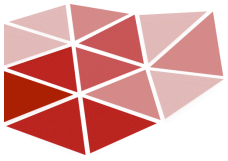

CG1 Übungsblatt 3

24. November 2008

Abgabe: Donnerstag, 15. Dezember 2008

AUFGABE 1: **Programmable Shaders**

In dieser Übung sollen Sie die Verarbeitung von Dreiecksnetze sowie programmierbare Shader erkunden. Unter http://www.cg.tu-berlin.de/menue/studium_und_lehre/computer_graphics_1/ steht Ihnen ein C/C++ Programmskelett zur Verfügung, welches Sie zur Lösung verwenden sollten.

1. Modelle in Form von Dreiecksnetzen sollen geladen und angezeigt werden können. Als Modellformat soll OFF verwendet werden. Dieses ist wie folgt aufgebaut:

Zeile	Inhalt	Beschreibung
1	OFF	Datei-Kennung
2	V F E	V: Anzahl der Dreiecksknoten, F: Anzahl der Polygone, E: Anzahl der Kanten
3	x y z	Koordinate des Knotens mit Index 0
4	x y z	Koordinate des Knotens mit Index 1
...	x y z	Koordinate des Knotens mit Index ...
V+2	x y z	Koordinate des Knotens mit Index V-1
V+3	n i_1 i_2 … i_n	Polygon mit n Knoten in den Positionen der durch i_k referenzierten Einträge
V+4	n i_1 i_2 … i_n	Polygon mit n Knoten in den Positionen der durch i_k referenzierten Einträge
...	n i_1 i_2 … i_n	Polygon mit n Knoten in den Positionen der durch i_k referenzierten Einträge
V+F+2	n i_1 i_2 … i_n	Polygon mit n Knoten in den Positionen der durch i_k referenzierten Einträge

   Implementieren Sie die eine Klasse `Mesh`, welche ein Dreiecksnetz repräsentieren kann. Diese Klasse soll eine Funktion `readOff(...)` besitzen, welche Dateien im oben spezifizierten Format einlesen kann. (10 Punkte)
2. Erweitern Sie Ihre Klasse `Mesh`, so dass diese aus den Vertices eines Dreiecksnetzes Normalen für jede Facette berechnen kann. Die Normalen sollten zusammen mit den anderen Informationen in `Mesh` gespeichert werden. Welche Vor- und Nachteile hat das von Ihnen gewählte Verfahren zur Normalenerzeugung? (10 Punkt)
3. Implementieren Sie eine Funktion `Mesh::Render()`, welche das Dreicksnetz mit OpenGL im Rahmen des zur Verfügung gestellten Programmes darstellt. Ihre Funktion sollte es auch ermöglichen, dass die Normalen visualisiert werden. (10 Punkt)
4. Implementieren Sie einen Vertexshader, welcher sowohl die notwendigen Transformationen für die Vertices also auch ein Phong Beleuchtungsmodell implementiert. Ihr Programm sollte dabei sowie die OpenGL fixed function pipeline, als auch Ihren Shader verwenden können, und es sollte die Möglichkeit bestehen dynamisch zwischen diesen Alternativen zu wechseln. (20 Punkt)
5. Eine Möglichkeit der visuelle Erscheinung eines Dreiecksnetzes mehr Detail hinzuzufügen ohne die geometrische Komplexität zu erhöhen, ist die Normalenvektoren lokal zu variieren. Wir beschränken uns in dieser Übung auf eine globale Variation. Ihr Programm soll es dafür ermöglichen, alle Normalen des Dreiecksnetzes synchron relativ zu ihrer urspünglichen Position zu drehen. Die Drehung soll dabei im Vertexshader erfolgen. Wann kann eine solche Drehung im Shader notwendig beziehungsweise sinnvoll sein? (10 Punkt)

Bonus: Implementieren Sie per-pixel Phong Shading und demonstrieren Sie dessen Vorteile. (5 Punkte)

Technische Universität Berlin
Computer Graphics

Prof. Dr. Marc Alexa, Prof. Dr. Bernd Bickel, Christian Lessig
Ronald Richter

AUFGABE 2: **Theoriefragen**

1. Phänomenologie von Beleuchtungsmodellen.
   a) Ein Dreieck wird mittels Gouraud shading gezeichnet. Welches Bild ist zu erwarten, wenn eine Normale in Richtung Lichtquelle zeigt und die anderen Normalen davon abweichen. (5 Punkte)
   b) Auf welche der Beleuchtungskomponenten {ambient, diffus, spekulär} hat die Position des Betrachters bei einer ansonsten statischen Szene Einfluss? Begründen Sie Ihre Antwort. (5 Punkte)

2. In 1978 führte Jim Blinn „bump mapping“ in der Computergrafik ein. Nennen Sie Vor- und Nachteile des Verfahrens. Warum wird es insbesondere im Bereich von Computerspielen häufig verwendet? (10 Punkte)

3. Aus welchem Grund werden beim Aufbau eines BSP Baumes aus einem Dreicksnetz oftmals die enthaltenen Dreiecke selbst zum Spannen der Partitionierungsebene verwendet? Nennen sie mindestens eine weitere Möglichkeit, Partitionierungsebenen zu bestimmen. (5 Punkt)

4. In der Vorlesung wurde das Phong Shadingmodell diskutiert. Eine alternative Formulierung, erneut von Blinn, verwendet den „half vector“.
   a) Was ist der half vector und welche Vorteile bietet er? (5 Punkte)
   b) Leiten Sie das Phong Shadingmodell unter Verwendung des half vectors her. (5 Punkte)
   c) Was ist bei der Verwendung des half vectors zu beachten? (5 Punkte)

5. Texturierung einer Kugel.
   a) Schlagen Sie neben den kanonischen sphärischen Koordinaten $(\theta, \phi) \in [0, \pi] \times [0, 2\pi]$ noch eine weitere Möglichkeit vor, um Texturkoordinaten für die Kugel zu erzeugen. (5 Punkte)
   b) Für beide Verfahren, wie groß ist die Flächenverzerrung am Äquator und an den Polen. (5 Punkte)
   c) Berechnen Sie die Flächenverzerrung für einen beliebigen Punkt, und verifizieren sie Ihre quantitativen Aussagen aus b.). (5 Punkte)
   d) Schlagen Sie ein praktikables Verfahren zur Texturierung der Kugel vor, welches einen Kompromiss zwischen Flächenverzerrung, Implementierungsaufwand, und Geschwindigkeit bietet. (5 Punkte)