Klausur Computergraphik (WS 2016/17)

Prüfer:	Prof. Dr. R. Dörner, Prof. Dr. C. Schulz, HS RheinMain
Bearbeitungszeit:	90 min
Zugelassene Hilfsmittel:	ein beidseitig handbeschriebenes DIN A4 Blatt, Stifte. (insbesondere Taschenrechner und eigenes Papier ist verboten)
Datum:	14. Februar 2017

Name:______________________________ Vorname: _________________________

Matr.-Nr. ___________________

____
Unterschrift

Hinweise:

- Überprüfen Sie Ihr Klausurexemplar auf Vollständigkeit (Umfang: 8 Blätter)
- Lösen die Aufgaben im dafür vorgesehenen Raum. Wenn der Platz nicht ausreicht, verwenden Sie die Rückseiten - wenn alle Rückseiten beschrieben sind, fordern Sie ein leeres Blatt bei der Aufsicht an. Schreiben Sie im vorgesehenen Raum einen Hinweis der Art "weiter siehe S. 3 Rückseite". Fehlt dieser Hinweis, ist die Lösung unleserlich oder gibt es mehrere Lösungen zu derselben Aufgabe, so werden keine Punkte vergeben.
- Wer einen Täuschungsversuch begeht oder einem Täuschungsversuch Vorschub leistet erhält die Note "nicht bestanden".
- Es darf nicht mit Bleistift geschrieben werden. Es sind nur Schreibfarben „blau“ oder „schwarz“ zulässig.
- Die Klausur ist in jedem Fall bestanden mit **36 Punkten.**

Es wurden __________ Punkte erreicht.

Note, Handzeichen:

Aufgabe 1

Gegeben sind die vier Stützpunkte A(4, 4, 0), B(0, 4, 0), C(4, 4, 4) und D (0, 0, 0) einer Bezier-Kurve Q(t), $t \in [0,1]$.

(a) Die Bezier-Kurve soll in zwei Bezier-Kurven R(t) und S(t) zerlegt werden ($t \in [0,1]$), die sich im Punkt Q(0,25) berühren und so einen Bezier-Spline bilden. Wie lauten die Stützpunkte A', B', C' und D' der Kurve S(t)? (Hinweis: Lösen Sie die Aufgabe rechnerisch mit dem DeCasteljau-Alg.)

6 P. A'(____, ____, ____), B'(____, ____, ____), C'(____, ____, ____), D'(____, ____, ____),

(b) Ist der Übergang zwischen R(t) und S(t) C^1-stetig (Begründung angeben!)?

2 P.

(c) Die Kurve H(t), $t \in [0,1]$, ist eine Hermite-Kurve, welche die gleichen Start- und Endpunkte wie Q(t) besitzt, sowie auch die gleichen Tangenten am Start- und Endpunkt. Geben Sie eine Formel aus nicht ausmultiplizierten Matrizen für H(0,25) an.

Die Basismatrix der Hermite-Kurven lautet dabei: $M_{Hermite} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$

4 P.

Σ2:

Aufgabe 2

Gegeben ist folgende VRML-Szene:

```
DEF T1 Transform{
        scale                   1 2 1
        translation             1 2 3
        children[
              DEF T2 Transform{
                      rotation      5 0 0 1.57
                      center        2 2 2
                      children[
                                    DEF S1 Shape{ geometry Sphere{  radius 2.0 } }
                                    DEF S2 Shape{ geometry Sphere{  radius 3.0 } }
                      ] # children T2
              } # T2
              DEF T3 Transform{
                                    scaleOrientation   5 0 0 1.57
                                    scale              2 2 2
                                    children[
                                                DEF S3 Shape{ geometry Sphere{ } }
                                    ] # children T3
              } # T3
              DEF S4 Shape{ geometry Sphere{ } }
        ] # children T1
} # T1
```

(a) Zeichnen Sie den Szenengraph (nur Shape- und Transform-Nodes, keine Fields)

3.5 P.

(b) Geben Sie eine Formel bestehend aus (nicht ausmultiplizierten) 4x4 Matrizen an, wie man die Koordinaten (x, y, z) eines Punktes aus dem lokalen Koordinatensystem von Kugel S3 umrechnet in lokale Koordinaten (x', y', z') im Koordinatensystem von Kugel S1.

Σ3:

7 P.

(c) Wie groß ist der Abstand der Mittelpunkte der Kugeln S2 und S3 gemessen in Weltkoordinaten?

7 P.

Σ4:

Aufgabe 3

Gegeben ist folgender Ausschnitt aus einem WebGL-Javascript, wobei die in der Lehrveranstaltung vorgestellten Hilfsfunktionen verwendet werden:

mat4() *„erzeugt eine 4x4 Einheitsmatrix“*,
mult(m1, m2) *„berechnet das Matrixprodukt der Matrizen m1 und m2“*,
transpose(m1) *„transponiert die Matrix m1“*,
inverse(m1) *„invertiert die Matrix m1“*,
rotate(alpha, [x,y,z]) *„erzeugt eine 4x4 Rotationsmatrix um die Achse* $(x,y,z)^T$ *um den Winkel alpha“*,
translate(x,y,z) *„erzeugt eine 4x4 Translationsmatrix für den Translationsvektor* $(x,y,z)^T$*“*,
scale(sx,sy,sz) *„erzeugt eine 4x4 Skalierungsmatrix für die Skalierungsfaktoren* (s_x,s_y,s_z)*“*,
perspective(fov, aspect, near, far) *„erzeugt eine Projektionsmatrix“*

```
// Projektionsmatrix
var projection = perspective(120.0, 1.0, 0.5, 200.0);

// Zeile A: hier die Model-Matrix mA anlegen

var mA =




// Zeile B: hier die View-Matrix mB anlegen

var mB =




// Zeile C: hier Matrix mC anlegen, die Objektkoordinaten in Clipping-Koordinaten umrechnet

var mC =


// Zeile D: hier Matrix mD anlegen, die Normalen von Objektkoordinaten in Kamerakoordinaten
// umrechnet

var mD =
```

3 P. (a) Ergänzen Sie das Programm nach Zeile A so, dass alle Modelle zuerst um 4 in x-Richtung skaliert und dann um -60° um die y-Achse um den Punkt P(1,2,3) gedreht werden.

4 P. (b) Ergänzen Sie das Programm nach Zeile B so, dass entsprechend lookAt(6,5,4,12,5,4,0,0,-1) die Kamera positioniert wird (verwenden Sie dabei nur die oben angegebenen Hilfsfunktionen)

2 P. (c) Ergänzen Sie das Programm nach Zeile C so, dass eine Matrix mC angelegt wird, die Vertices von Objektkoordinaten in Clipping-Koordinaten umrechnet

Σ5:

(d) Ergänzen Sie das Programm nach Zeile D so, dass eine Matrix mD angelegt wird, die Normalen von Objektkoordinaten in Kamerakoordinaten umrechnet

2 P.

(e) Wie ändert sich das Bild, wenn perspective(120.0, 1.0, 0.5, 200.0) abgeändert wird in: perspective(80.0, 80.0, 0.5, 800.0); ?

3 P.

(f) Ergänzen Sie den unten stehenden GLSL Vertex-Shader und Fragement-Shader möglichst einfach, um Gouraud-Shading zu realisieren. Der dabei ermittelte Farbwert soll danach noch in einen Grauwert mit der gleichen Helligkeit umgerechnet werden.

```
void main(){  // Vertex-Shader
        uniform mat4 matrix;     //  Matrix zur Umrechnung von Objekt- in Clippingkoordinaten
        attribute vec4 color;       // die dem Vertex zugeordnete Farbe
        attribute vec4 position;  //  die dem Vertex zugeordnete Position





}

void main() {  // Fragment-Shader






}
```

5 P.

(g) In GLSL gibt es das Schlüsselwort discard. Warum darf es nicht in einem Tesselation-Shader verwendet werden?

3 P.

Σ6:

Aufgabe 4

Gegeben ist ein Dreieck D mit den Koordinaten A(3, 3, 3), B(0, 0, 0) und C(1, 2, 3). Die Fläche liegt wie z.B. in VRML üblich im Umlaufsinn links.

(a) Wie lautet eine Flächennormale des Dreiecks?

3 P.

(b) Am Punkt L(5, 5, 1) befindet sich eine Punktlichtquelle. Welche weiteren Parameter benötigt man neben L und der Normalen von D, um die spekulare Reflektion eines Punktes in D nach Phong zu berechnen? Welche Bedeutung haben diese Parameter?

3 P.

(c) Eine Beleuchtungsrechnung ergibt, dass an Punkt A die diffuse Reflektion die Farbe (60°, ½, 1) und an Punkt B die Farbe (240°, ½, 1) vorliegt (Farbangaben sind im HLS-Farbsystem). Wie lautet die Farbe (im RGB-Farbsystem) am Punkt P(1,1,1), wenn Gouraud-Shading verwendet wird?

3 P.

Aufgabe 5

(a) Nennen Sie zwei Vorteile von NURBS gegenüber Bezier-Splines

1. ______________________________

2 P. 2. ______________________________

Σ7:

(b) Nennen Sie zwei Anwendungen von Quaternionen in der Computergrafik

2 P.

(c) Wie berechnet OpenGL aus Viewkoordinaten die normalisierten Gerätekoordinaten? Worauf wird das Viewing-Frustum dabei abgebildet?

3 P.

(d) Wozu dient der Bresenham-Algorithmus und welchen besonderen Vorteil bietet er gegenüber Verfahren, welche dieselbe Aufgabe übernehmen können?

2,5 P.

(e) Gegeben ist folgender Ausschnitt eines GLSL – Shaders:

```
vec4 v = vec4(1.0, 2.0, 3.0, 4.0);
vec4 u = vec4(5.0, 6.0, 7.0, 8.0);
v = u.aaab;
v.q = u.t;
```

2 P. Welchen Wert hat v nach Ausführung der letzten Zeile? v = (_____, ______, ______, _____)

Σ8: