# Klausur Computergraphik (SS 2020)

| | |
|---|---|
| Prüfer: | Prof. Dr. R. Dörner, HS RheinMain |
| Bearbeitungszeit: | 90 min |
| Zugelassene Hilfsmittel: | ein beidseitig handbeschriebenes DIN A4 Blatt, Stifte. (insbesondere Taschenrechner und eigenes Papier ist verboten) |
| Datum: | 17. Juli 2020 |

Name:______________________________ Vorname: ________________________

Matr.-Nr. ___________________

MUSTERLÖSUNG

_________________________________
Unterschrift

Hinweise:

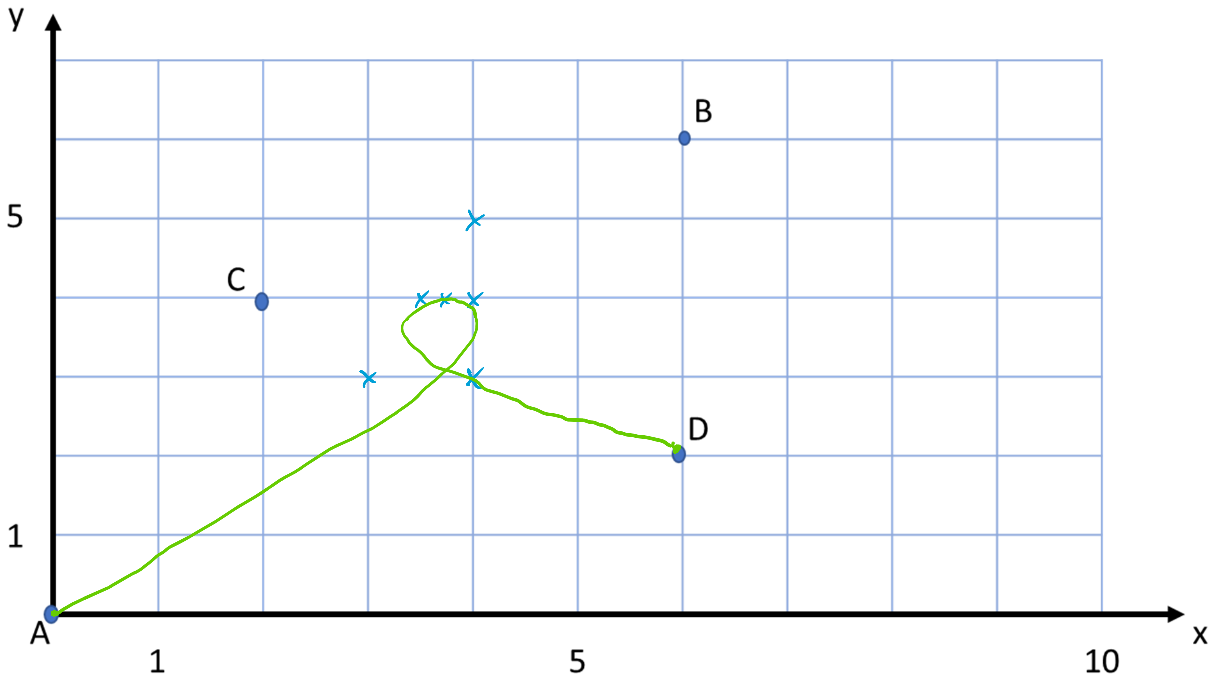
- Überprüfen Sie Ihr Klausurexemplar auf Vollständigkeit (Umfang: 8 Blätter)
- Lösen die Aufgaben im dafür vorgesehenen Raum. Wenn der Platz nicht ausreicht, verwenden Sie die Rückseiten - wenn alle Rückseiten beschrieben sind, fordern Sie ein leeres Blatt bei der Aufsicht an. Schreiben Sie im vorgesehenen 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 **49 Punkten.**

Es wurden __________ Punkte erreicht.

Note, Handzeichen:

## Aufgabe 1

Gegeben ist die Bezier-Kurve Q(t), t∈ [0,1] mit den Stützpunkten A, B, C, D.




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

4 P. Q(0,5) = ( 3,75 , 4 )

4 P. (b) Skizzieren Sie die Q(t)

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

R – erster Stützpunkt: ( 6 , 2 ), R – zweiter Stützpunkt: ( 10 , 0 )

4 P. R – dritter Stützpunkt: ( 7 , 5 ), R – vierter Stützpunkt: ( 6 , 6 )

(d) Der aus Q und R gebildete Bezier-Spline soll durch drei Bezier-Kurven $S_1$, $S_2$ und $S_3$ ersetzt werden, welche die gleiche Kurvenform haben. Wie lauten die Stützpunkte von $S_1$, $S_2$ und $S_3$?

$S_1$ – erster Stützpunkt: ( 0 , 0 ), $S_1$ – zweiter Stützpunkt: ( 3 , 3 )

$S_1$ – dritter Stützpunkt: ( 3,5 , 4 ), $S_1$ – vierter Stützpunkt: ( 3,75 , 4 )

$S_2$ – erster Stützpunkt: ( 3,75 , 4 ), $S_2$ – zweiter Stützpunkt: ( 4 , 4 )

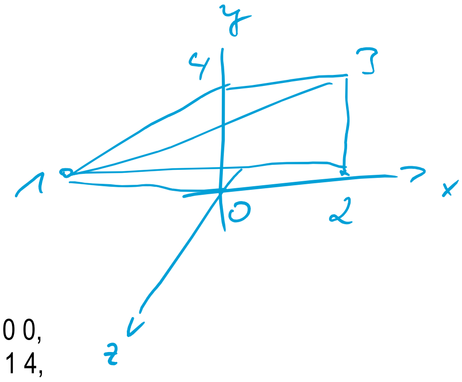
$S_2$ – dritter Stützpunkt: ( 4 , 3 ), $S_2$ – vierter Stützpunkt: ( 6 , 2 )

$S_3$ – erster Stützpunkt: ( 6 , 2 ), $S_3$ – zweiter Stützpunkt: ( 10 , 0 )

10 P. $S_3$ – dritter Stützpunkt: ( 7 , 5 ), $S_3$ – vierter Stützpunkt: ( 6 , 6 )

Σ2:

## Aufgabe 2

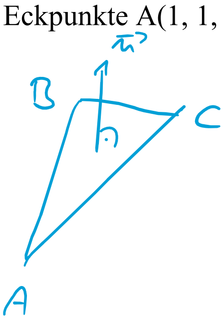



Gegeben ist folgende VRML-Szene:

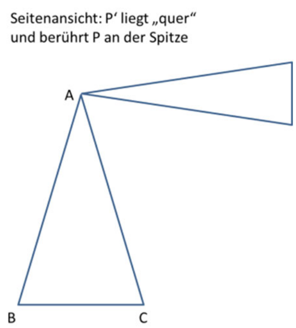
```
DEF T1 Transform{
  children [
    DEF S1 Shape{ geometry IndexedFaceSet{ coord Coordinate {
                                              point [ 0 0 0,
                                                      1 1 4,
                                                      2 0 0,
                                                      2 2 0,
                                                      0 2 0 ] } # Coordinate
                              coordIndex [ 4 0 2 3 -1 1 0 4 -1 1 0 2 -1 1 4 3 -1 ___123___ ]
                                           4320       140      ✓        134
                    } # IndexedFaceSet
    } # S1
   DEF  V1 Viewpoint{    orientation    2 2 0 1.57
                         position       2 0 -1
   } # V1
   DEF T2 Transform{     rotation   0  1  0  -1.57
                         center     1  1  4
                         scale      0,5  0.5  1
      children [
                         USE  S1
      ] } # T2
 ] } # T1
```

5 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, 1, 4), B(0,2,0) und C(2,2,0) definierten Dreieck D.




$$\vec{n} = \vec{AC} \times \vec{AB}$$
$$= (\vec{c} - \vec{a}) \times (\vec{b} - \vec{a})$$
$$= \begin{pmatrix} 1 \\ 1 \\ -4 \end{pmatrix} \times \begin{pmatrix} -1 \\ 1 \\ -4 \end{pmatrix}$$
$$= \begin{pmatrix} 0 \\ 8 \\ 2 \end{pmatrix}$$




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:

(d) Wie weit ist die Spitze der Pyramide P von der Position der Kamera V1 in Kamerakoordinaten entfernt?

$$\vec{P}^{(w)}_{Spitze} = \begin{pmatrix}1\\1\\4\end{pmatrix} \quad \vec{P}^{(v)}_{Kampos} = \begin{pmatrix}0\\0\\0\end{pmatrix} \quad M_{v\to w} = T(2,0,-1)\cdot R_{(1,1,0)}(90°)$$

$$\vec{P}^{(w)}_{Kampos} = M_{v\to w}\cdot\vec{P}^{(v)}_{Kampos} = \begin{pmatrix}2\\0\\-1\end{pmatrix}$$

$$d^{(v)} \underset{\substack{\uparrow\\ \text{keine Skalierung}\\ \text{im } M_{v\to w}}}{=} d^{(w)} = \left|\vec{P}^{(w)}_{Spitze} - \vec{P}^{(w)}_{Kampos}\right| = \left|\begin{pmatrix}1\\1\\4\end{pmatrix} - \begin{pmatrix}2\\0\\-1\end{pmatrix}\right| = \left|\begin{pmatrix}-1\\1\\5\end{pmatrix}\right| = \sqrt{27}$$

3 P.

(e) Wie lauten die Koordinaten des View-Up-Vektors in Kamerakoordinaten?

$$\vec{u}^{(v)} = \begin{pmatrix}0\\1\\0\end{pmatrix}$$

2 P.

(f) Geben Sie eine Formel, die nicht ausmultiplizierte 4×4 Matrizen enthält, zur Berechnung des View-Up-Vektors in Weltkoordinaten an.

$$\begin{aligned}
\vec{u}^{(w)} &= M_{v\to w}\cdot\vec{u}^{(v)}\\
&= T(2,0,-1)\cdot R_{(1,1,0)}(90°)\cdot\vec{u}^{(v)}\\
&= T(2,0,-1)^T\cdot R_z(-45°)\cdot R_y(90°)\cdot R_z(45°)\cdot\vec{u}^{(v)}\\
&= \begin{bmatrix}1&0&0&2\\0&1&0&0\\0&0&1&-1\\0&0&0&1\end{bmatrix}\cdot\begin{bmatrix}\cos-45°&-\sin-45°&0&0\\ \sin-45°&\cos-45°&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\cdot\\
&\quad\begin{bmatrix}0&0&1&0\\0&1&0&0\\-1&0&0&0\\0&0&0&1\end{bmatrix}\cdot\begin{bmatrix}\cos 45°&-\sin 45°&0&0\\ \sin 45°&\cos 45°&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\cdot\begin{pmatrix}0\\1\\0\\0\end{pmatrix}
\end{aligned}$$

5 P.



Σ4:

## Aufgabe 3

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 vec3 lightDirection;    // Richtung einer direktionalen Lichtquelle in Weltkoordinaten
      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. gl_Position = projection * view * model * position;

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

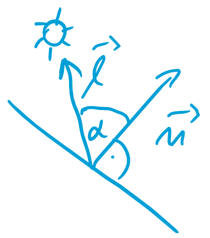
2 P. vec3 p = (1.0 / gl_Position.w) * gl_Position.xyz

(c) Warum ist „uniform vec3 lightDirection;" der Deklaration „attribute vec3 lightDirection;" vorzuziehen?

lightDirection hat für alle Vertices den gleichen Wert => Übergabe mit uniform ist möglich
(und erwünscht, da einfacher)

3 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 α berechnet werden?




$\alpha \hat{=} -$ lightDirection

3 P. $\cos\alpha = \frac{\langle \vec{l}, \vec{n} \rangle}{|\vec{l}| \cdot |\vec{n}|} \overset{\arccos}{\Longrightarrow} \alpha$

(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.

```
varying vec3 dColor;
dColor = lightIntensity * diffuseColor *
         max(0.0, dot(normalize(-lightDirection),
                      normalize(normal.xyz)));
```

4 P.



Σ5:

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

uniform vec3 ambientColor;

dColor = dColor + ambientColor;

2 P.

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

spekulares Licht

zu übergeben: - spek. Intensität d. Lichtquelle
- spek. Reflektionskoeffizient
- Shininess

4 P.

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

"- lightDirection" ist zu ersetzen durch:

"lightPos.xyz - pos.xyz" wobei

"vec4 pos = model * position;"

3 P.

### Aufgabe 4

Gegeben ist ein Quaternion $\hat{q} = (\cos 30°, \frac{\sqrt{2}}{2} \cdot \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix})$

(a) Wie lautet das zu $\hat{q}$ gehörige Einheitsquaternion (Hinweis: sin30° = ½)?

$(\cos 30°, \sin 30° \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix})$

3 P.

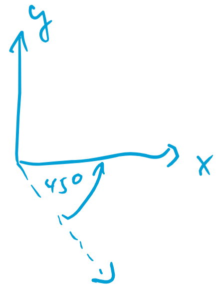
Σ6:

(b) Welche Rotation R wird durch das zu $\hat{q}$ gehörige Einheitsquarternion repräsentiert?

$$R_{(1,-1,0)}(15°)$$

3 P.

(c) Wie lauten die Eulerwinkel (Konvention: Reihenfolge x, y, z), welche die gleiche Rotation wie R erzeugen (es kann auch eine Reihe von Eulerwinkeln angegeben werden)?




1.) $(15°, 0°, 45°)$

2.) $(0°, 0°, -45°)$

$$R_z(-45°) \cdot R_x(15°) \cdot R_z(45°)$$

3 P.

(d) Geben Sie eine Formel aus nicht ausmultiplizierten 4×4 Matrizen an, mit denen die Drehung R in homogenen Koordinaten berechnet werden kann

$$M_R = \begin{bmatrix} \cos -45° & -\sin -45° & 0 & 0 \\ \sin -45° & \cos -45° & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 15° & -\sin 15° & 0 \\ 0 & \sin 15° & \cos 15° & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\cdot \begin{bmatrix} \cos 45° & -\sin 45° & 0 & 0 \\ \sin 45° & \cos 45° & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4 P.

## Aufgabe 5

(a) Erklären Sie den Begriff „Gimbal Lock“. Wie kann man einen Gimbal Lock vermeiden?

Verlust Freiheitsgrad bei Rotation mit Eulerwinkel, wenn eine Rotationsachse auf eine andere Rotationsachse gedreht wird.

3 P.

Lösung: Verwendung von Quaternionen



Σ7:

3 P.

(b) Was versteht man in der Computergrafik unter „BRDF“?

Bidirectional Reflectance Distribution Function:
Funktion gibt an, wie Licht aus einem bestimmten
Raumwinkel kommend auf andere Raumwinkel
durch Reflektion verteilt wird.

3 P.

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

$(u, v)$ mit $-1 \leq u \leq 1$
$-1 \leq v \leq 1$

3 P.

(d) Nennen Sie zwei Unterschiede zwischen der Reflektionsgleichung und der Renderinggleichung.

1. Emittiertes Licht wird berücksichtigt
2. Rekursive Formulierung

3 P.

(e) Was versteht man unter „View Frustum Culling“?

Ausschließen aller Objekte vom Rendering,
die außerhalb des Sichtvolumens liegen

3 P.

(f) Warum kann man mit dem Phong-Beleuchtungsmodell keinen Schatten berechnen?

Phong ist lokales Beleuchtungsmodell:
nur 1 Punkt wird betrachtet.
Schattenberechnung benötigt aber Betrachtung
von mind. 2 Punkten.



Σ8: