Karlsruher Institut für Technologie

Institut für Visualisierung und Datenanalyse
Lehrstuhl für Computergrafik
Prof. Dr.-Ing. Carsten Dachsbacher

Hauptklausur Computergrafik
WS 2021/2022

9.3.2022

Name	
Matrikelnummer	

Beachten Sie:

- Die Klausur umfasst 39 Seiten (12 Blätter) mit 10 Aufgaben.
- Es sind **keine Hilfsmittel** zugelassen.
- Sie haben **90 Minuten** Bearbeitungszeit.
- Schreiben Sie Ihre Matrikelnummer oben auf jedes bearbeitete Aufgabenblatt.
- Schreiben Sie Ihre Lösungen auf die Aufgabenblätter. Bei Bedarf können Sie weiteres Papier anfordern.
- Streichen Sie nicht zu bewertende Lösungen durch.
- Wir akzeptieren auch englische Antworten.

Aufgabe	1	2	3	4	5	6	7	8	9	10	**Gesamt**
Erreichte Punkte											
Erreichbare Punkte	17	6	17	22	15	15	12	47	15	14	**180**

Note

Aufgabe 1: Farbe und Perzeption (17 Punkte)

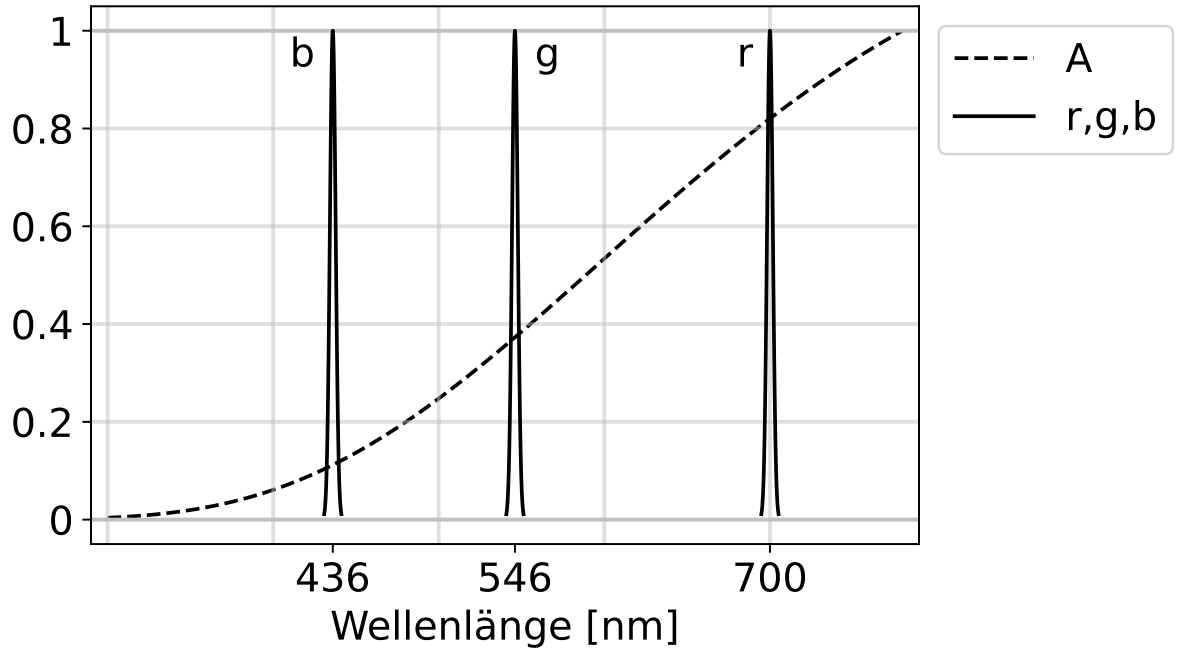


Abbildung 1: Spektrum A und Color-Matching Funktionen r, g, b des CIE RGB-Farbraums.

a) Die Abbildung zeigt Spektrum A als gepunktete Linie. Der CIE RGB Farbraum hat die drei monochromatischen Primärfarben mit Wellenlängen $r = 700nm, g = 546nm$ und $b = 436nm$ (oben als Dirac-Deltas skizziert). Geben Sie den *ungefähren* CIE RGB-Wert von Spektrum A an! **(3 Punkte)**

$(R, G, B) =$

Matrikelnummer: _____

b) Zeichnen Sie in die Abbildung das Spektrum eines wahrnehmbaren Farbeindrucks ein, der *nicht* von einem CIE RGB Monitor dargestellt werden kann! Begründen Sie, warum dies der Fall ist! Wo in der echten Welt (egal ob in der Natur oder technisch erzeugt) könnte ein solches Lichtspektrum auftreten? **(5 Punkte)**

c) Ein Monitor mit minimaler Intensität I_{min} und maximaler Intensität I_{max} soll ein Bild mit 8 Bit je Farbkanal darstellen. Dabei soll der Pixelwert 0 mit Intensität $I(0) = I_{min}$ und der Pixelwert 255 mit Intensität $I(255) = I_{max}$ dargestellt werden. Sie können annehmen, dass Menschen Helligkeitsunterschiede von bis zu 2% nicht wahrnehmen.

1) Auf welche Intensität $I(1)$ darf der Pixelwert 1 maximal abgebildet werden, sodass der Helligkeitsunterschied zwischen $I(0)$ und $I(1)$ für Menschen nicht wahrnehmbar ist? **(3 Punkte)**

$I(1) \leq$

2) Die Pixelwerte $n = 1, \ldots, 254$ sollen monoton so auf Intensitäten $I(n)$ abgebildet werden, dass der Helligkeitsunterschied aufeinanderfolgender Pixelwerte $n, n+1$ mit Intensitäten $I(n), I(n+1)$ $(n = 0, \ldots, 254)$ für Menschen nicht wahrnehmbar ist.

Gibt es eine solche Abbildung für $I_{min} = 10$ und $I_{max} = 1000$? Begründen Sie!

Tipp: Sie dürfen folgende Näherungen verwenden:

x	1.001	1.002	1.005	1.01	1.02	1.05	1.1	1.2	1.5	2
$\log_{10} x$	1/2300	1/1160	1/460	1/230	1/116	1/45	1/24	1/13	1/6	1/3

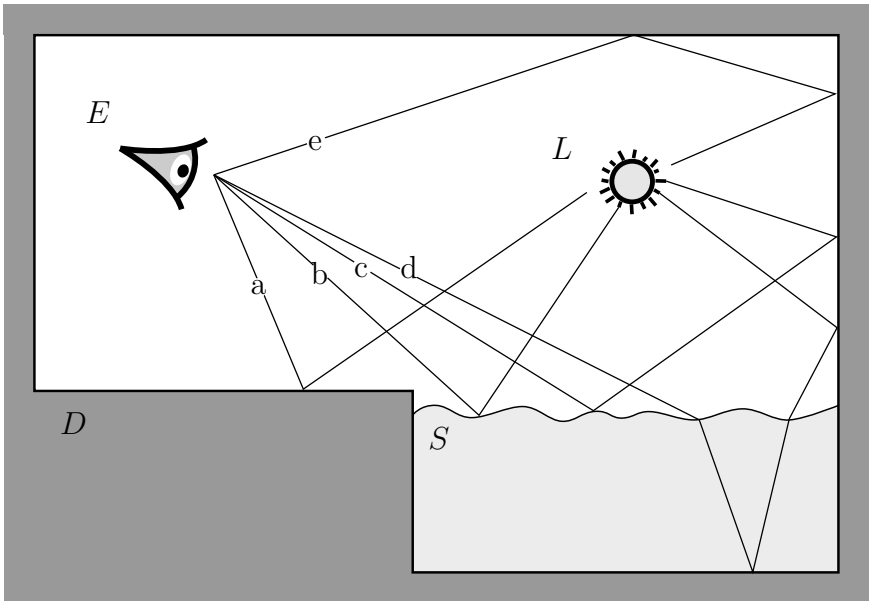
(6 Punkte)

Matrikelnummer: _____

Aufgabe 2: Whitted-Style Raytracing (6 Punkte)

Die folgende Abbildung zeigt eine Szene mit einer diffusen Oberfläche D ($k_d > 0$, $k_a = k_s = k_t = k_r = 0$) und eine spekulare Wasseroberfläche S ($k_a = k_d = k_s = 0$, $k_t > 0$, $k_r > 0$). Die Szene wird von einer Punktlichtquelle L beleuchtet und durch eine Lochkamera E beobachtet. Der Brechindex des Wassers ist höher als der Brechindex der umgebenden Luft. Die Abbildung zeigt außerdem Strahlfolgen (a) bis (e), die die Kamera E mit dem Licht L verbinden.

Geben Sie an, welche der Strahlfolgen *nicht* durch Whitted-Style Raytracing erzeugt werden, und begründen Sie stichpunktartig, warum diese nicht erzeugt werden können! **(6 Punkte)**



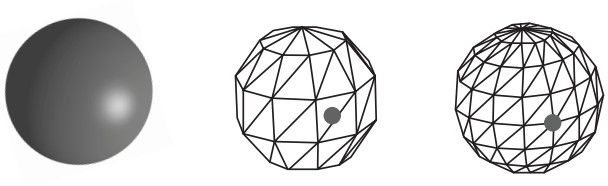
Aufgabe 3: Phong-Beleuchtungsmodell (17 Punkte)

a) Wie lautet die Formel des Phong-Beleuchtungsmodells? Markieren und benennen Sie die einzelnen Summanden nach dem modellierten Phänomen! **(4 Punkte)**

Matrikelnummer: _____

b) Fassen Sie in einem Satz zusammen, wie Gouraud-Shading funktioniert! **(3 Punkte)**

c) Die in der Abbildung gezeigte analytische Kugel (links) wird jeweils grob (mitte) und fein (rechts) tesseliert. Die Position des Glanzlichts aus der analytischen Berechnung ist mit einem Punkt für beide Tesselierungen markiert. Beschreiben Sie kurz, wie sich in den beiden Fällen die Tesselierung auf die Darstellung des Glanzlichts bei Gouraud-Shading auswirkt! Begründen Sie Ihre Antworten! **(5 Punkte)**

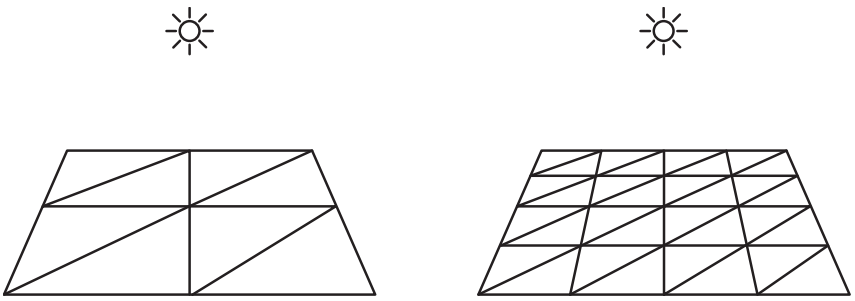


Mitte:

Rechts:

Matrikelnummer: _____

d) Eine *unverschattete diffuse* Ebene wird mit einer Punktlichtquelle beleuchtet und mit dem Phong-Beleuchtungsmodell ($k_a = k_s = 0, k_d > 0$) und Gouraud-Shading dargestellt. Warum hängt das Ergebnisbild vom Tesselierungsgrad der Ebene ab? **(5 Punkte)**



Aufgabe 4: Räumliche Datenstrukturen (22 Punkte)

a) Gegeben sei eine Szene mit N Dreiecken. Geben Sie die erwartete Zeitkomplexität abhängig von N in $\mathcal{O}$-Notation an für das Finden des nächsten Schnittpunktes entlang eines Strahls

1) ...falls keine Beschleunigungsstruktur zur Verfügung steht! **(1 Punkt)**

2) ...falls der Schnitttest durch eine Hüllkörperhierarchie (BVH) beschleunigt wird und kein pathologischer Fall vorliegt! **(1 Punkt)**

Matrikelnummer: _____

b) Nennen Sie eine Datenstruktur, bei deren Aufbau die Surface Area Heuristik (SAH) eingesetzt werden kann, und eine, bei der dies nicht der Fall ist! **(2 Punkte)** ☐

SAH kann eingesetzt werden:

SAH kann nicht eingesetzt werden:

c) Sie sollen eine BVH traversieren, um möglichst effizient die Entfernung zum ersten Schnitt mit einem Strahl zu bestimmen. Strahlen starten immer außerhalb von allen Hüllkörpern (AABBs). Ergänzen Sie die Methode `intersectBVH`, die initial mit einem Zeiger auf den Wurzelknoten der BVH und einem Strahl aufgerufen wird! **(18 Punkte)**

```
typedef struct {
    BVHNode *left, *right; // Zeiger auf Kindknoten, NULL falls keine Kinder
    std::vector<Primitive> primitives; // Liste von Primitiven falls Blattknoten
} BVHNode;

// liefert Entfernung zum nahsten Schnitt mit der AABB eines Knotens.
// FLOAT_MAX wenn kein Schnitt existiert.
float intersectAABB(BVHNode *n, Ray *r);

// liefert Entfernung zum nahsten Schnitt mit Primitiven in einem Blattknoten.
// FLOAT_MAX wenn kein Schnitt existiert.
float intersectPrimitives(BVHNode *n, Ray *r);

// vertauscht 2 Zahlen oder 2 Zeiger
SWAP(x,y)

// Sie können auch min(x,y) und max(x,y) verwenden

...


// Soll möglichst effizient Entfernung zum nahsten Schnitt zurückgeben
float intersectBVH(BVHNode *n, Ray *r)
{
    // ... falls Blattknoten:









    // ... falls kein Blattknoten:
    float t1 = FLOAT_MAX, t2 = FLOAT_MAX;
    BVHNode *n1 = n->left, *n2 = n->right;









}
```

Matrikelnummer: _____

Aufgabe 5: Transformationen (15 Punkte)

Diese Aufgabe führt schrittweise zu einer Transformationsmatrix, die einen dreidimensionalen Punkt um eine Achse rotiert. Die Achse geht durch den Punkt $\mathbf{p} = (p_1, p_2, p_3)^T \in \mathbb{R}^3$ und zeigt in Richtung $\mathbf{w}$. Außerdem gegeben sind die zwei Einheitsvektoren $\mathbf{u}, \mathbf{v}$, die zu $\mathbf{w}$ und zueinander orthogonal sind, d.h. $(\mathbf{u}, \mathbf{v}, \mathbf{w})$ ist eine orthonormale Basis des $\mathbb{R}^{3\times3}$.

Hinweis: Sie dürfen vorherige Matrizen und deren Inversen und Transponierten verwenden und müssen Matrixprodukte nicht auswerten. Sie dürfen Vektoren und geringerdimensionale Matrizen direkt als Matrixeinträge verwenden.

a) Geben Sie eine homogene Transformationsmatrix $T_p \in \mathbb{R}^{4\times4}$ an, die einen dreidimensionalen Punkt in homogenen Koordinaten um $\mathbf{p}$ verschiebt! Geben Sie außerdem die inverse Transformation $T_p^{-1} \in \mathbb{R}^{4\times4}$ an! **(3 Punkte)**

$T_p =$ $T_p^{-1} =$

b) Geben Sie eine Transformationsmatrix $M \in \mathbb{R}^{3\times3}$ an, die einen dreidimensionalen Punkt $p \in \mathbb{R}^3$ aus karthesischen Koordinaten in die Basis $(\mathbf{u}, \mathbf{v}, \mathbf{w})$ transformiert! Geben Sie außerdem die inverse Transformation M^{-1} an! **(4 Punkte)**

$M =$ $M^{-1} =$

c) Geben Sie eine *homogene* Transformationsmatrix $M_b \in \mathbb{R}^{4\times4}$ an, die einen dreidimensionalen Punkt in das Koordinatensystem mit Ursprung $\mathbf{p}$ und Basis $(\mathbf{u}, \mathbf{v}, \mathbf{w})$ transformiert! **(4 Punkte)**

$M_b =$

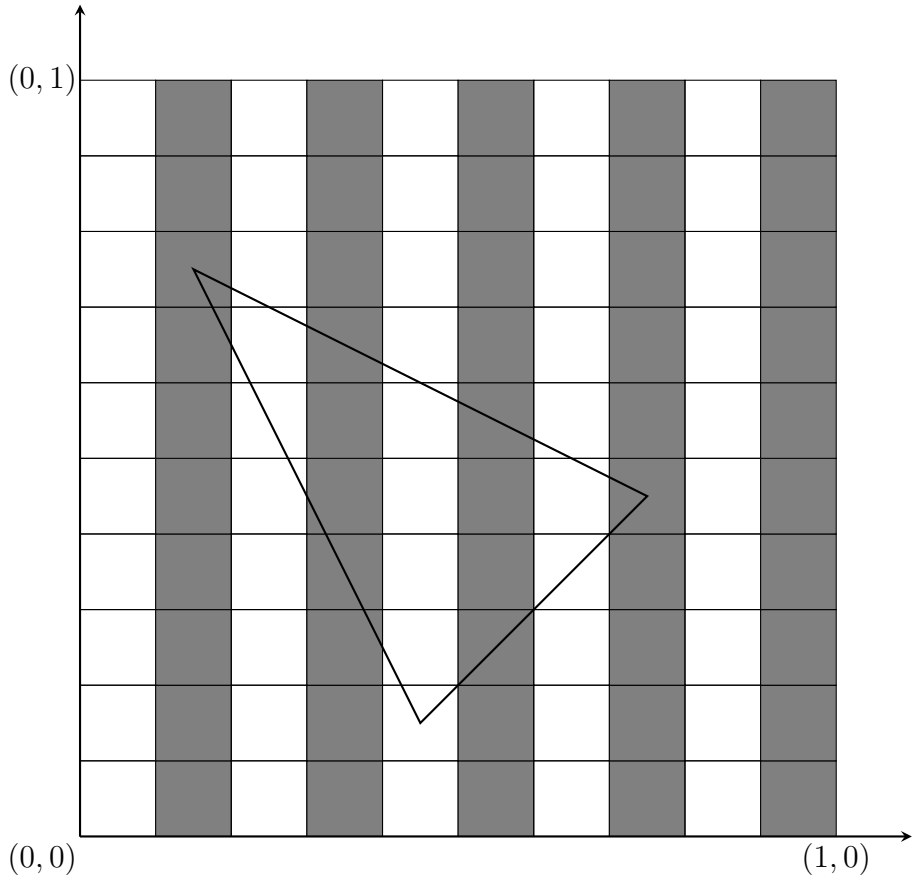
d) Sei $R_z(\theta) \in \mathbb{R}^{4\times4}$ die homogene Rotationsmatrix, die einen dreidimensionalen Punkt um den Winkel θ um die z-Achse $(0, 0, 1)$ rotiert. Geben Sie die Matrix M_L an, die einen Punkt um den Winkel θ um die Achse L mit Stützpunkt $\mathbf{p}$ und Richtung $\mathbf{w}$ rotiert! **(4 Punkte)**

$M_L =$

Matrikelnummer: _____

Aufgabe 6: Texturen und Baryzentrische Koordinaten (15 Punkte)

Gegeben sei die folgende 10×10 Textur (weiß entspricht einem Texel-Wert von 1, grau entspricht 0.4):



a) Eingezeichnet sind die Texturkoordinaten $(0.15, 0.75)$ (oben), $(0.45, 0.15)$ (unten) und $(0.75, 0.45)$ (rechts) eines Dreiecks. Bestimmen Sie die Position des Texturzugriffs mit baryzentrischen Koordinaten $(1/3, 1/3, 1/3)$ und zeichnen Sie die Position in die Skizze ein! Geben Sie den Wert des Zugriffs mit bilinearer Interpolation an! **(4 Punkte)**

b) Nun ändert sich die Texturkoordinate des rechten Dreiecksknotens um 0.05 nach links, also von $(0.75, 0.45)$ zu $(0.70, 0.45)$. Ändert sich der ausgelesene Texturwert an den baryzentrischen Koordinaten $(1/3, 1/3, 1/3)$, falls die Textur mit bilinearer Interpolation ausgelesen wird? Wenn ja, wie? Begründen Sie Ihre Antwort! **(4 Punkte)**

c) Nun soll die Textur als Displacement Map verwendet werden. Welche Auswirkungen hat die Verwendung von Mip Maps in diesem speziellen Anwendungsfall? **(4 Punkte)**

☐ d) Was versteht man unter anisotroper Texturfilterung? **(3 Punkte)**

Matrikelnummer: _____

Aufgabe 7: OpenGL-Pipeline (12 Punkte)

a) Gegeben ist eine Szene mit mehreren Dreiecken, die durch eine Punktlichtquelle beleuchtet werden. Die Position der Punktlichtquelle ist in Weltkoordinaten gegeben. In diesen Koordinaten soll auch das verwendete Lambertsche Beleuchtungsmodell mit $k_d = 0.5$ ausgewertet werden. Die Darstellung erfolgt mittels Phong Shading und OpenGL.

1) In welcher Pipeline-Stufe wird das Beleuchtungsmodell ausgewertet? **(1 Punkt)**

2) In welcher Pipeline-Stufe sollte die Koordinatentransformation von Objekt- in Weltkoordinaten angewandt werden, um eine möglichst effiziente Laufzeit zu erreichen? **(1 Punkt)**

3) Der Vertex Shader erhält als Eingabe die Vertexposition in Objektkoordinaten. Welche Ausgaben (`out`-Variablen) muss der Vertex Shader erzeugen? Welche Eingaben bekommt der Fragment Shader über `in`-Variablen und `uniform`-Variablen? Beschreiben Sie jede Variable kurz und nennen Sie einen sinnvollen Datentyp! **(6 Punkte)**

Vertex Shader

`out`-Variablen:

- Vertex Position in Clip-Koordinaten `vec4`
-
-

Fragment Shader

`uniform`-Variablen:

-

`in`-Variablen:

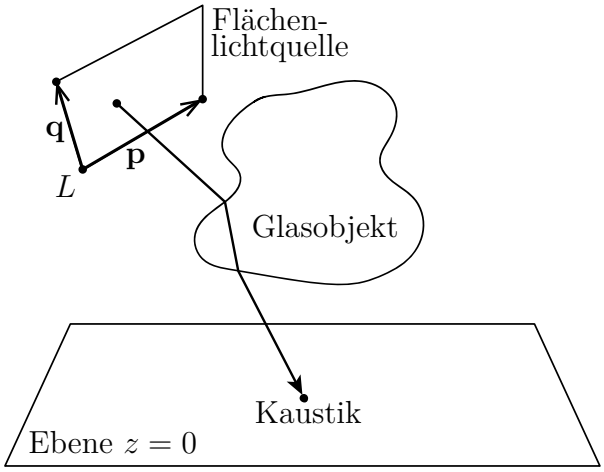
-
-

Matrikelnummer: _____

b) In welcher Stufe der OpenGL-Pipeline findet die Interpolation von Vertexattributen statt? Handelt es sich in modernem OpenGL dabei um eine fixe oder frei programmierbare Stufe? **(2 Punkte)** ☐

c) Nennen Sie zwei Fixed Function-Operationen, die nach der Rasterisierung, aber vor dem Schreiben des Framebuffers ausgeführt werden! **(2 Punkte)** ☐

Aufgabe 8: GLSL-Shader (47 Punkte)



Anmerkung: Sie können die einzelnen Teilaufgaben unabhängig voneinander lösen.

Gegeben ist eine Szene mit einer Kamera, einer Flächenlichtquelle, einem Boden und einem Glasobjekt. Das Glasobjekt wirft eine Kaustik auf den Boden, die mit OpenGL-Shadern dargestellt werden soll. Dafür werden virtuelle Photonen an der Flächenlichtquelle gestartet und deren Ausbreitung durch die Szene mittels Sphere Tracing berechnet. Danach wird die Energie jedes Photons durch additives Blending im Bildraum akkumuliert.

- Das Glasobjekt und der Boden werden durch ein gemeinsames vorzeichenbehaftetes Distanzfeld (negative Werte im Inneren) beschrieben.
- Der Boden liegt in der xy-Ebene.
- Die Lichtquelle ist definiert durch ihren Aufpunkt in der Ecke L und davon ausgehende aufspannende Vektoren $\mathbf{p}$ und $\mathbf{q}$ (siehe Abbildung).
- Ein Vertex-Buffer speichert für jedes virtuelle Photon seine Startposition auf der Lichtquelle sowie seine initiale Richtung jeweils in lokalen Koordinaten.

a) Der Vertex Shader erhält die Startposition und -richtung des Photons in lokalen Koordinaten der Lichtquelle, sowie Aufpunkt und aufspannende Vektoren des Lichts. Er soll die Position in Weltkoordinaten ausgeben, an der das Photon nach der Traversierung der Szene den Boden schneidet (falls ein Schnittpunkt existiert):

```
in vec2 phPos; // lokale Koordinaten des Photons bzgl. p und q. 0<=phPos.xy<= 1
in vec3 phDir; // lokale Richtung, -1 < phDir.xy < 1, phDir.z > 0
uniform vec3 L; // Ursprung der Lichtquelle
uniform vec3 p,q; // aufspannende Vektoren der Lichtquelle

... // auszufüllende Methoden

void main()
{
    vec3 worldPos = photonPosToWorld(phPos, L, p, q);
    vec3 worldDir = photonDirToWorld(phPos, L, p, q);
    gl_Position = photonTracing(worldPos, worldDir);
}
```

Vervollständigen Sie die dafür notwendigen folgenden Funktionen!

Matrikelnummer: _____

1) Transformieren Sie die Photonenposition in Weltkoordinaten! **(4 Punkte)**

```
vec3 photonPosToWorld(vec2 phPos, vec3 L, vec3 p, vec3 q)
{




}
```

2) Transformieren Sie die Photonenrichtung in Weltkoordinaten! **(4 Punkte)**

```
vec3 photonDirToWorld(vec3 phDir, vec3 L, vec3 p, vec3 q)
{




}
```

3) Vervollständigen Sie `sphereTrace(P,d)`, sodass die Funktion den nächsten Schnittpunkt eines Strahls mit Start `P` und Richtung `d` mit der Szene in `P` schreibt. Das Distanzfeld kann mit `distField(vec3 P)` abgefragt werden. Ein Schnittpunkt liegt vor, falls das Distanzfeld einen Wert kleiner `epsilon` zurückliefert. Nach 1000 Schritten wird abgebrochen. **(8 Punkte)**

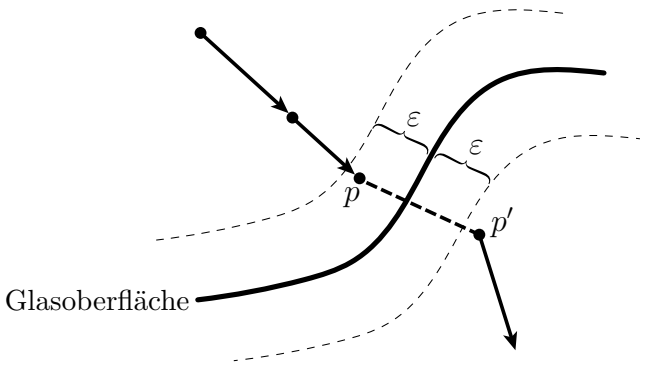
```
float distField(vec3 x); // wertet das Distanzfeld für Position x aus
vec3 distFieldGradient(vec3 x); // Gradient des Distanzfelds an Position x

const float epsilon = 0.1;
int steps = 0; // Abbruch nach insgesamt 1000 Schritten

bool sphereTrace(inout vec3 P, vec3 d)
{









    return steps < 1000;
}
```

4) Geben Sie in `photonTracing(P,d)` die Position aus, an der ein Photon mit Startposition `p` und -richtung `d` (nach eventuellen Lichtbrechungen) den Boden schneidet! Dafür schneidet `sphereTrace` das Photon mit der Geometrie der Szene. Sie können in `isInside` speichern, ob Sie sich innerhalb oder außerhalb des Glasobjekts befinden.

   - Falls ein Schnitt mit dem Glasobjekt gefunden wurde, berechnen Sie die neue, gebrochene Richtung des Photons in Abschnitt 1 mit `refract(d, normale, eta1, eta2)` für den Übergang von einem Medium mit Brechindex `eta1` in ein Medium mit Brechindex `eta2` (`normale` zeigt zum Medium mit `eta1`).
   - Passen Sie die Startposition des neuen Strahls an, um einen erneuten Schnittpunkt mit der selben Stelle zu vermeiden, indem Sie in Abschnitt 2 die aktuelle Position entlang des Gradienten um `2*epsilon` verschieben (siehe Skizze)!

(12 Punkte)

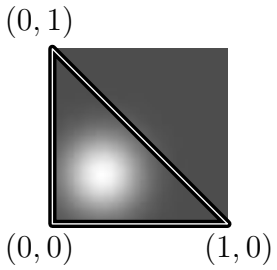
```
vec4 photonTracing(vec3 startPos, vec3 startDir)
{
    bool isInside = false;
    float etaLuft = 1.0;
    float etaGlas = 1.3;
    vec3 P = startPos;
    vec3 d = normalize(startDir);

    while(1)
    {
        bool success = sphereTrace(P,d);

        // Kein Schnitt mit Boden nach 1000 Schritten?
        if (!success) return vec4(0);

        // Boden geschnitten?
        if (abs(P.z) < epsilon) return vec3(P.yz,0.0);

        // ======== Abschnitt 1: ========
        // Berechnen Sie die Normale und die gebrochene Strahlrichtung
        vec3 n =









        // ======== Abschnitt 2: ========
        // Verschieben Sie die aktuelle Position p nach p'
```




```
        isInside = !isInside;
    }
}
```

b) Nun sind die Auftreffpunkte der Photonen auf dem Boden (sofern vorhanden) bekannt: Der Vertex Shader hat sie in Weltkoordinaten in `gl_Position` ausgegeben. Vervollständigen Sie den Geometry Shader, sodass dieser jedes Photon als Punktprimitiv entgegennimmt und ein texturiertes Dreieck erzeugt! Ein Eckpunkt des Dreiecks ist die Photonenposition mit Texturkoordinate $(0, 0)$, und das Dreieck soll sowohl in Welt- als auch Texturkoordinaten die Kantenvektoren $(1, 0, 0)$ und $(0, 1, 0)$ haben. Verwerfen Sie Photonen, für die kein Schnittpunkt mit dem Boden gefunden wurde! **(9 Punkte)**

```
layout (points) in;
layout (triangle_strip, max_vertices=3) out;

// aufspannende Vektoren in Weltkoordinaten der texturierten Fläche:
const vec4 u = (1,0,0,0);
const vec4 v = (0,1,0,0);

out vec2 texCoord; // Ausgabe: Texturkoordinate = Koordinaten des
                   // Dreiecks in der xy-Ebene

void main()
{
    vec4 P = gl_in[0].gl_Position;
    // ergänzen Sie hier den Code, um das Dreieck auszugeben
```




```
    EndPrimitive(); // Primitiv ist fertig => ausgeben
}
```

c) Der finale Ablauf besteht aus zwei Draw-Calls.

- Im ersten Draw-Call wird je Photon ein texturiertes Dreieck gezeichnet. Die Beiträge der einzelnen Photonen sollen durch additives Blending akkumuliert werden.
- Im zweiten Draw-Call wird die Oberflächenfarbe des texturierten Bodens gezeichnet. Dabei soll die Blending-Einstellung so sein, dass das Resultat die Oberflächenfarbe multipliziert mit den akkumulierten Photonenbeiträgen ist.

☐ Wie müssen Sie dafür die Blending-Einstellungen für die beiden Draw-Calls wählen? **(4 Punkte)**

Sie dürfen die Buchstaben als Abkürzung für folgende Parameter verwenden:

A	`GL_ONE`	E	`GL_SRC_COLOR`
B	`GL_ZERO`	F	`GL_DST_COLOR`
C	`GL_SRC_ALPHA`	Y	`GL_FUNC_ADD`
D	`GL_DST_ALPHA`	Z	`GL_FUNC_MULT`

Draw-Call 1 (Photonen):

glBlendEquation( )

glBlendFunc(,)

Draw-Call 2 (Boden):

glBlendEquation( )

glBlendFunc(,)

d) Nehmen Sie an, der Boden ist keine unendlich große Ebene. Stattdessen sind die Pixel, die zum Boden gehören, im Stencil Buffer mit dem Wert 1 markiert. Wie beschränkt man das Blending auf diesen Bereich? **(6 Punkte)**

Tipps:

- Reihenfolge der Parameter:

  `glStencilFunc(stencilfunc, refstencil, bitmask)`

  `glStencilOp(fail, zfail, zpass)`

- Mögliche Vergleichsfunktionen:

  `GL_ALWAYS, GL_NEVER, GL_LESS, GL_EQUAL, GL_NOT_EQUAL`

- Mögliche Masken: `0x00, 0x01, 0x02, ..., 0xff`

- Mögliche Stencil-Operationen:

  `GL_KEEP, GL_ZERO, GL_INCR, GL_DECR, GL_INVERT, GL_REPLACE`

```
glStencilFunc(             ,             ,             );

glStencilOp(               ,             ,             );
```

Aufgabe 9: Prozedurale Modellierung (15 Punkte)

a) Einfaches 3D Lattice Value Noise verwendet vorberechnete Zufallszahlen in einem 3D Gitter. Nennen Sie jeweils einen Vorteil und einen Nachteil dieses Verfahrens gegenüber der Auswertung eines Zufallszahlengenerators bei jedem Zugriff! **(3 Punkte)**

Vorteil:

Nachteil:

b) Perlin Noise verwendet Hash-Funktionen, um auf eine Tabelle mit Zufallszahlen zuzugreifen. Wie findet dieser Zugriff statt? Welche Vorteile bietet dieses Verfahren? **(4 Punkte)**

c) Die Abbildungen zeigen vier prozedurale Texturen, die auf der Basis von Noise-Funktionen berechnet wurden, wie Sie sie in der Vorlesung kennengelernt haben. Geben Sie für jede Textur an, ob

- die Zufallswerte linear oder kubisch interpoliert wurden
- eine Noise-Funktion ($k = 0$ Oktaven) oder eine Turbulenz-Funktion ($k > 0$ Oktaven) verwendet wurde
- welche der folgenden Verknüpfungen verwendet wurde

$$a(\mathbf{x}) = \sum_k \left(\tfrac{1}{f}\right)^k n(f^k \mathbf{x})$$

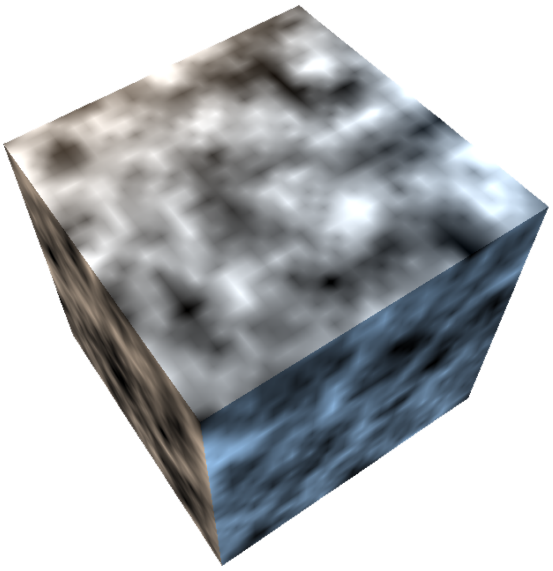
$$b(\mathbf{x}) = \cos\left(x + \sum_k \left(\tfrac{1}{f}\right)^k \left|n(f^k \mathbf{x})\right|\right)$$

$$c(\mathbf{x}) = \sum_k \left(\tfrac{1}{f}\right)^k \left|n(f^k \mathbf{x})\right| \text{ mit } -1 \leq n(\mathbf{x}) \leq 1$$

$$d(\mathbf{x}) = a(\mathbf{x} + s \cdot a(\mathbf{x})) \text{ mit } s \in \mathbb{R}^+$$

(8 Punkte)

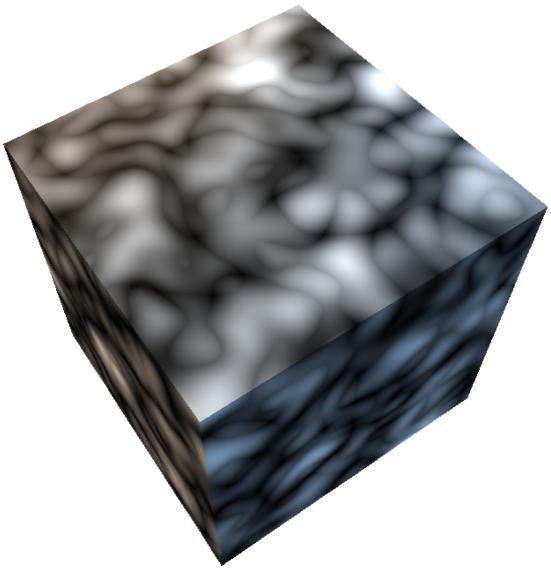
Beispiel: *kubisch, k* [>] *0, a(**x**)*



Interpolation:

Oktaven: k [ ] 0

Verknüpfung:

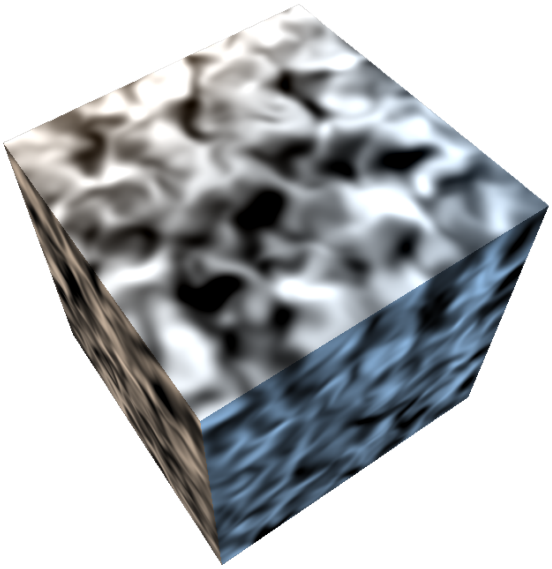


Interpolation:

Oktaven: k [ ] 0

Verknüpfung:

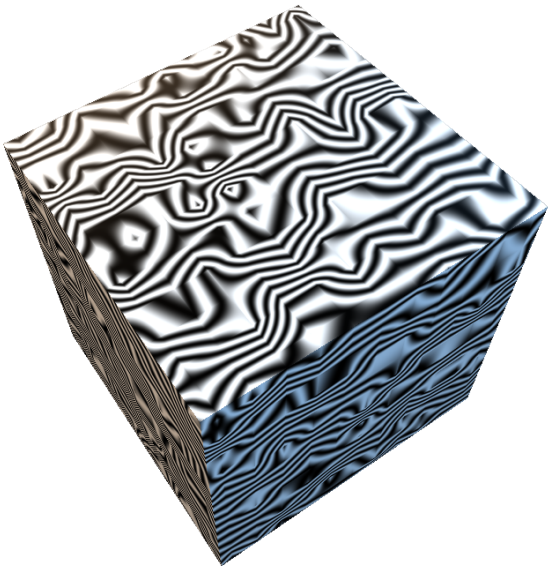
Matrikelnummer: _____



Interpolation:

Oktaven: k ☐ 0

Verknüpfung:



Interpolation:

Oktaven: k ☐ 0

Verknüpfung:

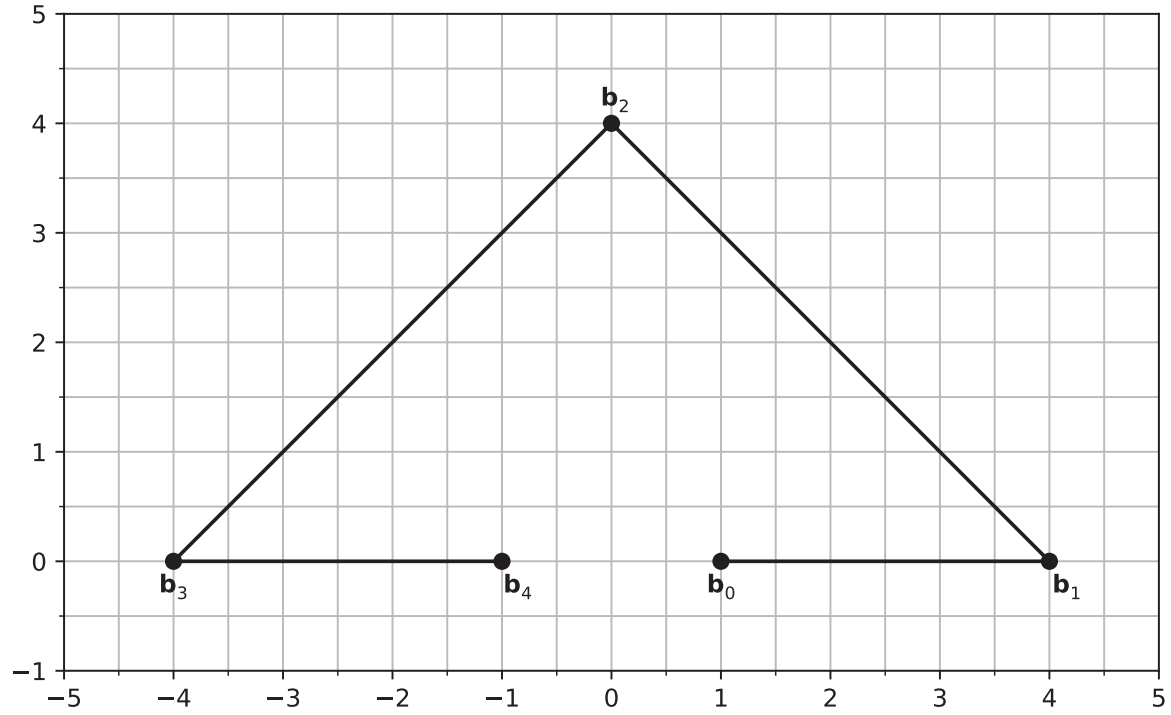
Aufgabe 10: Bézier-Kurven (14 Punkte)

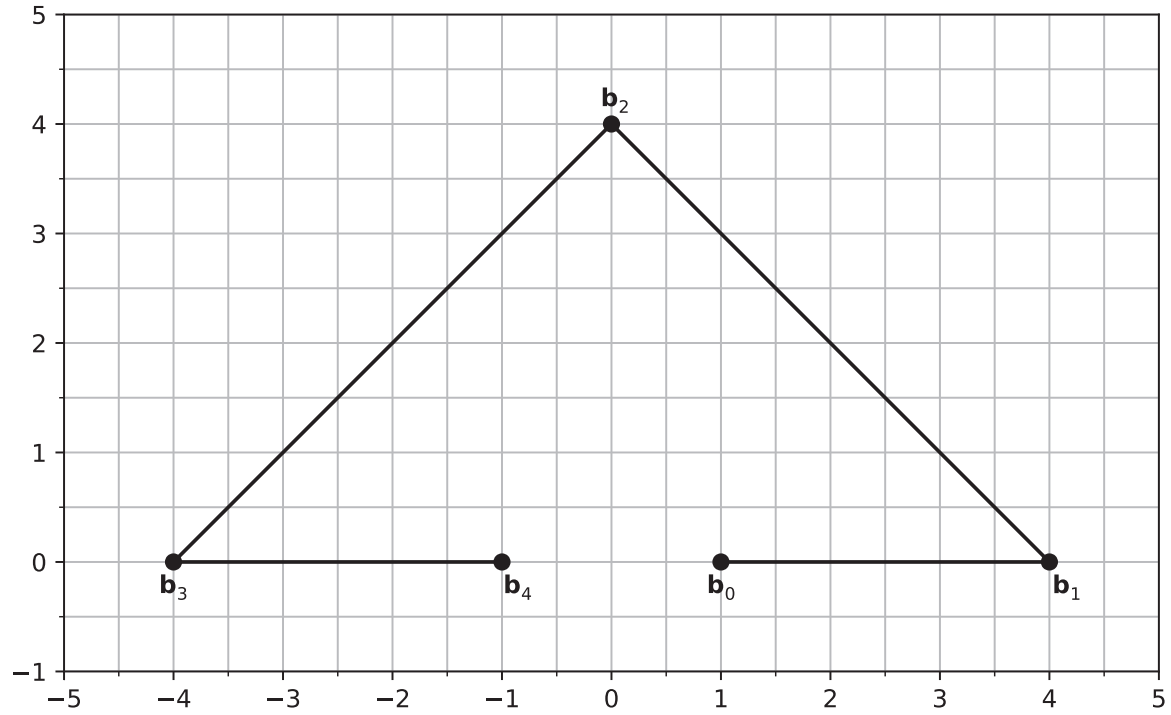
a) Gegeben sei die Beziér-Kurve $F(u) = \sum_{i=0}^{4} \mathbf{b}_i B_i^4(u)$ vom Grad 4 mit $u \in [0, 1]$ und

$$\mathbf{b}_0 = (0, 0), \quad \mathbf{b}_1 = (4, 0), \quad \mathbf{b}_2 = (0, 4), \quad \mathbf{b}_3 = (-4, 0), \quad \mathbf{b}_4 = (-1, 0).$$

Unterteilen Sie die Kurve *zeichnerisch* einmal in der parametrischen Mitte und nutzen Sie die Kontrollpolygone der Teilkurven, um $F(u)$ unter Ausnutzung der Eigenschaften von Bézier-Kurven zu skizzieren!

Zur Korrektur können Sie die zweite Zeichnung verwenden. Kennzeichnen Sie eindeutig, welche Zeichnung bewertet werden soll! **(7 Punkte)**





Matrikelnummer: _____

b) Gegeben sei die kubische Beziér-Kurve $F(u) = \sum_{i=0}^{3} \mathbf{b}_i B_i^3(u)$

$$\mathbf{b}_0 = (1,0), \qquad \mathbf{b}_1 = (0,0), \qquad \mathbf{b}_2 = (0,1), \qquad \mathbf{b}_3 = (1,1).$$

Eine zweite kubische Beziér-Kurve $G(u) = \sum_{i=0}^{3} \mathbf{c}_i B_i^3(u)$ soll bei $\mathbf{b}_3$ C^2-stetig anschließen und mit $F(u)$ eine geschlossene Linie ergeben. Bestimmen Sie zeichnerisch oder rechnerisch die Kontrollpunkte von $G(u)$!

Zur Korrektur können Sie die zweite Zeichnung verwenden. Kennzeichnen Sie eindeutig, welche Zeichnung bewertet werden soll! **(7 Punkte)**