Blatt 9

1 Aufgabe – OpenGL Setup (0 Punkte)

1.1 LWJGL Integration

Binden Sie den OpenGL-Wrapper LWJGL in ihre Entwicklungsumgebung ein. Dazu laden sie LWJGL in der aktuellsten Version von der entsprechenden Website in Ihr Arbeitsverzeichnis:

`http://lwjgl.org/download.php`

Importieren Sie im Anschluss das Projekt „CG14_OpenGL_Simple“. Sie werden einige Fehler feststellen die bei korrekter LWJGL-Einbindung verschwinden werden. Legen Sie im Folgenden eine neue Library für LWGJL an.

1. Build-Path → Add Libraries
2. User Library → Next → User Libraries → New → “lwjgl_2_9_1” → OK

Nun existiert in den User Libraries ein Bibliotheksordner für LWGJL 2.9.1. Importieren Sie JAR-Dateien aus dem zuvor heruntergeladenen Archiv.

1. Add External JARs…
2. `Laufwerk:\Nutzer\...\workspace\lwjgl-2.9.1\jar`
3. Alle vorhandenen JAR Dateien importieren

Abschließend muss für jedes importierte JAR-File der Pfad zur „Native Library Location“ gesetzt werden (initial mit „none“ belegt). Abhängig vom verwendeten Betriebssystem muss hier jeweils ein anderer Ordner importiert werden, die alle im selben Verzeichnis liegen.

1. Reiter bei „lwjgl_util.jar“ öffnen
2. Native library location: (NONE) → Edit…
3. External Folder…
4. `Laufwerk:\Nutzer\...\workspace\lwjgl-2.9.1\lwjgl-2.9.1\native\`**`windows, macosx, linux etc.`**
5. OK

Verfahren Sie entsprechend mit allen importierten JAR-Files der LWJGL Bibliothek.

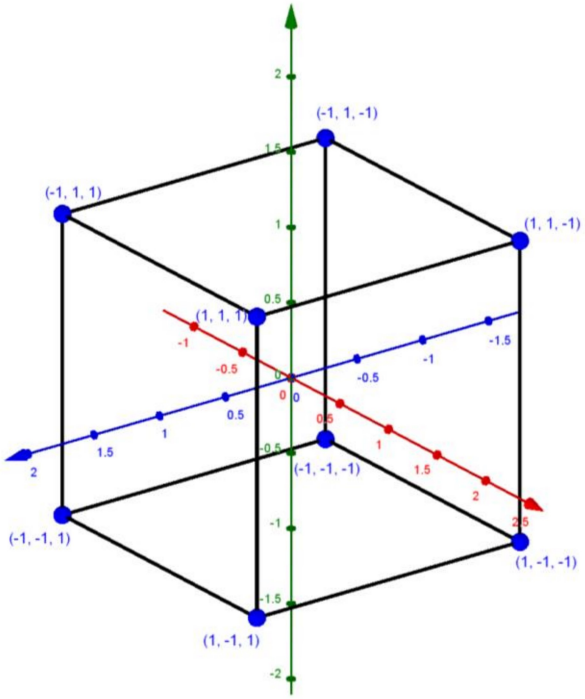
Führen Sie nun das Programm „`CubeScene_StudentVersion.java`“ aus. Es sollte ein Fenster eingefärbt in hellgrau dargestellt werden. Darüber hinaus gibt Ihnen die Konsole einen Info-Block über Ihr System und die genutzte Grafikkarte aus, sowie die aktuellen FPS (frames per second).

2 Aufgabe (10 Punkte)

Machen Sie sich mit den in Aufgabe 1 importierten Programmen vertraut. Beschreiben Sie Ihrem Tutor den Ablauf der Anwendung und gehen Sie dabei insbesondere auf die OpenGL Graphics Pipeline ein. Wozu dienen die Uniform-Variablen `model`, `viewProj` und `eyePosition`?

3 Aufgabe (45 Punkte)

Ihnen ist der folgende Würfel in einer 3D-Ansicht gegeben. Es handelt sich dabei um den Einheitswürfel. Ihre Aufgabe besteht im Wesentlichen aus den folgenden Schritten:



- Füllen Sie das Float-Array `vertices` mit den Punkten der Würfel-Flächen. Die TODOs im Quellcode helfen Ihnen beim Vorgehen.
- Füllen sie das Float-Array `normals` mit den entsprechenden Normalen der Flächen. Achten Sie auf die Reihenfolge d.h. liegen im `float[] vertices` als erstes die Vertices, die das „front face“ beschreiben, so liegen im `float[] normals` entsprechend die Normalen dieses face.
- Der Index-Buffer wird Ihnen vorgegeben. Erklären Sie ihrem Tutor wie die ersten beiden Zeilen im `int[] indices` zustande kommen.

4 Aufgabe (10 Punkte)

In der Anwendung wurde bereits die Funktionalität einer Kamera implementiert mit der man im Freiflug durch die Szene navigieren kann. Erweitern Sie das Programm, sodass sich der Würfel sichtbar (aber nicht zu schnell) um die Y-Achse dreht, ohne dass dies von Nutzereingaben initiiert wird.

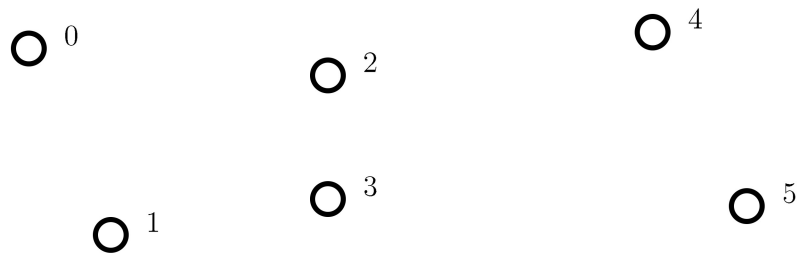
5 Aufgabe (10 Punkte)

Im der Anwendung wurde ein einfaches Beleuchtungsmodell implementiert. Interpretieren Sie dieses an Hand des gegebenen Quellcodes (insbesondere im Vertex Shader) und fertigen Sie eine Skizze an, wie man sich das Modell vorzustellen hat und gehen Sie dabei auch auf die spekulare Reflektion ein?

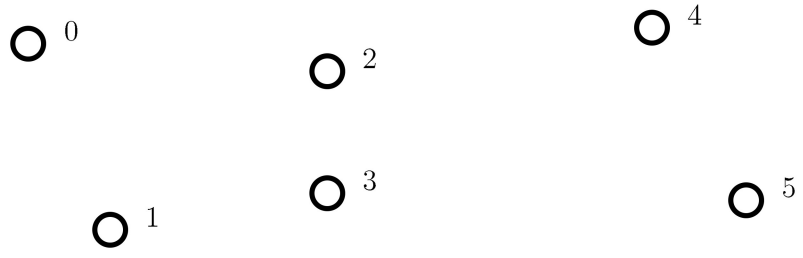
6 Aufgabe (5 Punkte)

Gegeben sei der Indexbuffer I=(0, 1, 2, 3, 4, 5). Zeichnen Sie die entstehende Geometrie zu den jeweiligen Topologien in die zugehörigen Skizzen.

GL_LINES



GL_LINESTRIP



7 Aufgabe (20 Punkte)

Beantworten Sie Ihrem Tutor Fragen zum Inhalt der Vorlesung. Bereiten Sie sich dabei insbesondere auf Fragen zu den technischen Grundlagen und Konzepten (Vertices, Fragments, Shader, Stationen der Graphics-Pipeline, etc.) von OpenGL vor.