Universität Leipzig
Institut für Informatik
Abteilung Bild- und Signalverarbeitung

Computergrafik-Praktikum
Sommersemester 2013
Betreuer: Steven Schlegel (schlegel@informatik.uni-leipzig.de)

Aufgabenblatt 2 – Abgabe in der Woche vom 10. Juni

Farbberechnung im Raytracer

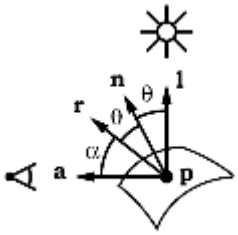

Abbildung 1: Winkel zur Auswertung des Beleuchtungsmodells.

Aufgabe 1: Implementieren Sie das Beleuchtungsmodell von Phong

Wurde ein gültiger Schnittpunkt **p** des Sichtstrahles mit der Szene ermittelt, so ist nun die Farbe des entsprechenden Pixels zu berechnen, unter Berücksichtigung der Materialeigenschaften und der Beleuchtungssituation. Die Farbe wird mit Hilfe eines Beleuchtungsmodells wie folgt ermittelt.

Zunächst benötigen wir die Flächennormale **n** im Punkt **p**. Um den Eindruck einer glatten Fläche zu erwecken, verwenden wir *Phong Shading*, d.h.

$$\mathbf{n} = \alpha_1 \mathbf{n}_1 + \alpha_2 \mathbf{n}_2 + \alpha_3 \mathbf{n}_3,$$

wobei $\mathbf{n}_i$ die Normalen der Eckpunkte $\mathbf{p}_i$ des Dreiecks und α_i die baryzentrischen Koordinaten von p bezeichnen (siehe letztes Aufgabenblatt).

Sei **a** der Vektor zum Augpunkt (umgekehrte Blickrichtung). Falls der Winkel zwischen **a** und **n** mehr als 90 Grad beträgt, so ist die Orientierung von **n** umzukehren. Angenommen, es existiert nur eine einzige (punktförmige) Lichtquelle, die vom Punkt **p** aus in Richtung des Vektors **l** sichtbar ist. Dann würde gespiegeltes Licht entlang der Richtung **r**, wie in Abbildung 4 dargestellt, reflektiert,

$$\mathbf{r} = 2(\mathbf{n} \cdot \mathbf{l})\,\mathbf{n} - \mathbf{l}$$

Hierbei wurde vorausgesetzt, daß die Vektoren **a**, **n** und **l** normalisiert sind. Die entscheidenden Größen für das Beleuchtungsmodell sind die Winkel θ (zwischen **n** und **l**) und α (zwischen **a** und **r**).

Die zu berechnende Farbintensität setzt sich aus drei Termen zusammen:

- Die *ambiente* Intensität ist ein (in der Regel schwacher) Anteil des Lichtes, das von anderen Flächenkomponenten diffus reflektiert wird und im Punkt **p** eintrifft. Der Einfachheit halber wird dieser Term als konstant (unabhängig von **p**) angenommen. (Eine genaue Berechnung dieser Intensität ist mit dem *Radiosity* Verfahren möglich.)
- Als *diffus* bezeichnen wir das Licht, welches von einer oder mehreren Lichtquellen ausgehend am Punkt **p** eintrifft und von dort (wie das ambiente Licht) gleichmäßig in alle Richtungen abgestrahlt wird. Die diffuse Intensität hängt daher nicht von der Position des Augpunktes, wohl aber von der Beleuchtungssituation ab.
- *Spiegelndes* Licht ist genau dann anzutreffen, wenn sich, vom Augpunkt gesehen, eine Lichtquelle in **p** spiegelt. In diesem Fall ergibt sich in der Umgebung von **p** ein (meistens sehr intensives)

"Highlight", dessen Größe auch von der Glänzigkeit (Shininess) des Materials abhängt.

Diese drei Terme sind in dem Beleuchtungsmodell von Phong zusammengefaßt, das die Intensität für eine bestimmte Wellenlänge l liefert:

$$I_\lambda = I_{a\lambda} k_a O_{d\lambda} + f_{att}(d_l)\,(I_{d\lambda} k_d O_{d\lambda} \cos\theta + I_{s\lambda} w(\theta) \cos^S\alpha)$$

Das Beleuchtungsmodell kann z.B. für rote, grüne und blaue Farbanteile berechnet werden, d.h. $\lambda \in \{\lambda_R, \lambda_G, \lambda_B\}$. Die einzelnen Konstanten und Funktionen haben folgende Bedeutung:

- $I_{a\lambda}$ beschreibt die Intensität des ambienten Lichtes (konstant für jede einzelne Wellenlänge)
- $I_{d\lambda}$ ist die diffuse Intensität der Lichtquelle (für jede Wellenlänge)
- $I_{s\lambda}$ ist die speculare Intensität der Lichtquelle (für jede Wellenlänge)
- $O_{d\lambda}$ ist die Materialfarbe (Intensität diffus reflektierten Lichtes für jede Wellenlänge)
- k_a und k_d sind zusätzliche Materialkonstanten (wellenlängenunabhängig) zur Gesamteinstellung der ambienten und diffusen Intensität
- $f_{att}(d_l)$ ist eine Funktion der Entfernung d_l der Lichtquelle zum Punkt **p**, zum Modellieren der Abschwächung (Attenuation) während der Ausbreitung des Lichtes
- $w(\theta)$ ist eine materialabhängige Funktion
- s ist eine Konstante für die Glänzigkeit des Materials. Der Term $\cos^S\alpha$ modelliert das Abfallen des spiegelnden Highlights entlang der Fläche und hat empirischen Charakter (keine theoretische Grundlage).

Die Abschwächung des Lichtes f_{att} ist in der Natur quadratisch proportional zum Ausbreitungsweg d, d.h. $f_{att}(d) = 1 / d^2$. Dies führt jedoch beim Rendering nicht immer zu zufriedenstellenden Resultaten. Wir verwenden daher eine allgemeinere Form dieser Funktion,

$$f_{att}(d) = 1 / (c_0 + c_1 d + c_2 d^2)$$

Die Konstanten c_i befinden sich in der Definitionsdatei und erlauben es, quadratische, lineare, und konstante Abschwächung oder jede gewünschte Kombination zu modellieren. Im Quellcode sind das Member der Struktur *Lightsource*: constAtt, linAtt, quadAtt. Der Fall $c_0 = c_1 = c_2 = 0$ ist gesondert zu betrachten. In dem Fall sollte $f_{att}(d) = 1$ gesetzt werden.

Das Beleuchtungsmodell für den Praktikumsraytracer hat noch einige (kleine) Besonderheiten: die Konstanten k_a und k_d, sowie die Funktion $w(\theta)$ werden durch Eins ersetzt. Statt dessen sind die ambienten, diffusen, und spiegelnden RGB-Farbanteile des Materials voneinander unabhängig und werden mit $O_{a\lambda}$, $O_{d\lambda}$ und $O_{s\lambda}$ bezeichnet. Die diffusen und spiegelnden Terme sind für jede vom Punkt **p** aus sichtbare Lichtquelle l aufzuaddieren. Das zu implementierende Beleuchtungsmodell hat also folgende Form:

$$I = I_{a\lambda} O_{a\lambda} + \sum_{l\ sichtbar} f_{att}(d_l)(I_{d\lambda} O_{d\lambda} (\mathbf{n} \cdot \mathbf{l}) + I_{s\lambda} O_{s\lambda} (max\ \{0, \mathbf{r}\cdot\mathbf{a}\})^s)$$

Hier wurde bereits $\mathbf{n}\cdot\mathbf{l}$ für $\cos\theta$ und $\mathbf{r}\cdot\mathbf{a}$ für $\cos\alpha$ eingesetzt. Der spiegelnde Term verschwindet für Winkel $\geq 90°$. Eine Lichtquelle l ist vom Flächenpunkt **p** (mit Normale **n**) aus sichtbar, wenn $\theta < 90°$ und wenn das Geradensegment von **p** bis zur Position der Lichtquelle keine anderen Objekte (Dreiecke) schneidet.

Im Quellcode sind folgende Member wichtig:

Struktur *Lightsource:* *position* – Position der Lichtquelle
ambient, diffuse, specular – die entsprechenden Lichtintensitäten $I_{a\lambda}$, $I_{d\lambda}$ und $I_{s\lambda}$
constAtt, linAtt, quadAtt – c_0, c_1 und c_2

Struktur *Material:* *ambient, diffuse, specular* – die entsprechenden Materialeigenschaften $O_{a\lambda}$, $O_{d\lambda}$ und $O_{s\lambda}$

Schatten

Bevor das Beleuchtungsmodell ausgewertet wird, wird überprüft, ob der Punkt **p** bezüglich der Lichtquelle l eventuell im Schatten liegt. Dazu wird ein Strahl von **p** in Richtung l geschickt. Falls dieser Strahl ein Dreieck auf dem Weg zur Lichtquelle schneidet, so liegt **p** bezüglich l im Schatten. Dann wird nur der ambiente Term des Phong-Modells ausgewertet. Der diffuse und speculare Anteil sind dann 0.

Aufgabe 2: Texturen

Falls das Objekt am Schnittpunkt mit einer Textur überzogen ist, wird der Farbwert der Textur mit dem eben ermittelten Farbwert multipliziert (entspricht in OpenGL GL_MODULATE). Um eine Textur auf ein Dreieck abzubilden, muss man wissen, welcher Teil der Textur wie auf das Dreieck gemappt werden soll. Dazu wird das Texturbild parametrisiert. Die zwei Parameter *s*, *t* gehen von 0 bis 1 und parametrisieren das Bild wie in Abbildung 6 dargestellt. Die an den Vertices des Dreiecks gegebenen Texturkoordinaten entsprechen dann genau den Bildkoordinaten *s* und *t*.
Um zu wissen, welches Pixel der Textur (also welcher Farbwert) für die jeweilige Stelle im Dreieck herangezogen werden muss, braucht man die dortigen Texturkoordinaten. Für die Eckpunkte des Dreiecks sind sie gegeben, also müssen sie im Dreieck interpoliert werden. Man verfährt dabei genauso, wie bei der Interpolation der Normalen.
Die Texturkoordinaten für jedes Dreieck sind im Quellcode in der *Triangle* Struktur unter *texCoords* gegeben (es sind natürlich nur die ersten 2 Einträge des jeweiligen *texCoords*-Vectors notwendig).

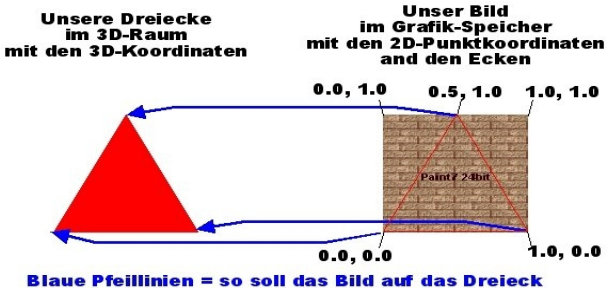

Abbildung 2: Texturierung eines Dreicks

Im Quellcode sind folgende Member wichtig:
Struktur *Triangle*: *texCoords* - die Texturkoordinaten
Struktur *Material:* *isTexture* – ist wahr, falls das Material mit einer Textur überzogen ist
texture – Die eigentliche Textur vom Typ *QImage* (Klassendokumentaion anschauen!)

Hinweis: *QImage* speichert die einzelnen RGB-Werte im Bereich von 0 bis 255. Die Farbwerte, die beim Raytracer berechnet werden, sind jedoch im Bereich von 0 bis 1 gegeben. Es ist also eine Umrechnung nötig (genau wie beim Rückgabewert der raytrace-Funktion).