Prof. Dr. R. Dörner
Fachbereich Design Informatik Medien
HS RheinMain

Praktikum zur Computergraphik

Übung 7

**zu Teil H (Projektion, Koordinatensysteme), I (Culling, Clipping)
J(Verdeckungsrechnung)**

Aufgabe 7.1

Diese Aufgabe behandelt nun die zweite Projektionsart, die es neben der perspektivischen Projektion gibt: die Parallelprojektion.

Eine Parallelprojektion soll durchgeführt werden. Die Projektionsstrahlen haben die Richtung (0,1,-1). Die Bildebene sei die Ebene mit der Ebenengleichung y – z = 0.

(a) Zeigen Sie: die Projektionsstrahlen stehen senkrecht auf der Bildebene.

(b) Um die Projektionsmatrix M_{par} anwenden zu können, muss zuerst dafür gesorgt werden, dass bestimmte Projektionsbedingungen eingehalten werden: die Bildebene in unserer Aufgabe befindet sich nicht an der entsprechenden Stelle. Wo sollte sich denn die Bildebene befinden? Und welche Transformation muss durchgeführt werden, damit die Standard-Projektionsbedingungen erhalten werden?
[Lösungshinweis: Rx(-45°)]

(c) Wie lautet die 4x4 Matrix, die dann die Parallelprojektion durchführt?
[Lösungshinweis: Mpar * Rx(-45°)]

(d) Wie lauten die Koordinaten des Punktes A(1,2,-1) nach der Projektion in Bildkoordinaten?
[Lösungshinweis: A‘(1 / 0,707…)]

Aufgabe 7.2

Ziel der Aufgabe ist es, die Abbildung eines Szenengraphs in OpenGL zu realisieren

Nehmen Sie als Ausgangspunkt den Szenengraph, der in Aufgabe 1.3 in VRML definiert wurde. Schreiben Sie ein entsprechendes Javascript, das die notwendige Model-Matrix aufbaut, die an den Vertex-Shader übergeben werden sollte. Gehen Sie davon aus, dass es die Funktionen drawCone() und drawSphere(). Außerdem gibt es die beiden Methoden pushMatrix() und popMatrix(), die einen Matrix-Stack realisieren.

Aufgabe 7.3

Ziel der Aufgabe ist es, sich mit der perspektivischen Projektion (die am häufigsten eingesetzte Projektionsart) unter OpenGL vertraut zu machen. Der richtige Einsatz des Kamera-Modells wird faktisch in jedem OpenGL Programm benötigt.

Gegeben ist folgender Auszug aus einem Javascript:

```
display( )
{
      glClearColor(1.0, 1.0, 1.0, 1.0);
      glClear(GL_COLOR_BUFFER_BIT);
      var viewMatrix = lookAt(0.0,0.0,3.0,0.0,0.0,0.0,0.0,-1.0,0.0);
      var modelMatrix = mat4();
      // modelMatrix = mult(modelMatrix, translate(0.0,0.0,-3.0));
      modelMatrix = mult (modelMatrix, rotate(30.0,0.0,0.0,1.0));
      var projectionMatrix = frustum(-1.0,1.0,-1.0,1.0,1.0,10.0);
      drawSphere();

}
```

(a) Durch welche translate- und rotate- Befehle kann der obige lookAt-Befehl ersetzt werden, um das gleiche Ergebnis zu erzielen? (Reihenfolge der Befehle beachten!) Hinweis zur Parameterreihenfolge von lookAt: Eye,Center,Up
[Lösungshinweis: var viewMat = mat4(); viewMat=mult(viewMat,rotate(180,0,0,1); viewMat = mult(viewMat, translate(0,0,-3)]

(b) Die Kommentarzeichen vor dem translate-Befehl werden entfernt. Wie wird sich das gerenderte Bild dadurch verändern? Begründen Sie Ihre Aussage!

(c) Wie verändert sich das Bild gegenüber der Original-Datei, wenn der lookAt-Befehl ersetzt wird durch lookAt(0.0,0.0,3.0,1.0,0.0,0.0,0.0,1.0,0.0); (Begründung nicht vergessen).

(d) Wie verändert sich das Bild gegenüber der Original-Datei, wenn der frustum-Befehl ersetzt wird mit frustum(-1.0,5.0,-1.0,1.0,1.0,100.0); (Begründung nicht vergessen). Hinweis: Parameterreihenfolge frustum: left,right,bottom,top,near,far

(e) (Zusatzaufgabe) Was passiert, wenn man den lookAt Befehl durch mat4() ersetzt?

Aufgabe 7.4

Die Verdeckungsrechnung ist ein wesentlicher Teil des Renderings. In der Computergraphik gibt es dafür eine Reihe von Algorithmen

(a) Der Maler-Algorithmus ist ein Algorithmus, der Verdeckungen beim Rendern berücksichtigen kann. Erläutern Sie die prinzipielle Funktionsweise dieses Algorithmus!

(b) Der Maler-Algorithmus funktioniert nicht immer. Geben Sie ein Beispiel für einen Fall an, der mit dem Maler-Algorithmus nicht richtig gerendert werden kann.

(c) Aufgrund der Beschränkungen des Maler-Algorithmus wird heute häufig ein anderer Algorithmus für die Verdeckungsrechnung eingesetzt, der in OpenGL

mit dem Befehl **glEnable(GL_DEPTH_TEST)** aktiviert wird. Wie heisst dieser Algorithmus? Geben Sie diesen Algorithmus in Pseudo-Code an!

Aufgabe 7.5

Wahr oder Falsch?

Das Raytracing Verfahren kann man für das Rendering verwenden.	
Jede durch ein Polygon begrenzte Fläche kann nur auf eine Weise trianguliert werden.	
Die Berechnung der diffusen Reflektion beim Phong-Beleuchtungsmodell ist von der Position des Betrachters unabhängig.	
Die Shininess beim Phong-Beleuchtungsmodell bestimmt, wie groß die Highlights werden: je größer die Shininess, desto kleiner die Highlights, desto glatter wirkt das beleuchtete Objekt.	
Bildkoordinaten erhält man, indem bei Clippingkoordinaten einfach den z-Wert weglässt.	
Für das Backface Culling ist die Richtung der Normalen irrelevant.	
Es ist günstiger das Backface Culling vor dem Clipping durchzuführen als die umgekehrte Reihenfolge zu wählen.	
Ein Vorteil des Bresenham Algorithmus gegenüber einem Digital Differential Analyser besteht darin, dass dieser durch einfacher und schneller ausführbare Operationen im Prozessor realisiert werden kann.	
Die Entscheidungsregel im Bresenham-Algorithmus für das Rasterisieren einer Linie mit einer Steigung m, die zwischen 0 und 1 liegt, lautet: liegt die Linie unter oder auf dem Entscheidungspunkt wird der Ost-Pixel gewählt, ansonsten der Nordost-Pixel.	
Um Gouraud-Shading zu realisieren, kann die im Vertex-Shader in der Beleuchtungsrechnung bestimmte Farbe einfach als varying-Variable an den Fragment-Shader übergeben werden.	
In VRML kann keine Parallelprojektion realisiert werden.