Ludwig-Maximilians-Universität München
LFE Medieninformatik
Prof. Dr. Heinrich Hußmann
Henri Palleis

Übungsblatt 9: Einführung in GLSL

Abgabe:

Dieses Übungsblatt ist einzeln zu lösen. Die Lösung ist bis **Montag, den 09.Juli 2012, 12:00 Uhr s.t.** über UniWorx (https://uniworx.ifi.lmu.de/) abzugeben.
Für Textantworten werden nur die Formate PDF und Plain-Text (UTF-8) akzeptiert. Benennen Sie die Dateien nach dem Schema <Übungsblatt>-<Aufgabe>.<extension>, d.h. die Lösung der ersten Aufgabe geben Sie in einer Datei 1-1.txt oder 1-1.pdf ab. Packen Sie alle Dateien in eine ZIP-Datei und laden Sie diese bei UniWorx hoch. Wenn Sie Formatierungsvorgaben nicht einhalten, können ihre Abgaben nicht korrigiert werden.

Aufgabe 1: Erste Schritte

Ziel dieser Aufgabe ist es, die Grundzüge der OpenGL Shading Language (GLSL) kennenzulernen. Lesen Sie hierzu z.B. folgenden Artikel: http://aerotwist.com/tutorials/an-introduction-to-shaders-part-1/.

i. Erklären Sie die Begriffe Vertex- und Fragment-Shader

ii. Laden Sie die ZIP-Datei blatt9.zip von der Vorlesungswebsite herunter und öffnen Sie die Datei start_here.html in einem Webbrowser, der WebGL unterstützt (http://get.webgl.org/). Sie sollten einen türkisfarbenen Kreis sehen.

iii. Gehen Sie die Datei durch und verdeutlichen Sie sich deren Struktur.

   a. Wodurch wird der Kreis erzeugt?

   b. Wie wird sein Material definiert?

   c. Wie wird die Szene gerendert?

iv. Worum handelt es sich bei Uniforms, Attributes und Varyings? Wie können OpenGL und GLSL miteinander kommunizieren?

Aufgabe 2: Toon Shading

Beim Toon-Shading handelt es sich um eine einfache Methode, mit der man Objekte nicht-fotorealistisch rendern kann. Dazu wird im einfachsten Fall eine Farbe in wenigen unterschiedlichen Tönen verwendet. Welcher Farbton verwendet wird um ein Pixel zu färben,

hängt dabei vom Winkel zwischen der Richtung des Lichts und der Normalen der Oberfläche ab: Je kleiner der Winkel, desto heller der Farbton.

Implementieren Sie entsprechende Vertex- und Fragment-Shader. Gehen Sie wie folgt vor:

Vertex-Shader:

- Berechnen Sie die Position des Knoten in Weltkoordinaten
- Definieren Sie ein virtuelles Punktlicht (d.h. eine Position in Weltkoordinaten)
- Berechnen Sie den Vektor vom Knoten zum virtuellen Licht (Lichtrichtung)
- Normalisieren sie den Normalvektor und den Vektor für die Lichtrichtung (normalize()-Funktion von GLSL)
- Berechnen Sie als Wert für die Intensität den Winkel zwischen Normale und Lichtrichtung (sie können die GLSL-Funktion dot(), verwenden, der zwei normalisierte Vektoren als Parameter übergeben werden)
- Sorgen Sie dafür, dass die Variable für die Intensität im Fragment-Shader verfügbar ist

Fragment-Shader

- Mit welcher Farbe der Pixel eingefärbt wird, soll von der Intensität abhängen
- Sie könnten z.B. folgende vier Farbtöne verwenden: rgba(1.0,0.5,0.5,1.0), rgba(0.6,0.3,0.3,1.0), rgba(0.4,0.2,0.2,1.0), rgba(0.2,0.1,0.1,1.0)

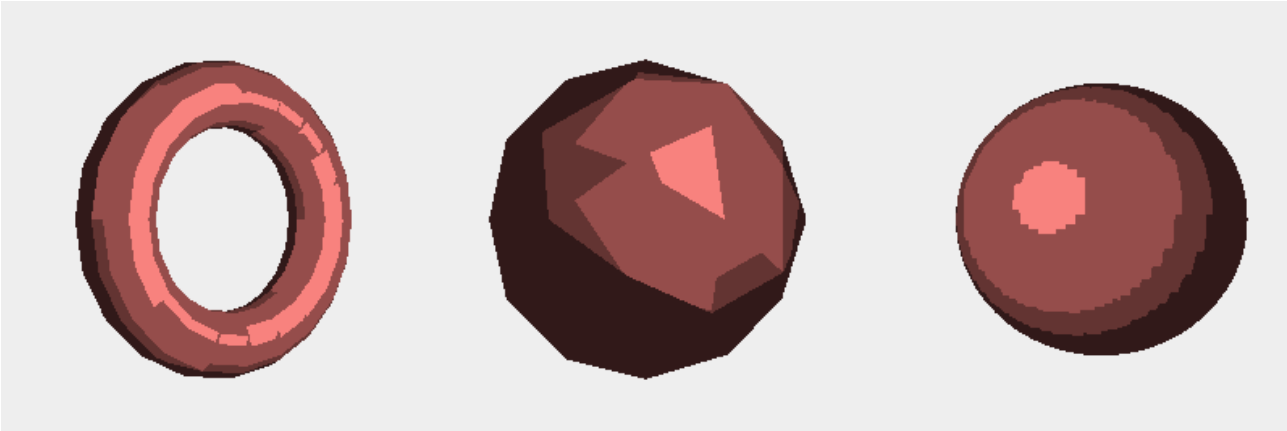

Abbildung 1 Toon-Shading

Aufgabe 3: Phong-Shading

Ziel dieser Aufgabe ist es, ein einfaches Phong-Shading zu implementieren. Gehen Sie von einem virtuellen Punktlicht bei Position (50.0, 100.0, -100.0, 1.0) aus. Licht und Material sind dabei folgendermaßen definiert:

Licht: Ambient = (0.2,0.2,0.2,1.0), diffus = (1.0,1.0,1.0,1.0) , spekular = (1.0,1.0,1.0,1.0)

Material: Ambient = (0.0,1.0,1.0,1.0), diffus = (0.8,1.0,0.3,1.0) , spekular = (1.0,1.0,1.0,1.0), shininess = 80.0

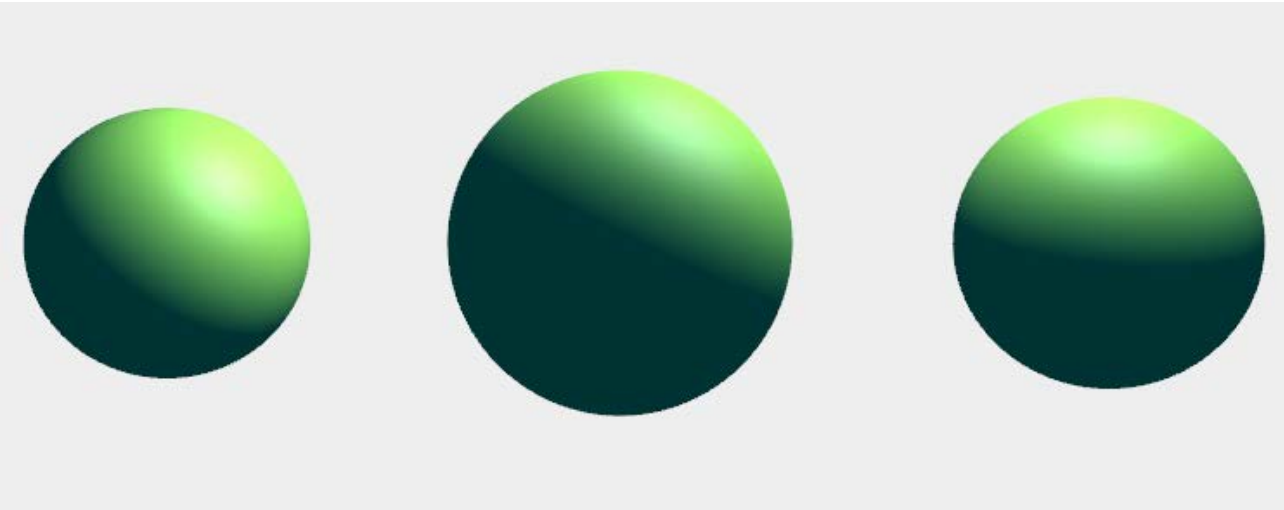

Abbildung 2 Phong-Shading mit GLSL

Optional:

Erweitern Sie die den Vertex-Shader derart, dass die drei Kugeln regelmäßig größer und kleiner werden (so, als würden Sie atmen). Definieren Sie hierzu außerhalb der Shader eine Funktion, die in Abhängigkeit einer fortlaufend inkrementierten Variable eine Schwingung berechnet (z.B. Sinus-Funktion). Dieser Wert kann zusammen mit dem Normalvektor im Vertex-Shader dazu verwendet werden, um den gewünschten Effekt zu erreichen.

Viel Erfolg.