Karlsruher Institut für Technologie

Institut für Betriebs- und Dialogsysteme
Lehrstuhl für Computergrafik
Prof. Dr.-Ing. Carsten Dachsbacher

Nachklausur Computergrafik WS 2011/12

Freitag, 13. April 2012

Kleben Sie hier **nach Bearbeitung der Klausur** den Aufkleber hin.

Beachten Sie:

- Trennen Sie vorsichtig die dreistellige Nummer von Ihrem Aufkleber ab. Sie sollten sie gut aufheben, um später Ihre Note zu erfahren.
- Die Klausur umfasst 16 Seiten (8 Blatt) mit 7 Aufgaben.
- Es sind **keine Hilfsmittel** zugelassen.
- Vor Beginn der Klausur haben Sie 5 Minuten Zeit zum *Lesen* der Aufgabenstellungen. Danach haben Sie **60 Minuten** Bearbeitungszeit.
- Schreiben Sie Ihren Namen und Ihre Matrikelnummer oben auf jedes bearbeitete Aufgabenblatt.
- Schreiben Sie Ihre Lösungen auf die Aufgabenblätter. Bei Bedarf können Sie weiteres Papier anfordern.
- Wenn Sie bei einer Multiple-Choice-Frage eine falsche Antwort angekreuzt haben und diesen Fehler korrigieren möchten, füllen Sie die betreffende Box ganz aus: 
  Falsche Antworten führen zu Punktabzug.
  Jede Multiple-Choice-Aufgabe wird mit mindestens 0 Punkten bewertet.
- Kleben Sie **nach Bearbeitung der Klausur** den Aufkleber mit Ihrem Namen und Ihrer Matrikelnummer auf dieses Deckblatt.

Aufgabe	1	2	3	4	5	6	7	**Gesamt**
Erreichte Punkte								
Mögliche Punkte	16	17	16	18	18	21	14	**120**

Note

Name: _____ Matrikelnummer: _____

Aufgabe 1: Beleuchtung (16 Punkte)

a) Welche 4 Vektoren werden benötigt, um die reflektierte Farbe an einem Vertex bzw. Oberflächenpunkt mit dem Phong-Beleuchtungsmodell zu berechnen? Illustrieren Sie Ihre Antwort mit einer Skizze, die den Oberflächenpunkt, die Betrachter- und die Lichtquellenposition beinhaltet. **(4 Punkte)**

b) Beim Phong-Beleuchtungsmodell setzt sich das reflektierte Licht aus 3 Komponenten zusammen: *ambient*, *diffus* und *spekular*.

Erläutern Sie kurz, *welche* dieser Komponenten ihren Wert *aus welchem Grund* ändert bzw. ändern, wenn:

- der Vertex verschoben wird, aber die Lichtquelle- und Betrachterposition unverändert bleibt? **(3 Punkte)**

- sich die Betrachterposition ändert, aber Lichtquelle und Vertex unverändert bleiben? **(3 Punkte)**

Name: _____ Matrikelnummer: _____

c) Welche Aufgabe hat der Phong-Exponent? Welche Auswirkungen hat es, wenn sein Wert größer bzw. kleiner gewählt wird? **(3 Punkte)**

d) Warum verbessert eine Unterteilung von Primitiven in kleinere die Darstellungsqualität, wenn Gouraud-Shading verwendet wird? **(3 Punkte)**

Name: _____ Matrikelnummer: _____

Aufgabe 2: Blending (17 Punkte)

a) Blending in OpenGL wird wie folgt konfiguriert:

```
glClearColor(1.0, 1.0, 1.0, 1.0);
glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
glEnable(GL_BLEND);
glBlendEquation(GL_FUNC_ADD);
```

I. Wie ist der endgültige *Rotwert* eines Fragments, das: **(2 Punkte)**

1. mit `glClear` gelöscht wurde, danach
2. von einem Fragment mit Farbe $RGBA = [0.2, 0.6, 0.8, 0.5]$, danach
3. von einem Fragment mit Farbe $RGBA = [0.4, 0.2, 0.3, 0.5]$ beschrieben wurde?

II. Wie ist der endgültige *Rotwert* eines Fragments, das: **(2 Punkte)**

1. mit `glClear` gelöscht wurde, danach
2. von einem Fragment mit Farbe $RGBA = [0.4, 0.2, 0.3, 0.5]$, danach
3. von einem Fragment mit Farbe $RGBA = [0.2, 0.6, 0.8, 0.5]$ beschrieben wurde?

III. Welche Konsequenzen hat dies für das Zeichnen von sich überlappenden semi-transparenten Objekten? **(2 Punkte)**

Name: _____ Matrikelnummer: _____

b) Ein Zaun wird mit OpenGL gezeichnet, indem ein `GL_QUAD`-Primitiv mit einer Textur gezeichnet wird, die den Wert 1 enthält, wo sich Lücken im Zaun befinden, und sonst den Wert 0 enthält. Warum sollte man in diesem Fall Alpha-Testing statt Blending verwenden? **(5 Punkte)**

c) Die Methode `DrawScene()` zeichnet die Szene mit geeigneten Shader-Programmen, so dass sie durch *eine* Lichtquelle beleuchtet wird.

Wie können Sie mit Hilfe von `DrawScene()` und Blending die Beleuchtung mit mehreren Lichtquellen durchführen, ohne den Shader-Programmcode zu verändern? Geben Sie genau an, welche Einstellungen Sie für das Blending und den Tiefenpuffer vornehmen müssen.

(Sie können textuell, in Pseudocode oder mit Hilfe von OpenGL-Befehlen antworten.) **(6 Punkte)**

Name: _____ Matrikelnummer: _____

Aufgabe 3: Texturierung (16 Punkte)

a) Erklären Sie kurz,

(i) welches Problem Mip-Mapping löst,

(ii) was für Vorbereitungen für eine Textur getroffen werden,

(iii) wie der Zugriff auf die Textur erfolgt.

(6 Punkte)

b) Erläutern Sie kurz, wofür Environment-Mapping verwendet wird. Welche Annahmen werden dabei getroffen? **(6 Punkte)**

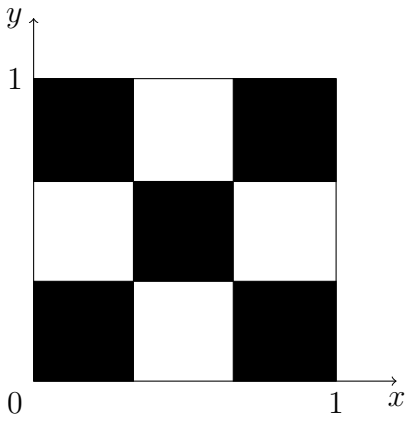
Name: _____ Matrikelnummer: _____

c) Die folgenden Zeilen eines OpenGL-Programms zeichnen ein `GL_QUAD`-Primitiv, auf das eine Textur aufgebracht wird:

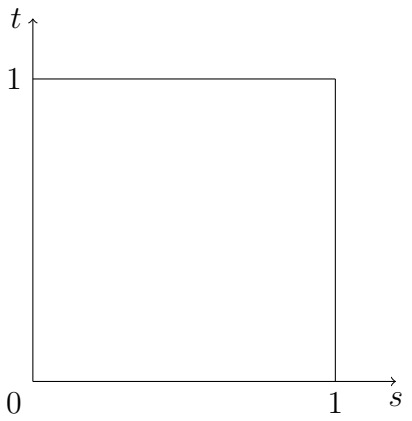
```
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);

glBegin(GL_QUADS);
    glTexCoord2f(0.0, 1.0);
    glVertex2f(0.0, 0.0);

    glTexCoord2f(1.5, 1.0);
    glVertex2f(1.0, 0.0);

    glTexCoord2f(1.5, -0.5);
    glVertex2f(1.0, 1.0);

    glTexCoord2f(0.0, -0.5);
    glVertex2f(0.0, 1.0);
glEnd();
```

Die Vorderseite des Quad-Primitives, wie es gezeichnet werden würde, ist auf dem Bild unten dargestellt. Zeichnen Sie die Eingabetextur in Ihrem originalen s/t-Koordinatensystem! **(4 Punkte)**

Ausgabe:



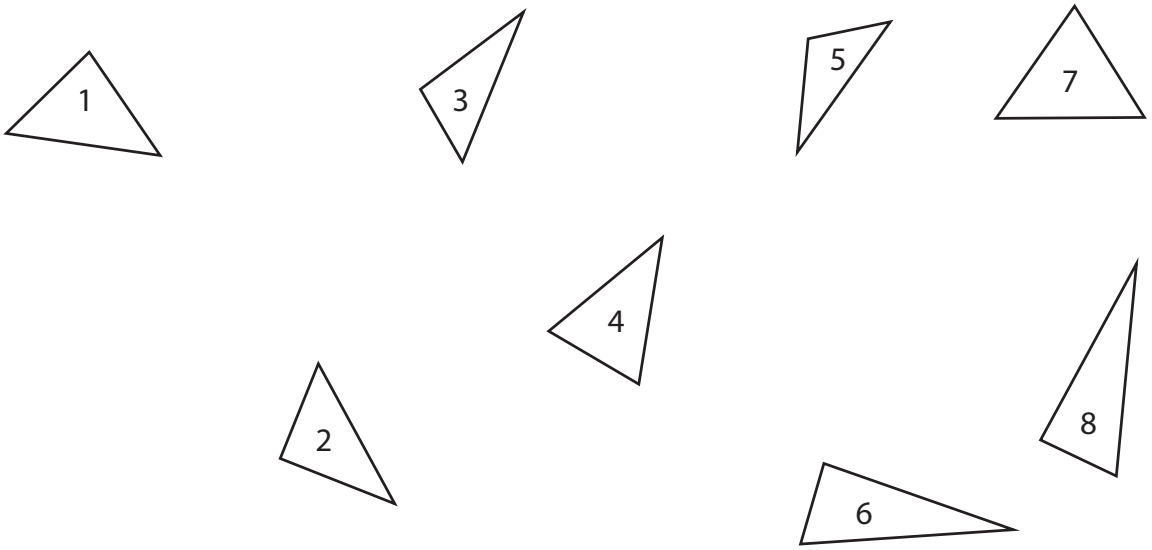
Eingabetextur:



Name: _____ Matrikelnummer: _____

Aufgabe 4: Hierarchische Datenstrukturen (18 Punkte)

a) Konstruieren Sie eine Bounding Volume Hierarchy mit AABBs und dem Unterteilungskriterium *Object-Median* entlang der Achse der größten Ausdehnung. Zeichnen Sie dazu die AABBs aller Bounding Volumes ein. Die Konstruktion soll stoppen, wenn jedes Blatt nur noch 2 Primitive enthält. Geben Sie zusätzlich den zugehörigen Baum mit all seinen Knoten und Blättern an. **(6 Punkte)**



Name: _____ Matrikelnummer: _____

b) Welches Ziel verfolgt man beim Einsatz von Surface Area Heuristics (SAH)? Erklären Sie kurz die Grundidee. Wie ändert sich die Konstruktion eines kD-Baumes bei Anwendung dieser Technik? Welche Größen fließen dabei zusätzlich mit ein? **(6 Punkte)**

c) Warum ist das *Object-Median*-Kriterium meist nicht das performanteste Unterteilungskriterium, wenn eine Bounding Volume Hierarchie beim Raytracing einer statischen Szene eingesetzt werden soll? In welchen Fällen ist das Object-Median-Kriterium *nicht* schlechter als Surface Area Heuristics? **(2 Punkte)**

d) Was ist *mailboxing* im Kontext von räumlichen Datenstrukturen zur Beschleunigung der Schnittberechnungen? Was ist der Vorteil dieser Methode? **(4 Punkte)**

Name: _____ Matrikelnummer: _____

Aufgabe 5: Raycasting mit OpenGL und GLSL (18 Punkte)

Sie sollen einen einfachen Raycaster in OpenGL mit GLSL-Shadern implementieren. Dazu wird ein bildschirmfüllendes Rechteck so gezeichnet, dass Ihr Fragment-Shader für jeden Pixel des Viewports aufgerufen wird. Im Vertex-Shader sind alle vorbereitenden Schritte, wie das Aufspannen der Bildebene und das Generieren der Sichtstrahlen, bereits implementiert. Sie müssen sich also nur noch um das eigentliche Raycasting im Fragment-Shader kümmern. Die Szene besteht aus einem Array mit Kugeln.

In Ihrem Fragment-Shader können Sie auf folgende Variablen zugreifen:

- `vec3 eye` gibt die Position des Betrachters in Weltkoordinaten an.
- `vec3 dir` ist die Strahlrichtung in Weltkoordinaten, die Sie vom Vertex-Shader erhalten. Mit `eye` und `dir` können Sie den Strahl beschreiben, der an der Position des Betrachters startet und durch die korrekte Position auf der Bildebene verläuft.
- `vec4 spheres[10]` ist ein Array mit 10 Kugeln. Jede Kugel wird durch einen vec4 beschrieben, wobei die ersten drei Komponenten ihren Mittelpunkt in Weltkoordinaten angeben, und die vierte Komponente ihren Radius angibt.

Um einen Strahl mit einer Kugel zu schneiden, können Sie folgende Funktion verwenden:

```
vec2 t = intersectRS(vec3 e, vec3 d, vec3 center, float radius);
```

Sie schneidet einen parametrisierten Strahl der Form $\mathtt{r}(t) = \mathtt{e} + t \cdot \mathtt{d}$ mit einer Kugel mit dem Mittelpunkt `center` und dem Radius `radius`.

- Gibt es keine Schnittpunkte, wird `vec2(-1.0, -1.0)` zurückgegeben.
- Gibt es einen Schnittpunkt, wird der entsprechende Parameter in der ersten Komponente von `vec2 t` zurückgegeben, und die zweite Komponente ist `-1.0`.
- Gibt es zwei Schnittpunkte, so werden beide Parameter zurückgegeben, wobei die erste Komponente von `t` den kleineren enthält; in diesem Fall gilt also $\mathtt{t.x} < \mathtt{t.y}$.

Vervollständigen Sie den Fragment-Shader auf der nächsten Seite:

Schneiden Sie den Sichtstrahl mit allen Kugeln und ermitteln Sie den Schnittpunkt, sofern es einen gibt, der am nächsten zum Betrachter liegt, sowie die Normale an diesem Punkt. Vervollständigen Sie die `if`-Abfrage, ob ein Schnittpunkt gefunden wurde. Tragen Sie die Koordinaten und die Normale des Schnittpunkts, der am nächsten zum Betrachter liegt, in die Funktion `computeShading(vec3 coord, vec3 normal)` ein.

Name: _____ Matrikelnummer: _____

```
uniform vec3 eye;            // Position des Betrachters in Weltkoordinaten
in vec3 dir;                 // Strahlrichtung in Weltkoordinaten

// Array mit 10 Kugeln. Jede Kugel wird durch einen vec4 beschrieben;
// erste drei Komponenten: Mittelpunkt; vierte Komponente: Radius.
uniform vec4 spheres[10];

out vec4 fragColor;          // Farbe des Fragments ausgeben

void main()     // Dieser Fragment-Shader wird fuer jeden Pixel
{               // der Bildebene aufgerufen









  if (                                                          )
  { // Schnittpunkt gefunden!




    fragColor = computeShading(                  ,               );

  }
  else
  {
    // Keinen Schnittpunkt gefunden, Hintergrund schwarz setzen
    fragColor = vec4(0.0, 0.0, 0.0, 1.0);
  }
}
```

Name: _____ Matrikelnummer: _____

Aufgabe 6: OpenGL (21 Punkte)

a) Was ist Back-Face-Culling? Wofür verwendet man es? **(2 Punkte)**

b) Welche Aufgabe hat der Vertex-Cache, wie Sie ihn in der Vorlesung kennengelernt haben? Welche Informationen werden im Cache gespeichert und welcher Verarbeitungsschritt in der Grafikpipeline wird dadurch beschleunigt? **(2 Punkte)**

c) Wie muss man Geometrie spezifizieren, damit man den Vertex-Cache nutzen kann? **(2 Punkte)**

d) Welchen Vorteil hat es, ein Dreiecksnetz mit Triangle Strips statt mit Triangle Lists zu zeichnen? **(2 Punkte)**

Name: _____ Matrikelnummer: _____

e) Wenn OpenGL-Shader Programme verwendet werden, um Texture Mapping zu implementieren, was ist die Aufgabe des Vertex Shaders und was die Aufgabe des Fragment Shaders? **(2 Punkte)**

f) Bewerten Sie die folgenden Aussagen, indem Sie *wahr* oder *falsch* ankreuzen. **(5 Punkte)**

Aussage	**Wahr**	**Falsch**
Der Vertex-Shader kann Vertizes löschen und generieren	☐	☐
Der Vertex-Shader kann Vertizes transformieren	☐	☐
Die Sichtrichtung im OpenGL-View-Space ist entlang der negativen Y-Achse	☐	☐
Man kann innerhalb eines Zeichenvorgangs mit `glDrawArrays()` (Draw-Call) zwei verschiedene Shader-Programm-Objekte verwenden	☐	☐
Bei T-Vertizes kommt es zu inkonsistenter Interpolation zwischen den umgebenden Dreiecken	☐	☐

Name: _____ Matrikelnummer: _____

g) Um die klassische OpenGL-Pipeline mit Gouraud-Shading nachzubilden werden u.a. die folgenden Schritte durchgeführt:

A Clipping

B Anwendung der Model-View-Transformation

C Tiefentest

D Auslesen von Texturen

E Anwendung der Projektionstransformation

F Beleuchtungsberechnung

Schreiben Sie die Reihenfolge der Schritte auf und kennzeichnen Sie mit Buchstaben, welche der Operationen Sie in einem Vertex-Shader (V) oder Fragment-Shader (F) implementieren, oder ob der entsprechende Schritt nicht frei programmierbar ist (N). **(6 Punkte)**

Name: _____ Matrikelnummer: _____

Aufgabe 7: Bézierkurven und Bézier-Splines (14 Punkte)

a) Gegeben sei der *quadratische* Bézier-Spline aus den beiden Bezierkurven

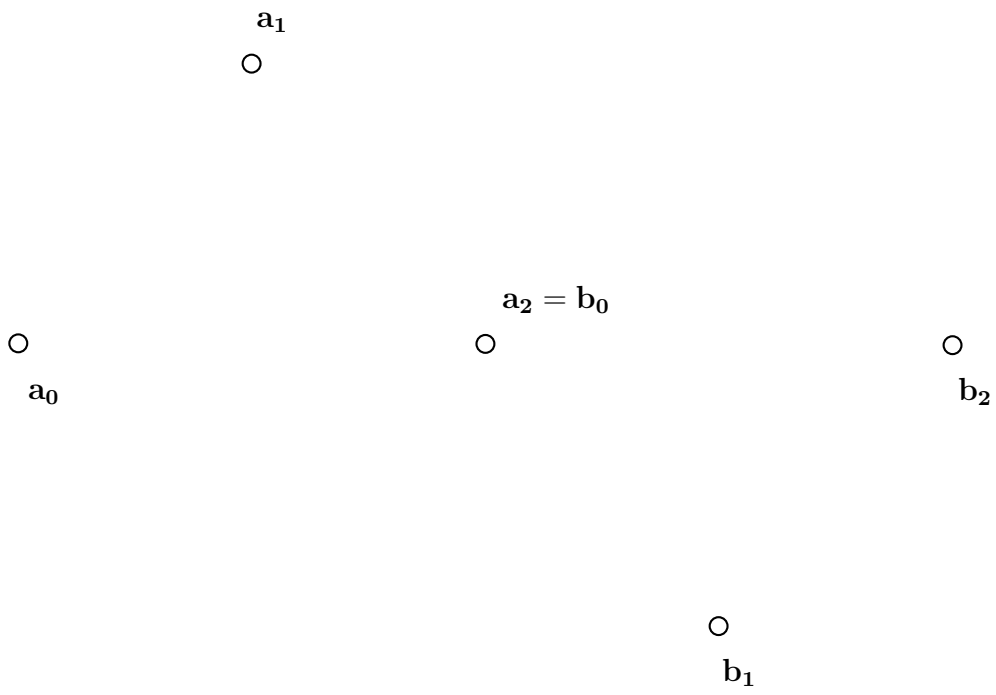
$$\mathbf{a}(u) = \sum_{i=0}^{2} \mathbf{a}_i B_i^n(u) \quad \text{und} \quad \mathbf{b}(u) = \sum_{i=0}^{2} \mathbf{b}_i B_i^n(u)$$

mit den unten dargestellten Kontrollpunkten.

I) Werten Sie die Teilkurven des Bézier-Splines jeweils an der Stelle $u = \frac{1}{2}$ zeichnerisch mit dem *de-Casteljau-Algorithmus* aus. Markieren Sie die beiden Punkte $\mathbf{a}(\frac{1}{2})$ und $\mathbf{b}(\frac{1}{2})$.

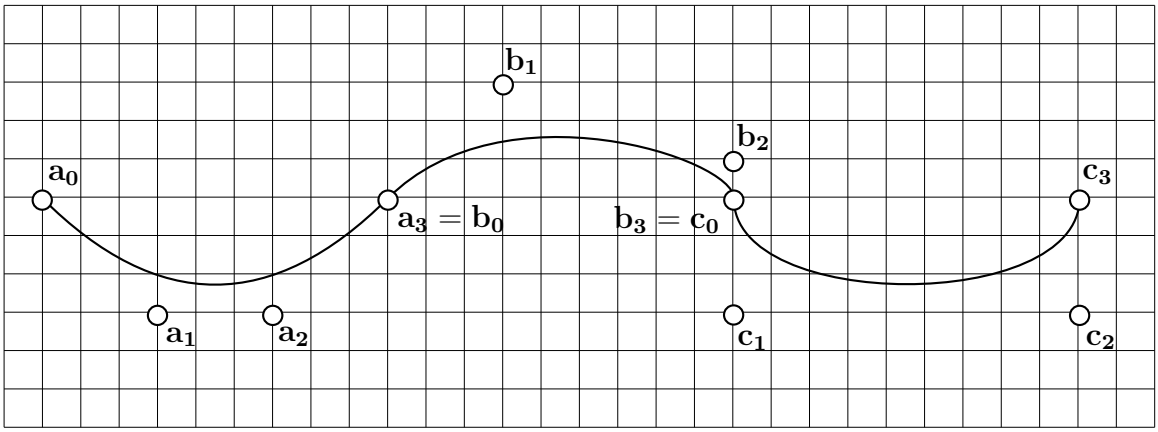
(3 Punkte)

II) Nennen Sie die drei wichtigsten Eigenschaften von Bézier-Kurven, die Ihnen bei der *Skizzierung* der Kurve helfen, und skizzieren Sie den *Kurvenverlauf* des Bézier-Splines. **(4 Punkte)**



Name: _____ Matrikelnummer: _____

b) Gegeben sei der abgebildete *kubische* Bézier-Spline.



I) Was für ein C^i-Übergang liegt an der Stelle $\mathbf{a_3} = \mathbf{b_0}$ vor? Begründen Sie kurz! **(2 Punkte)**

II) Was für ein C^i-Übergang liegt an der Stelle $\mathbf{b_3} = \mathbf{c_0}$ vor? Begründen Sie kurz! **(2 Punkte)**

c) Nennen Sie drei wichtige Eigenschaften der Bernstein-Polynome, die Sie in der Vorlesung kennengelernt haben. **(3 Punkte)**