{i}} \tag{27}$$

$$T_2 = \frac{2ai - cg}{2i\sqrt{i}} \tag{28}$$

$$T_1 = \frac{2bi - ch}{2i\sqrt{i}} \tag{29}$$

$$T_0 = \frac{c}{\sqrt{i}} \tag{30}$$

$$\tag{31}$$

Durch diese Umformung und die Verwendung der Differenzmethode lässt sich dieser term mit nur zwei Additionen auswerten.

Fehlt noch der spiegelnde Anteil in der Beleuchtungsgleichung:

$$I_s = K_s(N \cdot H)^n \tag{32}$$

Der Vektor H stellt die Winkelhalbierende vom Vektor L (zur Lichtquelle) und V (zum Betrachter) dar. Also:

$$H = \frac{L+V}{|L+V|} \tag{33}$$

Um diesen Vektor nicht explizit für jedes Pixel neu berechnen zu müssen, wird zunächst angenommen, der Betrachter befindet sich in unendlicher Entfernung, d.h. die Vektoren zum Betrachter sind alle parallel.

Indem man wie oben, mit Hilfe der Taylorformel, den Term $(N \cdot H)$ durch ein Polynom zweiten Grades approximiert und die Gleichung wiederum mit der Differenzmethode auflöst, lässt sich der Term mit einem Aufwand von zwei Additionen lösen.

Zusammen ergibt sich also für die komplette Beleuchtungsgleichung

$$I = I_a + I_d + I_s \tag{34}$$

ein Gesamtaufwand von sechs Additionen und einer Speicherreferenz.

Durch die parallelen Strahlen zum Betrachter ergibt sich allerdings (vor allem für planare) ein sehr unrealistisches Bild, da Polygone wie beim Flat-Shading konstant schattiert werden.

Darum wird der Ansatz erweitert und der Vektor zum Betrachter ähnlich wie die Normalenvektoren interpoliert:

$$V = Dx + Ey + F \tag{35}$$

Dadurch ändert sich der spiegelnde Term geringfügig zu:

$$I_s = K_s \left(N(x,y) \cdot H(x,y)\right)'' \text{ bzw.} \tag{36}$$

$$I_s = K_s \left(\frac{(Ax+By+C) \cdot (Dx+Ey+F)}{|Ax+By+C| \cdot |Dx+Ey+F|} \right)'' \tag{37}$$

Durch die oben verwendeten Methoden und Approximation mit der Tylorformel lässt sich auch diese Gleichung mit nur zwei Additionen und einer Speicherreferenz lösen. Allerdings wird die Berechnung der Tylor-Koeffizienten T_i etwas aufwendiger.

Vergleich: Gouraud-Shading ↔ Phong-Shading

Zum Vergleich des Rechenaufwandes wurden die CPU-Zeiten einer DEC microVAX-II zur Berechnung eines Objektes mit 14620 Polygonen, in einer Auflösung von 2048×2048 Pixeln verglichen:

Gouraud-Shading	406 Sekunden
Phong-Shading	3900 Sekunden
Phong-Shading mit Tylor (Betrachter im Unendlichen)	767 Sekunden (davon 119 Sekunden für Tylor-Koeffizienten)
Phong-Shading mit Tylor (Betrachter im Endlichen)	850 Sekunden (davon 202 Sekunden für Tylor-Koeffizienten)

2.6 Probleme bei interpoliertem Shading

Durch interpoliertes Shading kann man zwar den Rechenaufwand – je nach Grad der Interpolation – stark verringern, erzeugt damit aber auch unschöne Störeffekte:

- Unabhängig von der Qualität der Oberflächenschattierung bleibt ein Polygoner Schattenriss, den man nur durch noch kleinere Polygone verbessern kann, was aber den Rechenaufwand erhöhen würde.

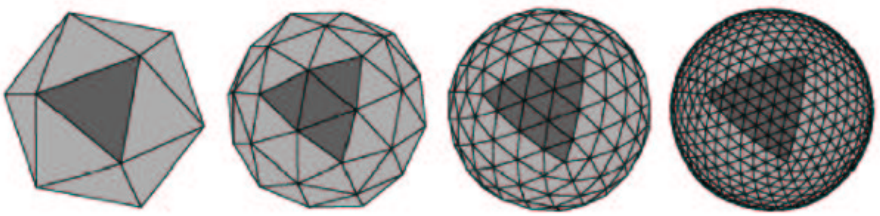

Abbildung 12: Polygoner Schattenriss

- Perspektivische Verschiebung kann auftreten, da die Schattierung erst nach der Umformung (Translation) vorgenommen wird. Schattierungen sind von der Ausrichtung eines Polygons abhängig; nach einer Rotation kann die Schattierung anders aussehen.
- Probleme an gemeinsamen Kanten treten auf, wenn sich zwei Polygone einen Punkt teilen, den das dritte Polygon nicht teilt. (Siehe Abbildung 13)
- Unrepräsentative Normalenvektoren, da alle Spitzenvektoren parallel sind (Siehe Abbildung 14).

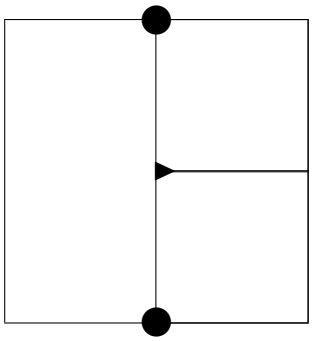

Abbildung 13: Probleme an gemeinsamen Kanten

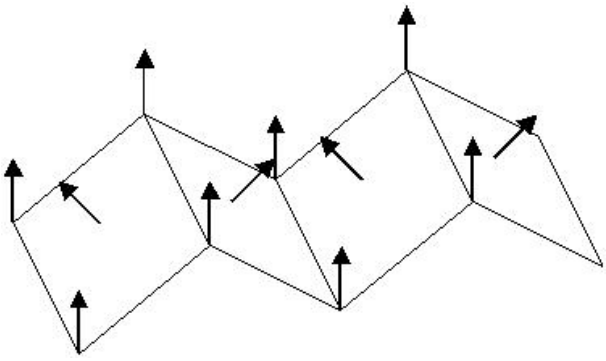

Abbildung 14: Unrepräsentative Normalenvektoren

3 Schattenmodelle

Analog zu Oberflächen-Sichtbarkeits-Algorithmen, die feststellen welche Bildteile vom Betrachter aus gesehen werden können, stellt der *Schatten-Algorithmus* fest, welche Objekte – oder Teile davon – beleuchtet werden. Derselbe Algorithmus kann also einmal vom Betrachter- und einmal vom Beleuchtungspunkt aus angewandt werden.

3.1 Schatten der Scan-Line-Generation

Bei Schatten der Scan-Line-Generation nimmt man die Lichtquelle als Mitte der Projektion an. Potentielle „Schattenverursacher“ sind die jeweiligen Polygonkanten, bei deren Abtastung (Kreuzung: Scan-Line – Polygonkante) die entsprechenden Flächen (Zeilen) dunkler gefärbt werden um so den Schattenwurf zu simulieren (Siehe Abbildung 15).

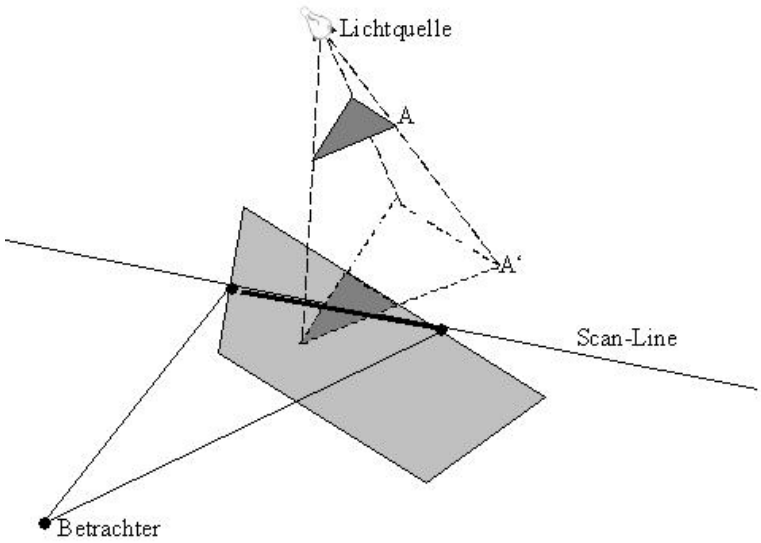

Abbildung 15: Schatten der Scan-Line-Generation

3.2 Präzisions-Algorithmus (two-pass)

Beim two-pass-Algorithmus wird derselbe Algorithmus (Vgl. oben) zweimal hintereinander ausgeführt:

1. vom Punkt der Lichtquelle: es wird festgestellt, welche Teile beleuchtet (und schattiert werden) – der Schatten wird bestimmt

2. vom Betrachtungspunkt: es wird festgestellt welche Teile dieser Umgebung vom Betrachter gesehen werden können

3.3 Schattenkörper

Schatten können erst durch die Erzeugung von Schattenkörpern erzeugt werden. Diese Schattenkörper werden für jedes Objekt durch die Lichtquelle und das Objekt selbst definiert. Um Rechenaufwand zu sparen, werden Schattenkörper *nur* für beleuchtete Polygone erstellt.

3.4 Schatten-Algorithmus (z-buffer, two-pass)

Analog zum two-pass-Algorithmus werden auch hier in zwei Durchgängen Sichtbarkeits- bzw. Beleuchtungsbestimmungen durchgeführt. Dabei wird durch explizite Bildberechnungen festgestellt, ob eine Oberfläche schattiert wird, oder nicht. Die Erweiterung des z-buffer-Algorithmus stellt (pro Lichtquelle) eine sogenannte „shadow-map“ zusammen, die Koordinaten von Objekten beinhaltet, die angeben, ob ein Punkt im Schatten liegt oder nicht. Nach dieser shadow-map können schließlich die Schattierungen realisiert werden.

3.5 Schatten-Algorithmus (bei globaler Beleuchtung)

Um sehr komplexe Schattierungen bei globaler Beleuchtung zu berechnen, bedient man sich beispielsweise dem Raytracing oder Radiosity, mit denen man sehr eindrucksvolle Schattierungen erzielen kann. Auf diese beiden Verfahren gehe *ich* nicht näher ein.

4 Physikalisch basierte Beleuchtungsmodelle

Alle Modelle, die im Rechner Anwendung finden, beruhen auf physikalisch basierten Grundlagen.

4.1 Verbesserung des Oberflächenmodells

Im *Torrance-Sparrow-Modell* wird die Oberfläche als Ansammlung mikroskopisch kleiner Facetten – jede als kleiner, glatter Spiegel – angenommen. Diese Geometrie und Verteilung um eine Lichtquelle im Unendlichen bestimmen die Intensität und Richtung der Reflexion.

4.2 Die Minimalverteilungs-Funktion

Wenn Polygone als perfekte kleine Spiegel angenommen werden, kann dieses Modell vereinfacht werden, wenn man nur die Polygone betrachtet, deren Normalenvektor in Richtung der Winkelhalbierenden $\vec{H}$ (zwischen $\vec{V}$ und $\vec{L}$) zeigen (Siehe Abbildung 16).

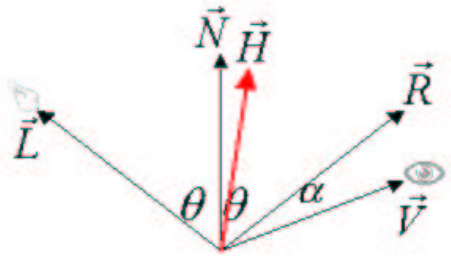

Abbildung 16: Minimal-Verteilung

4.3 Der geometrische Abschwächungsfaktor

Manche Polygone spiegeln das gesamte Licht. Andere spiegeln es teilweise auch auf andere Polygone, die damit wiederum Schatten werfen und selbst weiterreflektieren. Um dies zu beschreiben wird der Abschwächungsfaktor $\vec{G}$ zwischen 0 (total schattiert) und 1 (kein Schatten) eingeführt (Siehe Abbildung 17).

4.4 Fresnel-Bedingung

Die Fresnel-Bedingung ist anwendbar für unpolarisiertes Licht in Abhängigkeit vom Einfallswinkel des Lichtes (Spiegelung bei sehr flachen Winkeln). Blinn machte vereinfachende Annahmen, dass sich der Betrachter und die Lichtquelle jeweils im Unendlichen befinden. Damit kann man den Winkel zwischen Betrachter- und Lichtquellenvektor überall als gleich annehmen.

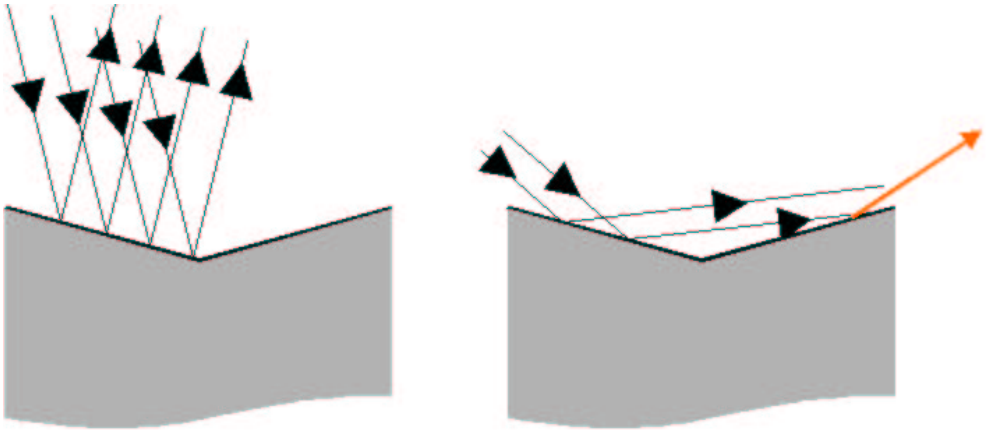

Abbildung 17: geometrische Abschwächung

Im Vergleich zwischen Phong- und Torrance-Sparrow-Modell (Abbildung 18) kann man erkennen, dass sich bei „steilen Winkeln“ (im Beispiel 30°) sehr ähnliche Ergebnisse einstellen, sich bei „flachen Winkeln“ (im Beispiel 70°) im Torrance-Sparrow-Modell eine Beleuchtungs-Spitze der Spiegel-Reflexion einstellt.

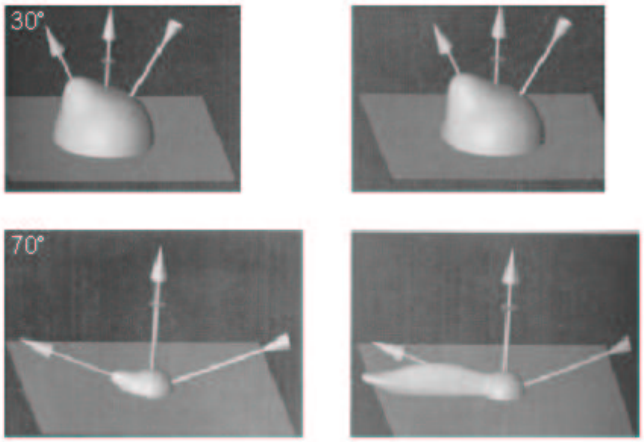

Abbildung 18: Phong- vs. Torrance-Sparrow-Modell

links: Phong- rechts: Torrance-Sparrow-Modell

5 Verteilte und Ausgedehnte Lichtquellen

5.1 Verteilte und Ausgedehnte Lichtquellen

Bisher sind alle Betrachtungen auf der Grundlage einer einzelnen Lichtquelle entstanden. In der Realität ist dies allerdings sehr selten der Fall und es wird die Betrachtung von *ausgedehnten* und *verteilten Lichtquellen* nötig.

Eine ausgedehnte Lichtquelle kann man sich als Ansammlung vieler einzelner Punktlichtquellen vorstellen.

Damit kann wieder der Ansatz aus Kapitel 1.6 angewandt werden, bei dem alle Einzellichtquellen zu einer resultierenden Intensität summiert werden.

5.2 Schatten bei verteilten und ausgedehnten Lichtquellen

Aus diesen verteilten oder ausgedehnten Lichtquellen ergeben sich Besonderheiten bei der Schattenbetrachtung. Schatten, die durch die eine Lichtquelle erzeugt werden, können durch eine andere wieder beleuchtet werden, sodass weichere Schatten entstehen. Die Schattenflächen, die von allen Lichtquellen erzeugt werden (besonders dunkel) heißen *Kernschatten* (oder: *Penumbra*), alle anderen heißen *Halbschatten* (oder: *Umbra*).

Am Beispiel der verteilten Lichtquellen (Abbildung 19) kann man auf der linken Seite die beiden einzelnen Schatten erkennen, die durch die beiden Lichtquelle und die Dose erzeugt werden. Die beiden Schatten überlagern sich nicht so wie im Beispiel auf der rechten Seite. Durch die Überlagerung (durch Zusammenrücken der Lichtquellen) der Schatten entstehen Halb- und Kernschatten.

Das Beispiel der ausgedehnten Lichtquelle (Abbildung 20) wie man es zum Beispiel von Leuchtstoffröhren kennt, zeigt sehr weiche Schattenübergänge und es ist keine klare Abgrenzung wie im vorigen Beispiel mehr möglich. Dieser Effekt entsteht, da man sich das oben genannte Modell der verteilten Lichtquellen hier für sehr viele Lichtquellen vorstellen kann, die alle Halb- und Kernschatten erzeugen, die dann in weicher Abstufung zusammenfallen.

Abbildung 19: Schatten bei verteilten Lichtquellen

Abbildung 20: Schatten bei ausgedehnten Lichtquellen

Zusammenfassung

- Es gibt verschiedene Beleuchtungsmodelle, die den Rahmen der Berechnungen liefern.
- Verschiedene, immer weiter verbesserte, Shadingmodelle bieten mehr oder weniger gute Kompromisse zwischen der Qualität der erzeugten Bilder und der dazu benötigten Rechnerzeit.
- Schatten können dem Beleuchtungsrahmen entsprechend (auch vereinfacht) berechnet werden.
- Die Modelle, die im Rechner Anwendung finden und simuliert werden, basieren auf physikalischen Beleuchtungsmodellen.
- Ausgedehnte und verteilte Lichtquellen verursachen einen erhöhten Rechenaufwand, da sie wie „viele“ Puktlichtquellen berechnet werden müssen.

Literatur

[1] **Computer Graphics: Principles an Practice**
Foley · van Dam · Feiner · Hughes

[2] **Fast Phong-Shading**
Christian Bäurle

[3] **Lokale Beleuchtungsmodelle**
Oliver Deussen

[4] **Visualisierung der dreidimensionalen Strukturen**
Stefan Seifert

[5] **Visualisierung von 3D-Modellen**
`www.aecweb.de/m/f-3dvis.htm`

[6] **Phong Shading by binary Interpolation**
S. Ouyang, D.E. Maynard
Computer & Graphics, Vol. 20, No.6

[7] **Fast Phong Shading**
Gary Bishop, David Weimer
SIGGRAPH 1986

[8] **Computergrafik**
W.D. Fellner
2. Auflage
BI 1991

Abbildungsverzeichnis