Bachelorarbeit
Surface Caching und Lightmapping

Markus Pawellek markuspawellek@gmail.com

Inhaltsverzeichnis

1 Grundlagen und Methoden

In den folgenden Kapiteln wird eine grundlegende Einführung und Definition der Verfahren gegeben, die für den weiteren Verlauf dieser Arbeit von Belang sind. Viele dieser Themen können hier nur angerissen werden, da ihre komplette Behandlung den Rahmen und das Ziel des Themas verfehlen würde. Für eine genauere Einführung in die einzelnen Themengebiete, wird dem Leser geraten sich mit den genannten Quellen auseinander zu setzen.

1.1 Szenengeometrie

Um in einem Computer ein zweidimensionales Bild einer dreidimensionalen Umwelt oder auch »Szene« (engl.: *scene*) zu generieren, benötigen wir ein mathematisches Modell, welches diese hinreichend gut beschreibt. Wir wollen uns einen Beobachter vorstellen, der die Umwelt und die Objekte in ihr von einem bestimmten Punkt im Raum aus betrachtet. Für viele Algorithmen, die diese Aufgabe lösen, ist dabei vor allem das Verhalten von Licht auf den Oberflächen dieser Objekte wichtig [9, 18, 20]. Der Einfachheit halber wollen wir davon ausgehen, dass Licht nur in den oberen Schichten eines Körpers mit dessen Material wechselwirkt. Diese Annahme reduziert die Komplexität des mathematischen Modells, die dann noch aus der Charakterisierung der Oberflächen besteht.

Die Oberflächen realer physikalischer Körper sind im Allgemeinen beliebig geformt und können nicht in geschlossener Form durch eine Gleichung beschrieben werden. Dennoch lassen sie sich im analytischen Sinne durch einfachere Hyperflächen (engl.: *shape* [18, S. 123 ff]) im Raum approximieren [10]. Für die Bildgenerierung wählt man für solch eine Fläche meist ein Dreieck. Es ist einfach zu beschreiben und flexibel genug um die meisten Objekte beliebig genau zu approximieren [2, 13]. Dabei wollen wir entartete Dreiecke, die nur aus einem Punkt oder einer Strecke bestehen, ausschließen, da sie für die betrachteten Render-Verfahren nicht darstellbar sind.

DEFINITION 1.1: (Dreieck)

Ein Dreieck $\triangle$ wird durch eine Sequenz (A, B, C) von Eckpunkten in $\mathbb{R}^3$, für die die Menge $\{B - A, C - A\}$ linear unabhängig ist, charakterisiert. Seien weiterhin

$$M \coloneqq \big\{(u, v) \in [0, 1]^2 \;\big|\; u + v \leq 1\big\}$$

$$\varphi\colon M \to \mathbb{R}^3, \qquad \varphi(u, v) \coloneqq (1 - u - v)A + uB + vC$$

Dann ist die Menge der Punkte S des Dreiecks gegeben durch $\operatorname{im}\varphi$ und (M, φ) stellt deren Standardparametrisierung dar. Der Notation wegen, identifizieren wir $\triangle$ mit S und definieren für alle $(u, v) \in M$

$$\triangle(u, v) \coloneqq \varphi(u, v)$$

Die baryzentrischen Koordinaten (u, v, w) eines Punktes $x \in \triangle$ sind weiterhin durch die folgenden Eigenschaften gegeben.

$$(u, v) \in M, \qquad w = 1 - u - v, \qquad \triangle(u, v) = x$$

Die analytische äußere Normale oder auch Normale ist definiert durch

$$\mu \coloneqq \frac{(B - A) \times (C - A)}{\|(B - A) \times (C - A)\|}$$

Jedes Dreieck besitzt auf seiner gesamten Fläche eine eindeutige konstante äußere analytische Normale. Für die Simulation von globalen Beleuchtungseffekten ist diese Eigenschaft jedoch ein Nachteil, weil die Beleuchtung eines Objektes stark vom Verlauf seiner Normalen abhängt . Nähern wir ein Objekt durch $n \in \mathbb{N}$ Dreiecke an, so nähern wir auch den Normalenverlauf des Objektes durch die stückweise konstanten Normalen der Dreiecke an. Der Fehler dieser Approximation tritt in Form eines facettenhaften Musters auf, welcher für das menschliche Auge gut erkennbar ist [18, S. 166]. In Abbildung 1 wird dieser Effekt genauer an einem Beispiel demonstriert. Folglich muss darauf geachtet werden, dass bei stetig differenzierbaren Oberflächen, die Stetigkeit der Normalen erhalten bleibt [10, S. 39 ff].

DEFINITION 1.2: (Normalen-Funktion / Shading-Normale)

Sei $\triangle$ ein Dreieck mit der Normalen μ. Dann wird $\nu\colon \triangle \to \mathcal{H}^2_\mu$ als Normalen-Funktion oder auch Shading-Normale von $\triangle$ bezeichnet.

Abbildung 1: Die Bilder zeigen die gerenderte »Dragon«-Szene. Das rechte Bild entspricht dem roten Bereich des Linken. In dieser Szene werden die Normalen des Objektmodells durch die analytischen Normalen der Dreiecke angenähert. Da das Modell sehr fein trianguliert ist, fällt dies im ersten Bild nicht auf. Zoomt man jedoch mit der Kamera heran werden die Fehler durch die Approximation deutlich und die einzelnen Dreiecke sind mit dem menschlichen Auge auszumachen.

Nach [18, S. 166, 584 ff] und [1, S. 38 ff, 183 ff] sind die typischen Verfahren für eine genauere Interpolation die »Vertex-Shading-Normalen« und das »Bump Mapping« (engl.: *bumb mapping*). Formal gesehen werden bei beiden Verfahren die Normalen der vorhandenen Geometrie durch eine Normalen-Funktion ersetzt.

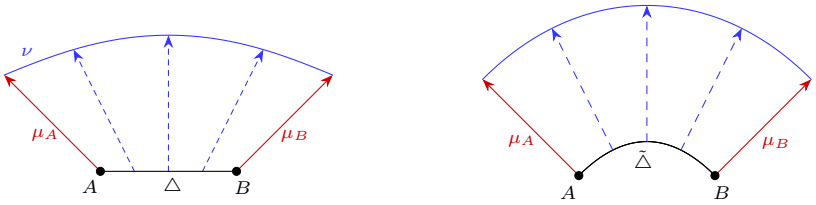

Abbildung 2: Die erste Skizze auf der linken Seite zeigt den Verlauf einer Vertex-Shading-Normalen ν anhand eines Beispiels. A und B sind dabei zwei Eckpunkte eines Dreiecks $\triangle$. μ_A und μ_B sind die jeweilig gegebenen Vertex-Normalen an den Eckpunkten. Im rechten Bereich der Abbildung ist die durch ν approximierte gekrümmte Fläche $\tilde{\triangle}$, auf der die Shading-Normale ν senkrecht steht, eingezeichnet.

DEFINITION 1.3: (Vertex-Shading-Normale)

Seien $\triangle$ ein Dreieck mit der Normalen μ und $\mu_A, \mu_B, \mu_C \in \mathcal{H}^2_\mu$ Normalen an den

Eckpunkten des Dreiecks. Sei weiterhin ν eine Shading-Normale auf $\triangle$, sodass für alle $x \in \triangle$ mit den baryzentrischen Koordinaten (u, v, w) gilt

$$\nu(x) \coloneqq \frac{w\mu_A + u\mu_B + v\mu_C}{\|w\mu_A + u\mu_B + v\mu_C\|}$$

Dann ist ν stetig und man nennt es eine Vertex-Shading-Normale von $\triangle$.

Durch das Setzen der Normalen an den Eckpunkten (engl.: *vertex*) eines Dreiecks können wir sicher gehen, dass der Verlauf der Normalen stetig von einem Dreieck zu einem anderen übergeht. Ein beipielhafter Verlauf einer Vertex-Shading-Normalen wird in Abbildung 2 gezeigt. Zu beachten ist, dass die Vektoren der Shading-Normale im Allgemeinen nicht mehr senkrecht auf der Geometrie stehen. Dadurch kann es, wie in [18, S. 574 ff] und [20, S. 150 ff] gezeigt, zu Artefakten bei der Generierung des Bildes kommen.

Der letzte Schritt zur vollständigen Beschreibung der Szenengeometrie, besteht darin, eine Menge von Dreiecken zu bilden, sodass diese ein physikalisches Objekt gut genug approximieren. In [18] und [2] werden solche Mengen auch »Meshs« (engl.: *triangle mesh*) genannt. Wir wollen hier eine unübliche Definition geben.

DEFINITION 1.4: (Mesh)

Seien $n \in \mathbb{N}$ und Dreiecke $\triangle_i$ mit Shading-Normalen ν_i für alle $i \in \mathbb{N}, i \leq n$. Weiterhin definieren wir

$$\mathcal{T} \coloneqq \bigcup_{i=1}^{n} \triangle_i \qquad \mathcal{A} \coloneqq \{x \in \mathcal{T} \mid \text{es gibt genau ein } i \in \mathbb{N}, i \leq n, \text{ sodass } x \in \triangle_i\}$$

$$\nu \colon \mathcal{T} \to \mathbb{S}^2 \cup \{0\}, \qquad \nu(x) \coloneqq \begin{cases} \nu_i(x) & : x \in \mathcal{A} \text{ mit } x \in \triangle_i \text{ für } i \in \mathbb{N}, i \leq n \\ 0 & : \text{sonst} \end{cases}$$

Dann nennen wir $\mathcal{T}$ eine Mesh mit Shading-Normaler ν, wenn für σ-fast-alle $x \in \mathcal{T}$ auch $x \in \mathcal{A}$ gilt.

Die Definition der Mesh verbietet, dass die Schnittpunkte der Dreiecke eine eigene Fläche im Raum bilden. So können wir sicher gehen, dass die Integration über die Punkte der Mesh eine eindeutige Lösung ergibt. Es ist jedoch erlaubt, dass sich Dreiecke in einem Punkt oder einer Strecke schneiden dürfen. Dies ist auch nötig, um komplexere Objekte formen zu können. In der Praxis ist die genannte Bedingung im Allgemeinen erfüllt und folglich auch keine fundamentale Einschränkung [2, 13, 18, 20].

1.2 Streuung von Licht an Oberflächen

Im letzten Abschnitt wurden die Grundbausteine einer Szene eingeführt, die deren Geometrie beschreiben. Wie bereits erwähnt, benötigen wir jedoch zusätzlich die Informationen, auf welche Art und Weise Licht mit den Oberflächen der Objekte interagiert. Dieses Verhalten wird durch die sogenannte »Bidirektionale Streuungsverteilungsfunktion« (engl.: *bidirectional scattering distribution function*, BSDF), die sich aus einer »Bidirektionalen Reflektanzverteilungsfunktion« (engl.: *bidirectional reflectance distribution function*, BRDF) und einer »Bidirektionalen Transmissionsverteilungsfunktion« (engl.: *bidirectional transmittance distribution function*, BTDF) zusammensetzt, geklärt. Strukturierte Einführungen zu diesem Thema erhält man in [1, 4, 14, 18, 20].

DEFINITION 1.5:

Sei $\mu \in \mathcal{S}^2$ die Normale am Punkt einer Oberfläche. Dann ist eine BSDF bezüglich μ gegeben durch eine integrierbare Abbildung $f\colon \mathcal{S}^2 \times \mathcal{S}^2 \to [0,\infty)$ mit den folgenden Eigenschaften.

(i) *Für alle $\omega_\mathrm{i} \in \mathcal{S}^2$ und alle $\omega_\mathrm{o} \in \mathcal{H}^2_\nu$ mit $\nu \coloneqq \operatorname{sgn}(\langle \mu, \omega_\mathrm{i} \rangle) \cdot \mu$ gilt*

$$f(\omega_\mathrm{i}, \omega_\mathrm{o}) = f(\omega_\mathrm{o}, \omega_\mathrm{i}) \qquad \textit{(Helmholtz-Reziprozität)}$$

(ii) *Für alle $\omega_\mathrm{i} \in \mathcal{S}^2$ gilt*

$$\int_{\mathcal{S}^2} f(\omega_\mathrm{i}, \omega)\, |\langle \mu, \omega \rangle| \;\mathrm{d}\sigma(\omega) \leq 1 \qquad \textit{(Energieerhaltung)}$$

Eine BSDF f gibt an, welcher Anteil des einfallenden Lichtes aus der Richtung $\omega_\mathrm{i} \in \mathcal{S}^2$ in die Richtung $\omega_\mathrm{o} \in \mathcal{S}^2$ gestreut wird. Gilt dabei $\omega_\mathrm{o} \in \mathcal{H}^2_\nu$, so befindet sich ω_o in der gleichen Hemisphere bezüglich μ wie ω_i und es handelt sich um eine Reflexion des Lichtes. Der verbleibende Fall beschreibt die Transmission. Die BSDF stellt damit eine Verallgemeinerung der idealen Reflexion und Brechung an Oberflächen dar. [18, S. 571 ff] und [20, S. 135 ff] weisen zudem nach, dass die Helmholtz-Reziprozität nur für den reflektierten Anteil des Lichtes gilt. In Abbildung 3 ist das Beispiel einer typischen BSDF gezeigt, die keine Transmission des Lichtes zulässt [18, S. 509 ff].

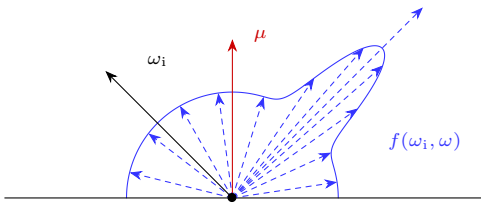

Abbildung 3: Die Abbildung zeigt die Verteilung des reflektierten Lichtes für alle $\omega \in \mathcal{H}^2_\mu$ bezüglich einer konstanten Einfallsrichtung $\omega_\mathrm{i} \in \mathcal{H}^2_\mu$ einer BSDF f bezüglich der Oberflächennormalen $\mu \in \mathcal{S}^2$.

Die hier eingeführten Funktionen für die Charakterisierung von Materialien können auf verschiedene Weisen verallgemeinert werden. Zu beachten ist vor allem die fehlende Abhängigkeit von der Wellenlänge des einfallenden und der des gestreuten Lichtes, welche Effekte wie Dispersion, Irisieren und Lumineszenz verhindert. Eine weiteres physikalisches Phänomen stellt die Volumenstreuung dar. Bei der Betrachtung dieser wird ein Material durch die sogenannte »BSSRDF« beschrieben [18, S. 671 ff].

BSDFs sind im Allgemeinen nicht in geschlossener Form notierbar [18, S. 507 f]. Aus diesem Grund wollen wir für die Konstruktion realistischer Materialien wie bei der Approximation von Oberflächen verschiedene einfache BSDFs zu Grunde legen. In den beiden folgenden Beispielen handelt es sich um die BSDFs f bezüglich der Normalen $\mu \in \mathcal{S}^2$ für die lambertsch diffuse Reflexion (engl.: *lambertian reflection*, [18, S. 531 f]) und für die ideale Reflexion (engl.: *specular reflection*, [20, S. 144 f]). Weitere BSDF-Modelle sind in [4, 18, 20] zu finden. Es seien $\omega_\mathrm{i}, \omega_\mathrm{o} \in \mathcal{S}^2$ mit $\nu \coloneqq \operatorname{sgn}(\langle \mu, \omega_\mathrm{i} \rangle) \cdot \mu$ gegeben.

$$f(\omega_\mathrm{i}, \omega_\mathrm{o}) = \frac{1}{\pi} \mathbb{1}_{\mathcal{H}^2_\nu}(\omega_\mathrm{o}) \qquad \text{(lambertsch diffuse Reflexion)}$$

$$f(\omega_\mathrm{i}, \omega_\mathrm{o}) = \frac{\delta_{\omega_\mathrm{o}}(2\,\langle \mu, \omega_\mathrm{i}\rangle\,\mu - \omega_\mathrm{i})}{|\langle \mu, \omega_\mathrm{i}\rangle|} \qquad \text{(ideale Reflexion)}$$

1.3 Beleuchtung und Szene

Die BSDF an einem Punkt beschreibt lediglich die Streuung des einfallenden Lichtes. Um den Render-Verfahren einen Sinn zu geben, benötigen wir demnach Lichtquellen. In [18, S. 707 ff] und [8] werden verschiedene Arten von Lichtquellen definiert, implementiert und gesampled. Der einfachste Weg Lichtquellen einzuführen, besteht in einer abstrakten Formulierung, die unabhängig von der Szenengeometrie agiert. Wir wollen eine sogenannte »Umgebungsbeleuchtung« (engl.: *hdr environment map* oder *infinite area light*, [18, S. 737 f]) einführen.

DEFINITION 1.6: (Umgebungsbeleuchtung)

Eine Umgebungsbeleuchtung ist definiert als eine integrierbare Abbildung $f\colon (0,\infty)\times \mathcal{S}^2 \to [0,\infty)$.

Diese Funktion beschreibt das Licht verschiedener Wellenlängen, welches von einer Kugeloberfläche mit quasi unendlich großem Radius in die Szene ausgesandt wird. Für einen Punkt der Szene, ist dessen Beleuchtung durch diese Funktion also unabhängig von dessen Position. Diese Tatsache wird in Abbildung 4 wieder an einem Beispiel gezeigt.

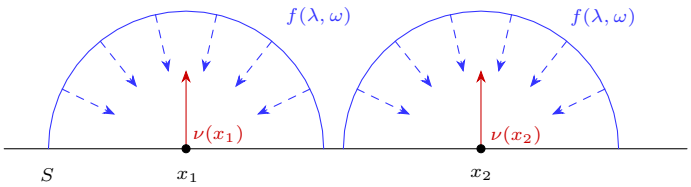

Abbildung 4: Die Darstellung zeigt, dass die Beleuchtung zweier Punkte x_1 und x_2 einer Oberfläche S mit den Normalen $\nu(x_1)$ und $\nu(x_2)$ durch eine Umgebungsbeleuchtung f nur abhängig von der Verdeckung der Punkte und nicht von deren Position ist. Bei x_1 und x_2 sind in diesem Falle die gleichen Lichtintensitäten zu messen. Es ist $\lambda \in (0,\infty)$ die Wellenlänge des Lichtes und $\omega \in \mathcal{S}^2$ der Raumwinkel.

Aber auch die Materialien der Szene sollten in der Lage sein Licht auszusenden. Wir führen hierfür eine Abbildung ein, die die dafür nötigen Eigenschaften erfüllt. Häufig werden diese Lichtquellen auch »Feldlichtquellen« (engl.: *area light*) genannt [18, S. 733 ff].

DEFINITION 1.7: (Emission)

Sei $\mathcal{T}$ *eine Mesh mit einer Shading-Normalen* ν*. Dann ist eine Emission von* $\mathcal{T}$ *durch eine integrierbare Abbildung* $E\colon \mathcal{T}\times(0,\infty)\times\mathcal{S}^2 \to [0,\infty)$ *gegeben.*

Wie bei der vorherigen Defintion beschreibt diese Funktion das ausgesandte Licht in Abhängigkeit der Wellenlänge und des Raumwinkels. Der Unterschied besteht darin, dass sie auf der Oberfläche einer Mesh variieren kann und sich damit auch die Beleuchtung eines Punktes je nach Position verändert.

Durch die Einführung der Lichtquellen erhalten wir nun die vollständige Beschreibung einer Szene durch Zusammenführung der bereits definierten Strukturen.

DEFINITION 1.8: (Szene)

Eine Szene ist ein Tupel $(\mathcal{T}, \nu, f, E, U)$ *bestehend aus einer Mesh* $\mathcal{T}$ *mit einer Shading-Normalen* ν*, einer integrierbaren Abbildung* $f\colon \mathcal{T} \times \mathbb{S}^2 \times \mathbb{S}^2 \to [0, \infty)$*, wobei* $f(x, \cdot, \cdot)$ *für* σ*-fast-alle* $x \in \mathcal{T}$ *ein BSDF bezüglich* $\nu(x)$ *darstellt, einer Emission* E *von* $\mathcal{T}$ *und einer Umgebungsbeleuchtung* U*.*

1.4 Raytracing

Im einfachsten Falle bezeichnet das Wort »Raytracing« (engl.: *ray tracing*) einen Algorithmus zur Ermittlung der Sichtbarkeit von dreidimensionalen Objekten bezüglich eines Ursprungpunktes (engl.: *origin*) im Raum [16, 18]. Häufig versteht man darunter jedoch auch eine Render-Technik für die Generierung eines gesamten Bildes aus einer gegebenen Szene, die auf dem eben genannten Raytracing-Algorithmus basiert [15, 16, 18].

Grundsätzlich gibt es viele Verfahren, um ein Szene auf ein Bild zu rendern [1, 5]. Die Erfahrung zeigt aber, dass vor allem in Bereichen, in denen die globalen Beleuchtungseffekte realistisch simuliert werden sollen, Raytracing eine wichtige Grundlage darstellt. Der Grund dafür besteht in der Tatsache, dass Raytracing das »Sichtbarkeitsproblem« (engl.: *hidden surface removal*, *visible surface determination* oder *occlusion culling*, [5, 6]) löst und durch die Verwendung von Strahlen eine Basis für die Lichtberechnung im Sinne der geometrischen Optik bereitstellt [16, 18, 20].

DEFINITION 1.9: (Sichtbarkeitsproblem)

Sei $\mathcal{S}$ *eine Szene. Dann ist die Sichtbarkeitsfunktion von* $\mathcal{S}$ *die folgende Abbildung.*

$$V_{\mathcal{S}}\colon \mathbb{R}^3 \times S(\mathcal{S}) \to \{0, 1\}$$

$$V_{\mathcal{S}}(o, x) = \begin{cases} 1 & : S(\mathcal{S}) \cap \{(1-\gamma)o + \gamma x \mid \gamma \in (0,1)\} = \emptyset \\ 0 & : \text{sonst} \end{cases}$$

Das Sichtbarkeitsproblem beschreibt dann die Aufgabe diese Funktion für gegebene Szenen und Parameter zu evaluieren.

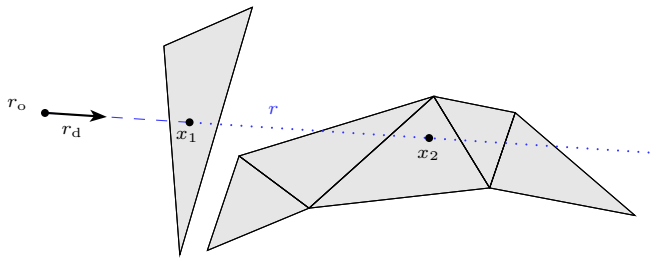

Abbildung 5: Die Abbildung zeigt eine Skizze, welche das Sichtbarkeitsproblem und den Raytracing-Algorithmus verdeutlicht. Die grau melierten Dreiecke sollen die gegebene Szene $\mathcal{S}$ darstellen. Der ausgesendete Strahl $r = (r_\mathrm{o}, r_\mathrm{d})$ trifft in der Szene genau zwei Punkte x_1 und x_2. Dabei wird x_2 durch x_1 verdeckt. Es gilt also $V_{\mathcal{S}}(r_\mathrm{o}, x_1) = 1$ und $V_{\mathcal{S}}(r_\mathrm{o}, x_2) = 0$.

Die Sichtbarkeitsfunktion gibt an, ob der Oberflächenpunkt x vom Beobachtungspunkt o aus in gerader Linie gesehen werden kann oder ob zwischen diesen Punkten ein Primitiv den Punkt x verdeckt. Daran anknüpfend besteht die Basis des Raytracing-Verfahrens auf dem Aussenden von »Strahlen« (engl.: *ray*) bezüglich eines Ursprungspunktes. Für diese Strahlen kann der Schnittpunkt mit der Szene ermittelt werden. Liegt der Schnittpunkt zwischen o und x, so beträgt der Wert der Sichtbarkeitsfunktion 0 ansonsten 1. Die Sichtbarkeitsfunktion kann somit für alle gegebenen Parameter durch Raytracing berechnet werden. Abbildung 5 zeigt diese Methode anhand einer Skizze.

DEFINITION 1.10: (Strahl)

Ein Strahl r sei gerade durch ein Tupel $(r_\mathrm{o}, r_\mathrm{d})$ mit einem Ursprungspunkt $r_\mathrm{o} \in \mathbb{R}^3$ und einer Richtung $r_\mathrm{d} \in \mathbb{R}^3 \setminus \{0\}$ charakterisiert. Die Menge $S(r)$ der Punkte des Strahls ist gegeben durch das Bild der Parametrisierung $([0,\infty), \varphi_r)$

$$S(r) \coloneqq \operatorname{im} \varphi_r, \qquad \varphi_r \colon [0,\infty) \to \mathbb{R}^3, \qquad \varphi_r(t) \coloneqq r_\mathrm{o} + t r_\mathrm{d}$$

Die Menge aller Strahlen sei mit R bezeichnet.

Die vollständige Evaluierung von $V_\mathcal{S}$ ist jedoch häufig nicht nötig. In Abbildung 5 ist klar, dass alle Punkte von r, die hinter x_1 liegen von dem Beobachtungspunkt r_o aus nicht sichtbar sind. Beim eigentlichen Raytracing-Algorithmus macht man sich dies zu Nutze und berechnet für eine vorgegebene Richtung nur den nächsten Schnittpunkt. Alle weiteren Schnittpunkte können ignoriert werden, da sie vom Strahl aus nicht erreicht werden.

DEFINITION 1.11: (Raytracing-Funktion)

Sei $\mathcal{S}$ eine Szene. Dann ist die Raytracing-Funktion der Szene $\mathcal{S}$ gegeben durch die folgende Abbildung.

$$\mathrm{rt}_\mathcal{S} \colon R \to (0,\infty]$$

$$\mathrm{rt}_\mathcal{S}(r) \coloneqq \begin{cases} \min\{t \in (0,\infty) \mid r(t) \in S(\mathcal{S})\} & : S(\mathcal{S}) \cap \varphi_r((0,\infty)) \neq \emptyset \\ \infty & : \text{sonst} \end{cases}$$

Wie bereits erwähnt, erweitert man diesen Algorithmus häufig mit der Berechnung eines gesamten Bildes, indem man für jeden Pixel des Bildes einen oder mehrere Strahlen durch einen analogen virtuellen Pixel im Szenenraum schießt und die Raytracing-Funktion für diese evaluiert. Abbildung 6 zeigt dieses Verfahren anhand einer Skizze. Um gleichzeitig das Shading zu ermöglichen, ermittelt man nicht nur den Schnittpunkt, sondern auch in welchem Primitiv sich dieser befindet und welche baryzentrischen Koordinaten er besitzt. Die Implementierung des Raytracing-Verfahrens soll hier nicht gezeigt werden, da eine einfache naive Implementierung meistens einen zu großen Rechenaufwand darstellt und die hier verwendete optimierte Variante weit über das Thema dieser Arbeit hinausgeht.

1.5 Radiometrie

Ein physikalischer Körper K mit der Oberfläche ∂K und äußerer Normalen-Funktion μ sendet aufgrund verschiedener physikalischer Effekte, wie zum Beispiel Reflexion oder Emission, elektromagnetische Wellen aus. Für die Simulation von Beleuchtungseffekten ist es nötig die Abhängigkeit

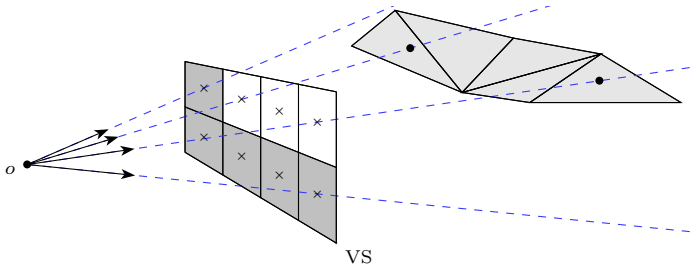

Abbildung 6: In der Skizze ist ein typisches Render-Verfahren auf der Basis des Raytracing-Algorithmus dargestellt. Bezüglich eines Beobachtungspunktes o wird durch jeden Pixel eines virtuellen Bildschirms VS ein Strahl geschossen und dessen Raytracing-Funktion evaluiert. Gibt es keinen Schnittpunkt, so wird der Pixel meistens schwarz gefärbt. Ansonsten kann zum Beispiel die Farbe des getroffenen Objektes angezeigt werden. Der Übersicht wegen sind im Bild nur vier der eigentlich acht Strahlen sichtbar.

dieser Abstrahlung für verschiedene Wellenlängen $\lambda \in (0,\infty)$, verschiedene Oberflächenpunkte $x \in \partial K$ und verschiedene Richtungen $\omega \in \Omega$ zu betrachten. Dafür führen wir die sogenannte »spektrale Strahldichte« (engl.: *spectral radiance*) ein. Sie gibt an, wie viel Energie pro Zeit, pro Wellenlängenänderung, pro Flächenelement und pro Raumwinkel abgestrahlt beziehungsweise empfangen wird. Sind also messbare Teilmengen $\Lambda \subset (0,\infty)$, $U \subset \partial K$ und $S \subset \Omega$ gegeben, so kann man die spektrale Strahldichte über die abgegebene Strahlungsleistung $\Phi(\Lambda, U, S)$ der Oberfläche U in den Raumwinkelbereich S definieren. Die zugehörige Abbildung ist dann wie folgt gegeben.

$$\mathcal{L}\colon (0,\infty) \times \partial K \times \Omega \to [0,\infty)$$

$$\Phi(\Lambda, U, S) =: \int_\Lambda \int_U \int_S \mathcal{L}(\lambda, x, \omega)\, |\langle \mu(x), \omega \rangle|\ \mathrm{d}\omega\, \mathrm{d}\sigma(x)\, \mathrm{d}\lambda$$

Abbildung 7 zeigt eine Skizze, welche die Struktur dieser Abbildung verdeutlicht. Der Erfahrung nach stellt die spektrale Strahldichte für das menschliche Auge die sinnvollste messbare Größe für die empfundene Helligkeit einer Oberfläche bezüglich eines Beobachtungspunktes dar.

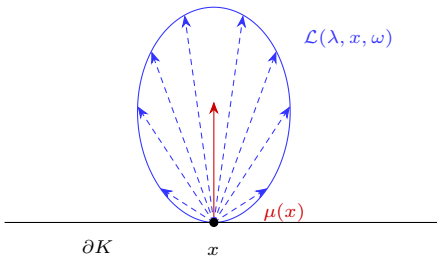

Abbildung 7: Die Skizze stellt ein qualitatives Beispiel einer spektralen Strahldichte $\mathcal{L}$ dar, die an einem Punkt x auf der Oberfläche ∂K und für eine Wellenlänge λ eine Verteilung in Abhängigkeit von ω auf der Einheitskugel besitzt. In der Abbildung ist dabei die Abstrahlung nur erlaubt, wenn $\langle \omega, \mu(x) \rangle > 0$, wobei $\mu(x)$ die äußere Normale im Punkt x beschreibt.

Eine weitere Größe, die wir hier einführen möchten, ist die sogenannte spektrale »Bestrahlungsstärke« oder auch spektrale Irradianz (engl.: *spectral irradiance*). Sie wird vor allem bei der noch folgenden Konstruktion der Irradiance Maps benötigt werden. Wir definieren sie für eine messbare Teilmenge $S \subset \Omega$.

$$\mathcal{E}\colon (0,\infty) \times \partial K \to [0,\infty), \qquad \mathcal{E}(\lambda, x) \coloneqq \int_S \mathcal{L}(\lambda, x, \omega)\, |\langle \mu(x), \omega \rangle|\ \mathrm{d}\sigma(\omega)$$

Die spektrale Bestrahlungsstärke quantifiziert die Energie pro Zeit, pro Wellenlängenänderung und pro Flächenelement, die an einem Punkt auf der Oberfläche des Objektes aus dem Raumwinkelbereich S empfangen wird.

Für die weiteren Algorithmen und Implementierungen reicht es entkoppelte Abbildungen zu betrachten, welche für jeden Punkt einer Szene und jede gegebene Richtung der Strahldichte beziehungsweise der Irradianz entsprechen. Später werden dies die Funktionen sein, die wir durch »Path Tracing« simulieren und durch geeignetes »Surface Caching« zwischenspeichern wollen.

DEFINITION 1.12: (Strahldichte- und Irradianz-Funktion)

Sei $\mathcal{S}$ eine Szene. Dann ist eine Strahldichte-Funktion von $\mathcal{S}$ gegeben durch eine integrierbare Abbildung

$$L_{\mathcal{S}} \colon (0,\infty) \times S(\mathcal{S}) \times \Omega \to [0,\infty)$$

Die zu $L_{\mathcal{S}}$ gehörige Irradianz-Funktion ist definiert durch

$$E_{L_{\mathcal{S}}} \colon (0,\infty) \times S(\mathcal{S}) \to [0,\infty), \qquad E_{L_{\mathcal{S}}}(\lambda, x) \coloneqq \int_{\Theta_{\mu(x)}} L_{\mathcal{S}}(\lambda, x, \omega)\, \langle \mu(x), \omega \rangle \ \mathrm{d}\sigma(\omega)$$

Ist aus dem Kontext klar, um welche Strahldichte-Funktion es sich handelt, schreiben wir auch $E_{\mathcal{S}}$.

1.6 Rendergleichung

Die »Rendergleichung« (engl.: *rendering equation* oder *light transport equation*) ist eine Integralgleichung, die 1986 von James T. Kajiya entwickelt wurde, um die bis zu diesem Zeitpunkt verbeiteten Rendertechniken für die Simulation globaler Beleuchtungseffekte auf eine gemeinsame Basis zu stellen. Sie selbst kann mithilfe der Prinzipien der geometrischen Optik und dem Energieerhaltungssatz hergeleitet werden. Insbesondere handelt es sich um eine Fredholm-Integralgleichung 2.Art, die eine eindeutige Lösung besitzt.

DEFINITION 1.13: (Rendergleichung)

Sei $\mathcal{S}$ eine gegebene Szene mit Strahldichte-Funktion $L_{\mathcal{S}}$. Dann gehorcht $L_{\mathcal{S}}$ der Rendergleichung, wenn für alle $x \in S(\mathcal{S})$ und $\omega_0 \in \Omega$ gilt

$$\begin{aligned} L_{\mathcal{S}}(x, \omega_0) &= E_{\mathcal{S}}(x, \omega_0) \\ &\quad + \int_{S(\mathcal{S})} V_{\mathcal{S}}(x,y) G_{\mathcal{S}}(x,y) f_{\mathcal{S}}(x, \omega(x,y), \omega_0) L_{\mathcal{S}}(y, \omega(x,y))\ \mathrm{d}\sigma(y) \end{aligned}$$

Der Übersicht halber wurden dabei die folgenden Symbole verwendet.

$$G_{\mathcal{S}}(x,y) \coloneqq \frac{|\langle \nu(x), \omega(x,y) \rangle|\, |\langle \nu(y), \omega(x,y) \rangle|}{\|x-y\|^2} \qquad \omega(x,y) \coloneqq \frac{y-x}{\|y-x\|}$$

Aufgrund der komplexen Struktur verwenden wir auch eine weitere Variante.

$$L_{\mathcal{S}}(x, \omega_0) = E_{\mathcal{S}}(x, \omega_0) + \int_{\Omega} f_{\mathcal{S}}(x, \omega, \omega_0) \tilde{L}_{\mathcal{S}}(x, \omega)\, |\langle \nu(x), \omega \rangle| \ \mathrm{d}\sigma(\omega)$$

DEFINITION 1.14:

$$L_{\mathcal{S}}(\lambda, x, \omega_0) = E_{\mathcal{S}}(\lambda, x, \omega_0) + \int_{\Omega} f_{\mathcal{S}}(x, \omega, \omega_0) \tilde{L}_{\mathcal{S}}(\lambda, x, \omega) \, |\langle \nu(x), \omega \rangle| \; \mathrm{d}\sigma(\omega)$$

$$\tilde{L}_{\mathcal{S}}(\lambda, x, \omega) \coloneqq \begin{cases} L_{\mathcal{S}}(\lambda, \varphi_r(\mathrm{rt}_{\mathcal{S}}(r)), -\omega) & : \mathrm{rt}(r) < \infty \\ \mathrm{HDR}_{\mathcal{S}}(\lambda, \omega) & : \text{sonst} \end{cases}$$

Gehorcht eine Strahldichte-Funktion der Szene $\mathcal{S}$, so simuliert sie die globalen Beleuchtungseffekte dieser Szene auf einer physikalischen Basis, wodurch die entstehenden Lichtverhältnisse für das menschliche Auge unheimlich real wirken.

BILD EINBINDEN FÜR DIE LÖSUNG

(KEINE EXTRA SECTION FÜR PATH TRACING) Um die Gleichung zu lösen gibt es verschiedene Verfahren, wie zum Beispiel »Path Tracing«, »Bidirectional Path Tracing«, »Metropolis Light Transport« oder auch »Photon Mapping«. Im Grunde genommen, kann jeder Algorithmus verwendet werden. Für diese Arbeit wurde Path Tracing verwendet.

1.7 Surface Caching

Der Begriff »Surface Caching« (deutsch: *Oberflächenpuffer*) wurde vor allem durch das Computerspiel »Quake« geprägt. Es war das erste Spiel welches sogenannte »Surface Caches« und Lightmaps verwendete, um so die grafischen Berechnungen während der Laufzeit auf ein Minimum zu beschränken. Der Begriff selbst ist nicht standardisiert und wird meistens verwendet um Algorithmen, die in der »Quake-Engine« vorkommen, genauer zu erläutern. Wir wollen diesen Begriff für eine Oberklasse von Algorithmen und Datenstrukturen verwenden, die es entweder ermöglichen Informationen auf Oberflächen einer Szene zu speichern oder diese Informationen aus einem geeigneten Oberflächenpuffer wieder auszulesen. Insbesondere wollen wir uns mit dem Speichern von Strahldichten diffuser Lichtverteilungen beschäftigen, bei denen die Strahldichte-Funktion L nur vom Ort der Szene und nicht mehr vom betrachteten Raumwinkel abhängt.

Zu beachten ist, dass die hier genannten Verfahren im Allgemeinen nur für statische Szenen Sinn ergeben, da eine Veränderung der Geometrie und Lichtverhältnisse dazu führt, dass der Surface Cache keine aktuellen Informationen mehr enthält. Er müsste demnach bei jeder Änderung der Szene neu berechnet werden. Da die betrachteten Szenen teilweise sehr groß sein können, wäre dieser Schritt ineffizient. Zwei typische Verfahren zum zwischenspeichern von Strahldichten werden in den beiden folgenden Unterabschnitten genauer erläutert.

1.7.1 Vertex Lighting

Bei der Implementierung einer Szene $\mathcal{S}$ haben wir Vertices eingeführt, um die Speicherkosten von $\mathcal{S}$ zu verringern. Dabei existiert jeder Vertex unabhängig von allen Primitiven. Nur die Primitive besitzen die Informationen, aus welchen Vertices sie selbst bestehen. Dies führt zu der Idee, die diffuse Strahldichte-Funktion $L_{\mathcal{S}}$ an den Positionen aller Vertices zu evaluieren und zu speichern. Möchte man dann die Szene rendern, so kann man für jeden Strahl, der ein Schnittpunkt x mit der Szene hat, das getroffene Primitiv p und die zugehörigen baryzentrischen Koordinaten (u, v, w) des Schnittpunktes bestimmen. Die Strahldichte an dieser Position wird dann durch die folgende Gleichung interpoliert.

$$L_{\mathcal{S}}(x, \cdot) \approx w L_{\mathcal{S}}^{(0)} g + u L_{\mathcal{S}}^{(1)} + v L_{\mathcal{S}}^{(2)}$$

Es handelt sich hierbei um Gourad-Shading oder auch bilineare Interpolation auf dem Dreieck.

1.7.2 Lightmap

Lightmaps sind Datenstrukturen, welche vorberechnete Helligkeitswerte für Oberflächen einer Szene in einer Textur speichern. Damit sind sie vor allem für Anwendungen, die sich extrem auf die Grafikkarte des Computers stützen, geeignet, da Grafikkarten speziell eingebaute Hardware besitzen, um Texturen zu verarbeiten. Der gesamte Prozess der Generierung und Nutzung einer solchen Datenstruktur wird auch »Lightmapping« genannt. Abbildung 8[1] zeigt ein einfaches Beispiel.

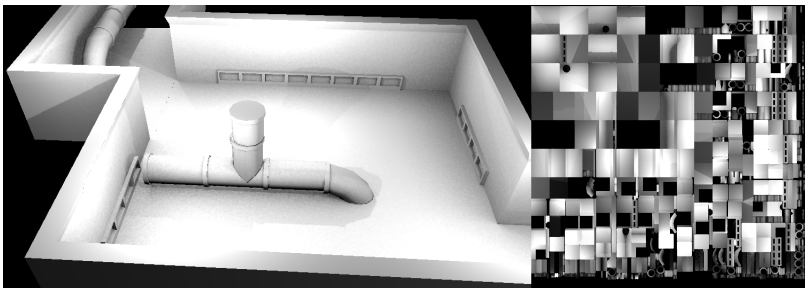

Abbildung 8: Die rechte Seite der Abbildung zeigt ein typisches Beispiel einer Lightmap-Textur. Mit ihrer Hilfe kann im linken Bereich des Bildes die Szene »gelightmapped« werden. Beim eigentlichen Rendern der Szene sind nun keine Sichtbarkeitstests mehr nötig, um Schatten zu generieren.

Lightmaps werden im Allgemeinen manuell optimiert, um den speziellen Gegebenheiten einer Szene gerecht zu werden. Dies erfordert eine Menge zusätzlichen Zeit- und Arbeitsaufwand, sobald eine neue Szene betrachtet werden soll. Lightmaps besitzen außerdem auch alle Nachteile von Texturen. Die Größe der Speicher heutiger Grafikkarten ist im Vergleich zu den normalen Arbeitsspeichern eher gering. Damit ist natürlich auch die Größe der Lightmap beschränkt. Ein weiterer Nachteil ist, dass sich eine Textur, die quadratische Pixel besitzt, niemals vollständig an ein Dreieck in der Szene anpassen kann. Die Folge ist, dass bei zu dicht gelegenen Helligkeitswerten auf der Lightmap, die zu verschiedenen Dreiecken gehören, Interpolations-Artefakte auftreten.

[1] Quelle: `https://upload.wikimedia.org/wikipedia/commons/1/15/Lightmapped_Scene_with_Lightmap.png`
Donnerstag 27.April 17.20 Uhr

$$\mathcal{H}^2_\mu, \mathcal{S}^2, \mathcal{H}^2_\mu, \mathbb{S}^2, \mathcal{A}$$

Diese Definition zeigt diverse Probleme auf, die bei der Modellierung von Objekten entstehen können. Dreiecke der Primitive in einer Szene dürfen sich im Allgemeinen beliebig schneiden. Dadurch ist es im Allgemeinen nicht möglich jedem Oberflächenpunkt eine eindeutige äußere Normale zuzuordnen. Bei der Generierung der Irradiance Maps werden sich diese Punkte als Fehler äußern.

Die abstrakte Definition ist für die Implementierung leider noch nicht ausreichend.

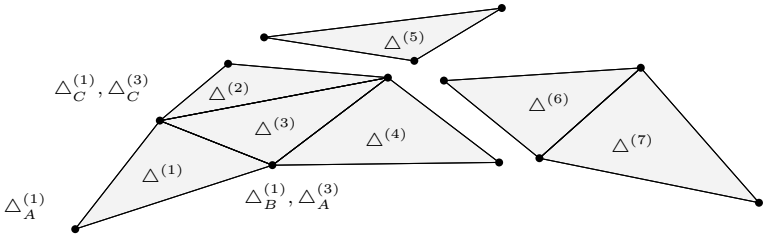

Abbildung 9: Die Abbildung zeigt eine beispielhafte Menge von Dreiecken $\left\{\triangle^{(i)} \;\middle|\; i \in \mathbb{N}, i \leq 7\right\}$. Verschiedene Gruppen von Dreiecken bilden eine Approximation einer Oberfläche im Raum. Dabei werden die Eckpunkte der Dreiecke geteilt und es gilt zum Beispiel $\triangle_B^{(1)} = \triangle_A^{(3)}$ und $\triangle_C^{(1)} = \triangle_C^{(3)}$.

Aufgrund der Definition von physikalischer BRDFs muss bei Linearkombination eine weitere Bedingung beachtet werden. Das folgende Theorem klärt dies genauer.

THEOREM 1.1: (Linearkombination von BRDFs bildet wieder eine BRDF)

Seien $\triangle$ ein Dreieck und ν eine äußere Normalen-Funktion auf $\triangle$. Seien weiterhin $n \in \mathbb{N}$ und für alle $i \in \mathbb{N}$ mit $i \leq n$ ideale BRDFs f_i und Koeffizienten $\alpha_i \in [0,1]$ mit $\sum_{i=1}^n \alpha_i \leq 1$ gegeben. Dann ist die folgende Funktion wieder eine physikalische BRDF.

$$f \coloneqq \sum_{i=1}^{n} \alpha_i f_i$$

Literatur

[1] Akenine-Möller, Tomas, Eric Haines, and Naty Hoffman: *Real-Time Rendering*. A K Peters, Ltd., third edition, 2008.

[2] Botsch, Mario, Mark Pauly, and C Rossl: *Geometric modeling based on triangle meshes*. Acm Siggraph 2006 ..., pages 1–148, 2006. `http://dl.acm.org/citation.cfm?id=1185839`.

[3] Bronstein, Semendjajew, Musiol und Mühlig: *Taschenbuch der Mathematik*. Europa-Lehrmittel, zehnte Auflage, 2016, ISBN 978-3-8085-5789-1.

[4] Cohen, Michael F. and John R. Wallace: *Radiosity and Realistic Image Synthesis*. Academic Press Professional, 1995, ISBN 0-12-059756-X.

[5] Cohen-Or, Daniel, Yiorgos L. Chrysanthou, Cláudio T. Silva, and Frédo Durand: *A survey of visibility for walkthrough applications*. IEEE Transactions on Visualization and Computer Graphics, 9(3):412–431, 2003, ISSN 10772626.

[6] Durand, F: *3D Visibility: analytical study and applications*. Université Joseph Fourier, Grenoble, France, 1999. http://scholar.google.com/scholar?hl=en{\&}btnG=Search{\&}q=intitle:3D+Visibility+:+Analytical+Study+and+Applications{\#}0{\%}5Cnhttp://scholar.google.com/scholar?hl=en{\&}btnG=Search{\&}q=intitle:3D+Visibility:+analytical+study+and+applications{\#}0.

[7] Elstrodt, Jürgen: *Maß- und Integrationstheorie*. Springer, siebte korrigierte und aktualisierte Auflage, 2011, ISBN 978-3-642-17904-4.

[8] Jensen, Henrik Wann: *A practical guide to global illumination using photon maps*. SIGGRAPH 2000 Course Notes CD-, 38:20–es, 2000. http://portal.acm.org/citation.cfm?doid=1103900.1103920{\%}5Cnhttp://171.67.77.70/courses/cs348b-00/course8.pdf.

[9] Kajiya, James T: *The rendering equation*. In *ACM Siggraph Computer Graphics*, volume 20, pages 143–150. ACM, 1986.

[10] Kühnel, Wolfgang: *Differentialgeometrie: Kurven - Flächen - Mannigfaltigkeiten*. Springer Spektrum, sechste Auflage, 2013, ISBN 978-3-658-00614-3.

[11] LaMothe, André: *Tricks of the 3D Game Progamming Gurus: Advanced 3D Graphics and Rasterization*. Sams Publishing, 2003.

[12] Levoy, Marc and Pat Hanrahan: *Light field rendering*. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96, pages 31–42, 1996, ISSN 00978930. http://portal.acm.org/citation.cfm?doid=237170.237199.

[13] Morvan, Jean Marie and Boris Thibert: *Smooth surface and triangular mesh: comparison of the area, the normals and the unfolding*. SMA '02: Proceedings of the seventh ACM symposium on Solid modeling and applications, pages 147–158, 2002.

[14] Nicodemus, Fe, Jc Richmond, and Jj Hsia: *Geometrical considerations and nomenclature for reflectance*. Science And Technology, 60(October):1–52, 1977, ISSN 10411135. http://graphics.stanford.edu/courses/cs448-05-winter/papers/nicodemus-brdf-nist.pdf.

[15] Nikodym, Tomas: *Ray Tracing Algorithm For Interactive Applications Tomas Nikodym*. page 76, 2010.

[16] Parker, Steven, William Martin, Peter Pike J Sloan, Peter Shirley, Brian Smits, and Charles Hansen: *Interactive ray tracing*. Proceedings of the 1999 symposium on Interactive 3D graphics SI3D 99, pages:119–126, 1999. http://portal.acm.org/citation.cfm?doid=300523.300537.

[17] Pharr, M. and G. Humphreys: *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, second edition, 2010.

[18] Pharr, M., W. Jakob, and G. Humphreys: *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, third edition, 2017.

[19] Shirley, P., B. Wade, P. M. Hubbard, D. Zareski, B. Walter, and D. P. Greenberg: *Global Illumination via Density-Estimation*. Proceedings of 6th Workshop on Rendering, pages 219–230, 1995. http://link.springer.com/chapter/10.1007/978-3-7091-9430-0{_}21{\%}5Cnhttp://www.cs.utah.edu/{~}shirley/papers/de95.pdf.

[20] Veach, Eric: *Robust Monte Carlo Methods for Light Transport Simulation*. Dissertation at the Department of Computer Science of Stanford University, 134(December):759–764, 1997, ISSN 13509462. http://graphics.stanford.edu/papers/veach{_}thesis/.

[21] Veach, Eric and Leonidas J. Guibas: *Metropolis light transport*. Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97, pages 65–76, 1997, ISSN 00978930. http://portal.acm.org/citation.cfm?doid=258734.258775.

[22] Walter, Bruce, Philip M Hubbard und Peter Shirley: *Global Illumination Using Local Linear Density Estimation*.

[23] Ward, Gregory J. und Paul S Heckbert: *Irradiance gradients*. ACM SIGGRAPH 2008 classes on - SIGGRAPH '08, Seite 1, 2008. http://portal.acm.org/citation.cfm?doid=1401132.1401225.

[24] Yuksel, Cem, John Keyser und Donald H. House: *Mesh colors*. ACM Transactions on Graphics, 29(2):1–11, 2010, ISSN 07300301.