**Ausarbeitung für das GDV-Praktikum**

**Von Feridun Aydogan(756852) und Timo Nock(755486)**

**1.Was sehen sie nach der Ausführung des Programmes im Graphik-Fenster?**

Nach der Ausführung des Programmes sehen wir ein rotes Quadrat mit Schwarzem Hintergrund und am Graphik—Fensterrahmen sehen wir die eingegebenen Namen.



**2.Sind die Anweisungen *glBeginn* und *glEnd* unbedingt notwendig? Was passiert, wenn man sie weglässt?**

Das Programm wird gebildet und ausgeführt, jedoch wird die Graphik bzw der Code der zwischen glBeginn und glEnd steht, nicht ausgeführt bzw angezeigt. Weshalb glBeginn und glEnd notwendig ist.

Laut der Webpage „<https://wiki.delphigl.com/index.php/glBegin>“ umschließt glBegin und glEnd  eine Liste von [Eckpunkten (Vertices)](https://wiki.delphigl.com/index.php/Eckpunkt) die eine [Primitive](https://wiki.delphigl.com/index.php/Primitive) oder Gruppe von [Primitiven](https://wiki.delphigl.com/index.php/Primitive) darstellt. glBegin erhält ein einzelnes Argument (*mode*), das angibt, auf welche Art und Weise die [Eckpunkte (Vertices)](https://wiki.delphigl.com/index.php/Eckpunkt) interpretiert werden.

Ohne glBeginn und glEnd: Mit glBeginn und glEnd:

 

**3. An welchen Positionen im Programm müssen diese glColor4f-Anweisungen stehen?**

Die glColor4f-Anweisung muss vor glVertex3f stehen, denn dadurch werden alle nachfolgenden „Vertexe´“ in der entsprechenden Farbe, durch die Anweisung gefärbt.



**4. Welche Position im Code ist am besten zum Aufruf von glClear geeignet und warum?**

Am besten vor glLoadIdentity(), da dieser alles zurücksetzt!



**5. In welcher Reihenfolge müssen glClear und glClearColor aufgerufen werden?**

Natürlich sollte bzw. muss zuerst glClearColor() aufgerufen werden, bevor glClear() aufgerufen wird.

**6. Was passiert, wenn Sie in einer Animation glClear(GL\_COLOR\_BUFFER\_BIT); weglassen?**

Die die Frames für jede Konstante überlappt sich.

**7. Welche der beiden Flächen sehen Sie?**

Wir sehen die grüne Fläche.

**8. Erzeugen Sie die Flächen mal in einer anderen Reihenfolge: Was fällt Ihnen auf und warum ist das so?**

Die zuletzt Codierte Fläche bzw. die Fläche die am spätesten erstellt wird überzeichnet die alte Fläche.

**9. a) Entspricht die Ansicht der Flächen nun Ihren Erwartungen?**



Ja, wenn man die Fenstergröße verändert, verändert sich die ansicht bzw. man sieht das streckne und stauchenstreckt und staucht

**b) Beschreiben Sie kurz, wie der Z-Buffer funktioniert.**

Durch Tiefeninformationen in einem sogenannten Z-Buffer („Z-Puffer“) stellt das Verfahren pixelweise fest, welche Elemente einer Szene gezeichnet werden müssen und welche verdeckt sind

**10. Welche Fläche sehen Sie und warum sehen Sie gerade diese Fläche?**

Man sieht die Rückseite mit den Farben Blau, Cyan, Schwarz und Grün , weil die Kamera vom Ursprung nach vorne schaut, also vom inneren des Würfels



**11. Probieren Sie (ohne Translation des Würfels) die folgenden Kamerapositionen aus und dokumentieren Sie die dazu verwendeten gluLookAt-Aufrufe:**

**a) Betrachtung der Szene von vorne oben; (Kameraposition: (0., 1., 1.). Sind die Parameterwerte in glOrtho richtig gesetzt? Falls NEIN, was stimmt nicht?**

Um etwas mehr „glOrtho“ zu verstehen fand ich folgendes auf der Webpage: „<https://wiki.delphigl.com/index.php/glOrtho>“ **glOrtho** schaltet OpenGL praktisch in einen 2D-Modus , weshalb die Z-Koordiante keine Rolle mehr im Bezug auf die letztendliche Größe eines Objektes hat (weit entfernte Objekte (mit hoher Z-Koordinate) werden genau so groß gezeichnet, wie nahe.)

Und bei mehrfacher Ausführung des Codes mit folgenden Kamerapositionen…

gluLookAt(0., 0.4, 1., 0., 0., 0., 0., 1., 0.);

gluLookAt(0., 0.5, 1., 0., 0., 0., 0., 1., 0.);

gluLookAt(0., 0.6, 1., 0., 0., 0., 0., 1., 0.);

gluLookAt(0., 0.7, 1., 0., 0., 0., 0., 1., 0.);

gluLookAt(0., 0.8, 1., 0., 0., 0., 0., 1., 0.);

merkt man das der Würfel Verschwindet bzw. der Würfel befindet sich wahrscheinlich nicht im Frustum der Kamera.

Wir stellen fest das glOrtho falsch gesetzt ist und wir zu nahe am Objekt sind.

**b) Betrachtung der Szene direkt von rechts; Kameraposition: (1., 0., 0.). Wie lauten die kompletten Aufrufe von glortho und gluLookAt?**



gluLookAt ( 1., 0., 0., 0., 0., 0., 0., 1., 0.)

glOrtho( -1., 1., -1., 1., 0.0, 3.0)

**c) Betrachtung der Szene von rechts oben: wie lautet die Kameraposition:**

**(?, ?, ?).**

Wir betrachten von rechts oben mit folgendem code:

gluLookAt ( 1., 1, 0., 0., 0., 0., 0., 1., 0.)

**12. Ist es möglich, den Rotate-Befehl vom „oberen“ Teil in die beiden Äste des Szenegraphs zu verlagern? (Wenn JA, wohin und wie? Wenn NEIN, warum nicht?)**

Man kann die Rotate-Befehle vom „oberen“ teil in die beiden Äste des Szenengraphs verteilen. Und zwar unter den glPushMatrix() befehlen.