GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Prof Dr. Winfried Kurth,
Alex Tavkhelidze

Praktikum Computergrafik, WiSe 17/18
Übungsblatt 4

- ✓ **Abgabefrist**: 19.01.2018 09:00:00
- ✓ Abgabe erfolgt per E-Mail an jeos@mail.com
- ✓ **Betreff**: CG17WS ÜB4
- ✓ **Erste Zeilen der E-Mail**: Name der Autoren und Matrikelnummern.
- ✓ Der **lauffähige Code** soll **als Anhang** in der E-Mail mitgeschickt werden.
- ✓ Der Quelltext muss dabei mit Eclipse in **ein ZIP-Archiv** exportiert worden sein
  (*siehe die Anleitung in Folien zum Übungsblatt 1*)
  Bemerkung:
  Für jede programmierbezogene Aufgabe muss eine separate ZIP-Datei exportiert & beigefügt werden.

Quellen

Aufgaben http://www.uni-forst.gwdg.de/~wkurth/cg17_u04.pdf

Codeblöcke	http://www.uni-forst.gwdg.de/~wkurth/cg17_1.txt http://www.uni-forst.gwdg.de/~wkurth/cg17_2.txt http://www.uni-forst.gwdg.de/~wkurth/cg17_3.txt

Liste der Aufgaben:

1. **Vorlage/Codeblöcke:** *siehe die mitgeschickte / beigefügte Datei **1.txt** (für bessere Lesbarkeit in* **Eclipse** *oder in* **Notepad++** *mit abgehackter **Spracherkennung -> Java** öffnen)*
   Mithilfe der Vorlage, erstellen Sie den Code, der die folgendermaßen beschriebene Szene aufbaut:
   - Begrenzter flacher Boden (also kein Bodenhorizont) mit dem Farbverlauf, der bzw. in einem bestimmten Teil der ganzen Szene verlegt wird;
   - Vom Boden etwas abgehobener Würfel, mit den einfärbig und dabei unterschiedlich gefärbten Flächen;
   - Gleichmäßige Bewegung (d.h. mit einem konstanten Schritt) des Betrachters in 6 Richtungen durch die Szene mit den entsprechenden Tastaturtasten – *siehe Details in der Vorlage **0.txt*** (beim **Übungsblatt 3**, in Kommentarzeilen);
   - Gleichmäßige 360°-Rotation der Blickrichtung in 4 Richtungen – steuerbar durch die entsprechenden Tastaturtasten (*siehe Details in der Vorlage **0.txt***, beim **Übungsblatt 3**, in Kommentarzeilen);
   - Mouse-gesteuerte gleichmäßige Rotation des Würfels in 4 unterschiedlichen Richtungen, nur bei der gedrückten Umschalttaste (d.h. sonst muss der Würfel unbeweglich bleiben und auf Mouse-Ereignisse nicht reagieren).

     Anmerkungen:
     - der Abgleich der tatsächlichen Rotationsrichtung des Würfels mit der maus-bestimmten Richtung wird dabei nicht benötigt, allerdings sollten die beiden logisch verknüpft sein (also die beiden sollen sich ähneln) – feinere Lösungen werden allerdings empfohlen, aber kein Muss.
     - Die Art und Weise, wie man diese „Würfel-Maus“ Kopplung einrichtet (d.h.:

- die dafür zu verwendenden Befehle unter der **MouseListener** / **MouseMotionListener** Schnittstelle(n) **und**
- die nachfolgenden **Berechnungen/Transformationen/Zuordnungen** von maus-induzierten Richtungen zu entsprechenden Rotationsrichtungen des Würfels),

steht unter freier Wahl.

Hinweis: falls man die Berechnungen einsetzt, könnten die einfachen (beispielsweise trigonometrischen) Methoden der Kernklasse **java.lang.Math** eventuell verwendet werden.

2. Rüsten Sie die Szene in der Aufgabe #1 mit folgenden zusätzlichen Funktionalitäten aus:
   - Die Tastaturtaste **B** soll die Rolle eines Schalters übernehmen, der den aktuell eingesetzten Boden zu einem anderen Boden wechselt – diese andere Option ist Quadratmaschenboden, der in der ganzen Szene verlegt wird (also die Szene soll mit dem Bodenhorizont rundum versehen sein);
   - Nur wenn die Feststelltaste ist eingeschaltet (d.h. wenn der entsprechende Indikator auf der Tastatur leuchtet), sollte die Rotation ausschließlich von der Tastatur bedient werden;
   - **Vorlage/Codeblöcke:** *siehe die mitgeschickte / beigefügte Datei **2.txt** (für bessere Lesbarkeit in* **Eclipse** *oder in* **Notepad++** *mit abgehackter **Spracherkennung -> Java** öffnen)*
     Die aktuelle *FPS-Rate (frames per second)* muss genau dort auf dem OpenGL-Schaufenster gezeichnet werden (*sowie auch während der ganzen Animation bleiben*), wo man zum ersten Mal mit der Maus klickt – verwenden Sie dafür **TextRenderer**.
   - Sie müssen mindestens eine komplett andere Art der maus-gesteuerten Rotation einführen - im Zusatz zur schon eingerichteten Steuerweise. Das bedeutet ***wenigstens ein anderes Maus-Ereignis***, das als Basis genommen wird. Sie werden aber ***angeregt***, auch die nachfolgenden **Berechnungen/Transformationen/Zuordnungen** wesentlich andersartig einzuführen.

3. **Vorlage/Codeblöcke:** *siehe die mitgeschickte / beigefügte Datei **3.txt** (für bessere Lesbarkeit in* **Eclipse** *oder in* **Notepad++** *mit abgehackter **Spracherkennung -> Java** öffnen) - die detaillierten Erläuterungen schauen Sie sich in Kommentare an.*
   Versehen Sie die Szene aus der Aufgabe #2 mit folgenden Elementen:
   a. Eine Lichtquelle (*Spotlight*);
   b. Wählen Sie ein bestimmtes Material aus dieser Sammlung und anschließend versehen Sie damit den Würfel;
   c. Visualisieren Sie die Lichtquelle:
      i. dafür ist das geometrische Primitiv glutSolidCone aus der Klasse GLUT einzusetzen – platzieren Sie diesen Kegel genau an die Stelle, an welche sich die Lichtquelle befindet; **Hinweis**: dies erfolgt nachdem Sie eine neue Instanz der Klasse (des Objekts) GLUT, beispielsweise durch die Variable *glut,* einführen und glut.glutSolidCone(GLdouble base, GLdouble height, GLint slices, GLint stacks) aufrufen;
      ii. dabei ist zu beachten, dass die Grundfläche des Kegels senkrecht zur Beleuchtungsrichtung bleiben soll;

Leichtere Option: sollten Sie bei der Umsetzung von i. und ii. in Schwierigkeiten geraten, verwenden Sie einfach glutSolidSphere (eine Sphäre, die keine Richtungsanpassung zur Beleuchtungsrichtung benötigt) - geben Sie für die Parameter GLint slices und GLint stacks genug große ganzzahlige Werte ein, damit die Sphäre weniger eckig aussieht.

iii. Es wird nur angeregt, die Farbe des GLUT-Objekts auf weiß zu setzen – richten Sie dafür einen emissiven (selbstleuchtenden) Lichtanteil der Lichtquelle ein (Parameter **GL_EMISSION**) und setzen Sie einen genug großen Wert für diesen Parameter;

d. Die Lichtquelle soll den Würfel in 4 Richtungen umkreisen, gesteuert über die Tastatur: **I** Taste (oben), **K** Taste (unten), **J** Taste (links), **L** Taste (rechts).

Anmerkung: die Codeblöcke in der Datei **3.txt** zeigen nur die Lauffähigkeit eines Beleuchtungsmodels und dienen auf keinen Fall als eine streng einzuhaltende Struktur bei der Lösung von 3. Aufgabe -

- ✓ genau die richtige Reihenfolge von Befehlen (***vor allem*** – Transformationen und Normalen)
- ✓ sowie auch die zufriedenstellende Anpassung von Parameterwerten (***vor allem*** – (RGBA-)Intensitäten)

sind zu bedenken und gehören zum Kern dieser Aufgabe.

Mehrere Kommentarzeilen sind dabei äußerst behilflich und werden daher zum Lesen empfohlen.
