

Übung zu Computergraphik I

– Übungsblatt 5 –

Lehrstuhl für Computergraphik
und Multimediasysteme

Andreas Görlitz, Jan Mußmann

Abgabe: Bis spätestens 21. November 2017, 10 Uhr

Besprechung: Dienstag 28. November 2017 und Mittwoch 29. November 2017

Hinweis: Die Programmieraufgaben müssen per E-Mail an **Jan Mußmann** eingereicht werden. Geben Sie dabei bitte immer Ihren **Namen**, Ihre **Matrikelnummer**, sowie Ihre **Übungsgruppe (Di. / Mi.)** an. Geben Sie nur die von Ihnen **geänderten Dateien** ab (`GLWidget.cpp`, `vs.glsl` und `fs.glsl`).

Hinweis: Die Programmieraufgaben müssen per E-Mail an Ihren Tutor eingereicht werden. Geben Sie dabei bitte immer Ihren Namen, Ihre Matrikelnummer, sowie Ihre Übungsgruppe an.

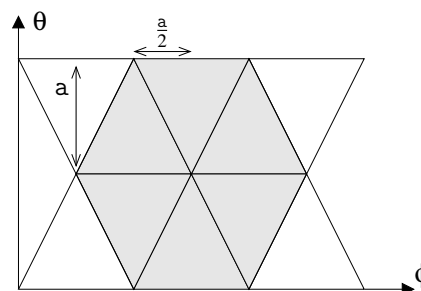
Aufgabe 1 Phong-Beleuchtung (6 Punkte)

In der folgenden Aufgabe soll eine kleine OpenGL-Anwendung geschrieben werden, in der eine beleuchtete Sphäre dargestellt wird. Nehmen Sie als Ausgangsbasis das auf der Webseite bereitgestellte Programmgerüst `ueb05.zip`. Um das Projekt zu kompilieren, folgen Sie der Anleitung auf der CG1 Übungsseite.

Das gegebene Programm enthält die Funktion `setupGeometry(float a, float r)`, die Punkte auf der Oberfläche einer Sphäre berechnet und darstellt. Die Berechnung der Oberflächenpunkte erfolgt in Abhängigkeit zweier Winkel $\theta \in [0, \pi]$ und $\phi \in [0, 2\pi]$ nach der folgenden Formel, wobei r der Sphärenradius ist:

$$\mathbf{r}(\theta, \phi) = r \begin{pmatrix} \sin\theta \cos\phi \\ \sin\theta \sin\phi \\ \cos\theta \end{pmatrix}$$

Im Folgenden soll eine Triangulierung der Sphäre durchgeführt werden. Außerdem soll die Sphäre mit Hilfe des Phong-Modells beleuchtet werden. In der folgenden Abbildung ist die beleuchtete Sphäre sowie das Schema der Triangulierung zu sehen:



Anmerkung: Im Gegensatz zu Übung 4 werden in dieser Aufgabe die Positionen und Normalen in einem *QVector* gespeichert. Objekte dieser Klasse verwalten den Speicher für Sie dynamisch, so dass Sie die Daten leichter hinzufügen (`vertices << glm::vec3(x, y, z)`) und die Größe des aktuell verwendeten Speichers bestimmen können (`vertices.size()`).

- 1.1 Ergänzen Sie die Funktion `setupGeometry(float a, float r)`, so dass die Sphäre als Dreiecksnetz dargestellt wird (siehe Abbildung). Sie können hierzu die bereits definierten `for`-Schleifen verwenden.

Hinweis: Mit den Tastatur-Tasten „+“ bzw. „-“, können sie den Wert `a` erhöhen bzw. verringern.

- 1.2 Implementieren Sie die Phong-Beleuchtung im Vertex-Shader.
- 1.3 Erklären Sie das Verhalten des spekularen Lichtanteils bei kleinen Werten von `a`.¹ Erklären Sie wie sich der spekulare Lichtanteil verhielte, wenn man die Beleuchtung im Fragment-Shader implementieren würde.
- 1.4 Das Blinn-Phong-Modell verwendet im Unterschied zur Phong-Beleuchtung anstelle des reflektierten View-Vektors $\hat{\mathbf{r}}_v$ den sogenannten Half-Way-Vektor

$$\hat{\mathbf{h}} = \frac{\hat{\mathbf{v}} + \hat{\mathbf{l}}}{\|\hat{\mathbf{v}} + \hat{\mathbf{l}}\|}.$$

Anstelle des Terms $(\hat{\mathbf{l}} \cdot \hat{\mathbf{r}}_v)$ in der Formel für den spekularen Lichtanteil wird das innere Produkt $(\hat{\mathbf{n}} \cdot \hat{\mathbf{h}})$ gebildet. Implementieren Sie das Blinn-Phong-Modell und vergleichen Sie das Ergebnis mit dem Phong-Modell.

¹Drücken Sie hierzu sehr oft die Taste „-“ auf Ihrer Tastatur.