Klausur Computergraphik (SS 2016)

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:	22.Juli 2016

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 **35 Punkten.**

Es wurden _____ Punkte erreicht.

Note, Handzeichen:

Aufgabe 1

Gegeben ist die Bezier-Kurve Q(t), $t \in [0,1]$ mit den Stützpunkten A(0,0,0), B(0,2,2), C(-2,0,0) und D(4,0,0).

(a) Berechnen Sie den Punkt Q(0,5) mit dem Algorithmus von deCasteljau.

4 P.

(b) An dem Kurvenpunkt D soll eine weitere Bezier-Kurve R(t), $t \in [0,1]$ angeschlossen werden, so dass ein C^1-stetiger Übergang entsteht und die Kurve in Punkt A mit Tangente $(0,0,-1)^T$ endet. Wie lauten die Stützpunkte von R?

4 P.

(c) Der aus Q und R gebildete Bezier-Spline soll durch drei Bezier-Kurven S_1, S_2 und S_3 ersetzt werden, welche die exakt gleiche Kurvenform haben und die G^1-stetig ineinander übergehen. Wie lauten die Stützpunkte von S_1, S_2 und S_3?

3 P.

Σ2:

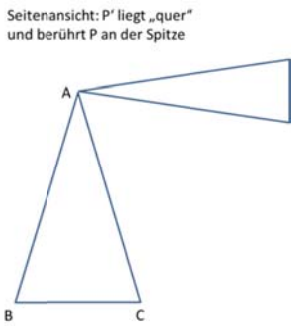
Aufgabe 2

Gegeben ist folgende VRML-Szene:

```
DEF T1 Transform{
  children [
    DEF S1 Shape{ geometry IndexedFaceSet{ coord Coordinate {
                                                    point [  0 0 0,
                                                             1 1 4,
                                                             2 0 0,
                                                             2 2 0,
                                                             0 2 0 ] } # Coordinate

                                          coordIndex [ 4 0 2 3 -1 1 3 4 -1 1 0 4 -1 1 0 2 -1 _____ ]
                                } # IndexedFaceSet
    } # S1
    DEF T2 Transform{




      children [


    ] } # T2
] } # T1
```

4 P. (a) Das IndexFaceSet soll eine Pyramide P mit quadratischer Grundfläche beschreiben. Korrigieren Sie das Feld coordIndex und ergänzen Sie ggf. fehlende Flächen.

(b) Berechnen Sie die Normale, die bzgl. der Pyramide nach außen zeigt, auf dem durch die Eckpunkte A(1, 4, 4), B(2,2,0) und C(0,2,0) definierten Dreieck D.



3 P. (c) Die Spitze P soll eine Pyramide P‘ berühren, welche die gleiche Höhe, aber eine nur halb so große quadratische Grundfläche wie P hat. Die Grundflächen von P und P‘ stehen senkrecht zueinander und die Grundfläche von P‘ befindet sich auf der Seite von Punkt C (vgl. Skizze oben). Ergänzen Sie dazu den Transformnode T2. Benutzen Sie dabei den DEF-USE-Mechanismus.

4 P.

Σ3:

Aufgabe 3

Gegeben ist folgender Ausschnitt aus einem WebGL-Javascript, wobei die in der Lehrveranstaltung vorgestellten Hilfsfunktionen verwendet werden:
mat4() *„erzeugt eine 4x4 Einheitsmatrix“*,
transpose(M) *„transponiert die Matrix M“*,
inverse(M) *„invertiert die Matrix M“*

Außerdem werden die folgenden neuen Hilfsfunktionen verwendet:
rotateX_NEW(alpha, M) *„erzeugt eine 4x4 Matrix R für die Rotation um die x-Achse um Winkel alpha und berechnet R * M“*,
rotateX_NEW(alpha, M) *„erzeugt eine 4x4 Matrix R für die Rotation um die x-Achse um Winkel alpha und berechnet R * M“*,
translate_NEW(x,y,z, M) *„erzeugt eine 4x4 Translationsmatrix T für den Translationsvektor $(x,y,z)^T$ und berechnet T * M “*,

```
// Model-Matrix berechnen
var model = mat4();
model = rotateX_NEW(45.0, model);
model = inverse(model);
model = translate_NEW(1.0, 2.0, 3.0, model);
model = rotateZ_NEW( 45.0, model);

// View-Matrix berechnen
var view = mat4();
view = translate_NEW(0.0, 0.0, 3.0, view);
view = rotateX_NEW(180.0, view);
view = rotateZ_NEW(90.0, view);
```

(a) Der Punkt P hat die Objektkoordinaten P(1,1,1). Wie lauten seine Weltkoordinaten? Geben Sie eine Formel aus nicht ausmultiplizierten Matrizen und Vektoren an.

3 P.

(b) Wie lauten die Kamerakoordinaten von Punkt P? Geben Sie eine Formel aus nicht ausmultiplizierten Matrizen und Vektoren an.

4 P.

Σ4:

(c) Geben Sie möglichst wenige VRML-Transformnodes an, welche die gleiche Transformation in Weltkoordinaten beschreiben wie die obige Matrix model.

3 P.

(d) Wie ändern sich die Objektkoordinaten von P, wenn im obigen Programm die Zeile inverse(model) durch transpose(model) ersetzt wird (Begründung angeben)?

2 P.

(e) Welcher lookAt-Befehl erzeugt die gleiche View-Matrix wie die Matrix view im obigen Programm?

4 P.

Σ5:

Aufgabe 4

Gegeben ist folgender Vertex-Shader in GLSL:

```
void main(){  // Vertex-Shader
    uniform mat4 view;              // View-Matrix
    uniform mat4 model;             // Model-Matrix
    uniform mat4 projection;        // Projektionsmatrix
    uniform vec4 lightPos;          // Position einer Punktlichtquelle in Weltkoordinaten (homogene Koord.)
    uniform vec3 lightIntensity;    // Intensität der Punktlichtquelle (für RGB)
    attribute vec3 diffuseColor;    // die dem Vertex zugeordnete Farbe (für RGB)
    attribute vec4 normal;          // die dem Vertrex zugeordnete Normale in Weltkoordinaten
    attribute vec4 position;        // die dem Vertex zugeordnete Position in Objektkoordinaten
}
```

(a) Der Vertex-Shader oben ist unvollständig, weil eine unbedingt notwendige Programmzeile fehlt. Wie lautet diese Zeile?

2 P.

(b) Schreiben Sie eine Zeile GLSL-Code, welche die 3D Koordinaten von position einer Variablen p zuordnet:

vec3 p =

2 P.

(c) Wie ändert sich das zum Shader gehörige WebGL-Javascript, wenn statt „uniform vec4 lightPos;“ die Zeile „attribute vec4 lightPos;“ im Shader steht?

2 P.

(d) Im Vertex-Shader soll eine Phong-Beleuchtungsrechnung durchgeführt werden und zwar nur für den diffusen Teil. Die Formel lautet $C_{diff} = I_{diff} \cdot R_{diff} \cdot \cos \alpha$. Was ist die Bedeutung des Winkels α und wie kann der $\cos \alpha$ berechnet werden?

2 P.

(e) Welcher Code ist in den Vertex-Shader dazu aufzunehmen? Hinweis: Denken Sie daran, dass das berechnete Ergebnis auch der weiteren Renderpipeline zur Verfügung stehen soll.

4 P.

Σ6:

(f) Welche Codezeilen müssen im Shader ergänzt werden, wenn neben dem diffusen Licht auch das ambiente Licht bei der Beleuchtungsrechnung berücksichtigt werden soll?

2 P.

(g) Welcher Lichtanteil aus dem Phong-Modell ist bisher nicht berücksichtigt? Welche Informationen müssen übergeben werden, um diesen Lichtanteil berechnen zu können?

3 P.

(h) Statt einer Punktlichtquelle soll direktionales Licht für die Beleuchtung verwendet werden. Wie ändert sich dadurch der Vertex-Shader?

3 P.

Aufgabe 5

(a) Was versteht man unter „Aliase“ in der Computergraphik im Zusammenhang mit dem Nyquist-Theorem?

3 P.

Σ7:

(b) Geben Sie ein Beispiel für eine Farbe im RGB-Farbsystem, deren H-Wert im HLS-Farbsystem nicht definiert ist.

2 P.

(c) In welchem Wertebereich liegen normalisierte Gerätekoordinaten in OpenGL?

2 P.

(d) Nennen Sie einen Vorteil von TriangleStrips

2 P.

(e) In VRML soll die Position einer Shape S animiert werden. Welche Routen werden dazu benötigt? Beschreiben Sie dabei auch jeweils den Ausgangspunkt und den Endpunkt der Route.

3 P.

Σ8: