BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Prof. Dr.-Ing. Hartmut Schirmacher
Fachbereich Informatik und Medien

(7)

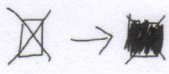
Klausurprüfung Computergrafik 2
Sommersemester 2014, Gruppe 2

(vom Dozenten auszufüllen)

Aufgabe												Summe	von max.	Note
Punkte												56	70	1,7

Bitte beachten Sie unbedingt folgende Hinweise:

- Prüfen Sie die Vollständigkeit der Unterlagen:
  - **9 Blätter** (inkl. Deckblatt) mit insgesamt **14 Aufgaben**.
- Füllen Sie das Deckblatt aus, indem Sie Ihren Namen und Ihre Matrikelnummer in die dafür vorgesehenen Felder schreiben.
- Sollte eines Ihrer Blätter vom Deckblatt getrennt werden, versehen Sie es unbedingt mit Ihrem Namen und Ihrer Matrikelnummer. Wenn Blätter nicht zugeordnet werden können, werden fehlende Blätter mit 0 Punkten bewertet.
- Für Multiple-Choice-Aufgaben können jeweils **keine**, **eine** oder **mehrere** korrekte Antworten aufgeführt sein.
- Füllen Sie bei Aufgaben mit mehreren Feldern (_____) *alle* Felder aus.
- Sie können die Rückseiten der Blätter für Nebenrechnungen oder längere Antworten nutzen. Machen Sie ggf. auf der Vorderseite deutlich, dass es auf der Rückseite weitergeht (z.B. mittels "s. Rückseite!")
- Falls Sie ein Kästchen versehentlich angekreuzt haben, schwärzen Sie es vollständig, um es als „nicht angekreuzt“ zu markieren.



1. JavaScript: Funktionen, Objekte, Closures

5 Punkte

Betrachten Sie die folgenden beiden konkurrierenden und funktionsfähigen Umsetzungen eines Taschenrechners.

```
2
3   var MakeCalculator1 = function() {
4
5       return {
6           result: 0,
7           add:   function(n) { this.result += n; return this.result; },
8           mult:  function(n) { this.result *= n; return this.result; },
9           reset: function()  { this.result = 0;  return this.result; }
10      }
11
12  };
13
14  var MakeCalculator2 = function() {
15
16      var result = 0;
17
18      return {
19          add:   function(n) { result += n; return result; },
20          mult:  function(n) { result *= n; return result; },
21          reset: function()  { result =0;   return result; }
22      };
23  };
24
```

Welche der beiden Methoden verwendet eine Closure?

☐ `MakeCalculator1()` ☑ `MakeCalculator2()`

Von welchem Typ is der Rückgabewert der jeweiligen Funktion?

`MakeCalculator1:` ☐ Funktion ☑ Objekt ☐ Sonstiges

`MakeCalculator2:` ☐ Funktion ☑ Objekt ☐ Sonstiges

Welchen Unterschied der beiden Implementierungen gibt es bzgl. der Datenkapselung?

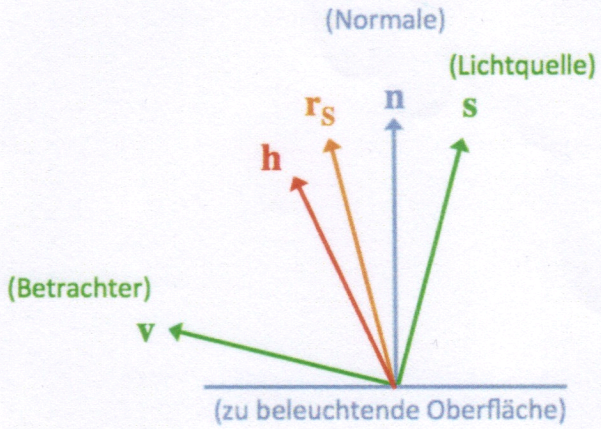
Antwort:

Bei MakeCalculator1 ist „mult" eine Eigenschaft des Objekts, bei MakeCalculator2 wird sie über eine Closure gespeichert.

2. Skalarprodukt und Phong

5 Punkte

Das folgende Diagramm zeigt schematisch die Vektoren, die bei der lokalen Beleuchtungsberechnung nach Phong oder Blinn-Phong eine Rolle spielen. Alle Vektoren seien normalisiert.



r_s : Spiegelung von s an n

h : Winkelhalbierende zwischen s und v

Die Beleuchtungsmodelle bestehen jeweils aus einem ambienten, einem diffusen und einem spekularen Term. Auf welchen Winkeln bauen die Terme auf?

Der diffuse Term basiert auf dem Winkel zwischen ___n___ und ___s___.

Der spekulare Term nach Phong basiert auf dem Winkel zwischen ___r_s___ und ___v___.

Was gilt, wenn es für den zu beleuchtenden Punkt "Dämmerung" ist, also genau am Übergang von Tag und Nacht?

Der Winkel zwischen ___n___ *und* ___s___ *beträgt* ___90___ *Grad. Das Skalarprodukt der beiden o.a. Vektoren beträgt* ___0___.

3. Lineare Interpolation

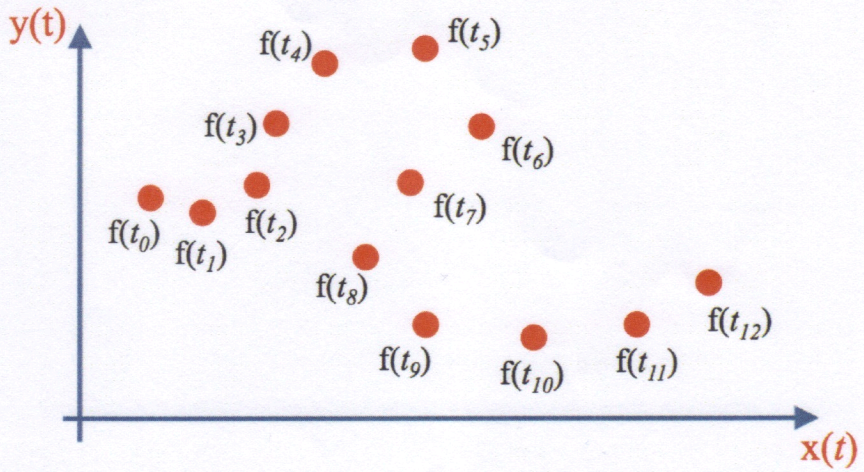
5 Punkte

Gegeben sind zwei Punkte **A** und **B**. Mit welcher Formel berechnet man die Position des Punktes **C**, der auf zwei Drittel der Strecke von **A** nach **B** liegt?

C = $\left(1-\frac{2}{3}\right)A + \frac{2}{3}B = \frac{1}{3}A + \frac{2}{3}B$

4. Tangente einer diskreten parametrischen Kurve — 5 Punkte

Gegeben sei eine parametrische Kurve $\mathbf{f}(t) = (\, x(t), y(t)\,)$. Die Werte dieser Kurve sind nur an einigen diskreten Stellen $t_0, t_1, t_2, \ldots t_N$ bekannt. Die nachfolgende Zeichnung zeigt beispielhaft eine solche Kurve.



Welcher Art ist die Tangente $\mathbf{f}'(t)$?

☐ Skalar ☑ Zweidimensionaler Vektor ☐ Dreidimensionaler Vektor

Wie berechnet man approximativ die Tangente der Kurve an einem der inneren Punkte $t_1, t_2, \ldots t_{N-1}$?

$\mathbf{f}(t_i) = \frac{1}{2}\left[f(t_{i+1}) - f(t_{i-1}) \right]$

5. Interpolation einer Folge von Punkten — 5 Punkte

Gegeben ist eine Folge $(\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_N)$ von $N+1$ Punkten, durch die Sie eine C_1-stetige Kurve legen möchten, z.B. um einen glatten Pfad durch diese Punkte zu definieren.

1. Wenn Sie den Pfad durch nur eine einzige Bézierkurve berechnen möchten, wie viele Kontrollpunkte bzw. welchen Grad muss diese Kurve dann mindestens besitzen?

   Kontrollpunkte: 4 Grad: 3 nur richtig für N=3

2. Wenn Sie den Pfad nach dem Schema von Catmull-Rom wie im Unterricht vorgestellt konstruieren, wie viele Kurvenstücke von welchem Grad verwenden Sie dann?

   Kurvenstücke: n Grad: n+1

6. Szenengraph / Skelett und Haut

5 Punkte

Welche Dinge gehören beim Modellieren eines hierarchischen Objekts in das Skelett, und welche Dinge in die Haut? Beschreiben Sie einen Fall, der illustriert, warum es ungünstig ist, nicht zwischen Skelett und Haut zu unterscheiden.

Antwort:

Skelett beschreibt wie einzelne Komponenten des Modells aneinander stehen (z.B. Kopf über Hals)
Die Haut beschreibt die Dimension der Teile (Kopf ist größer als Hals).
Da die Transformationen verkettet aufeinander angewandt werden, wäre es doof beides extra von Transformationen zusammen anzuwenden, da man so immer wieder „zurücktransformieren" (relativ dazu) muss.
z.B. Kopf soll groß sein -> dann wäre der Hals auch gleich groß und müsste relativ zum Kopf „zurücktransformiert" werden -> große Formeln mit keiner konstanten Bezugsgröße!

7. Struktur der programmierbaren Grafik-Pipeline

5 Punkte

Im folgenden geht es um die Reihenfolge und die Zusammenhänge zwischen den Pipeline-Stufen von WebGL. Welche der folgenden Aussagen sind richtig?

oh ☒ Auf Attribute-Variablen kann im Fragment-Shader lesend zugegriffen werden.

oh ☑ Primitive Assembly wird vor dem Vertex Shading benötigt.

☑ Die Ausgabe der Rasterisierung wird im Fragment-Shader verwendet.

☑ Stencil Test ist eine Framebuffer-Operation.

☑ Im Vertex Shader hat man keinen Zugriff auf die Attribute benachbarter Vertices.

8. Kategorien von Shader-Variablen in GLSL — 5 Punkte

(3P)

Was bedeuten die beiden Variablentypen **`uniform`** und **`attribute`** in der OpenGL Shading Language? In welchem Shadertyp können Sie jeweils wie verwendet werden, und wodurch unterscheiden sie sich?

Antwort:

uniform ✓

- globale Variable, die sich innerhalb eines drawElements()-Befehl nicht mehr ändert
- Vertex- wie Fragment-Shader können darauf zugreifen (read-only)

attribute

- wird dem Vertexshader übergeben
- Vertex-Shader darf lesen und schreiben −1
- enthält Eigenschaften pro Vertex! −1

9. Vertex Shader für Fragment-Phong und Texturierung — 5 Punkte

(2P)

In Ihrer Applikation werden geometrische Modelle mittels einer im Fragment-Shader berechneten Phong-Beleuchtung sowie mit Hilfe von Texturen dargestellt. Welche drei Aufgaben muss der Vertex-Shader in diesem Fall erfüllen?

1. Clip-koordinaten erzeugen ✓
2. Attribut-Buffer und Element-Buffer - beschleunigen ?
3. gl_Position = projectionMatrix * modelViewMatrix * v ? Siehe 1

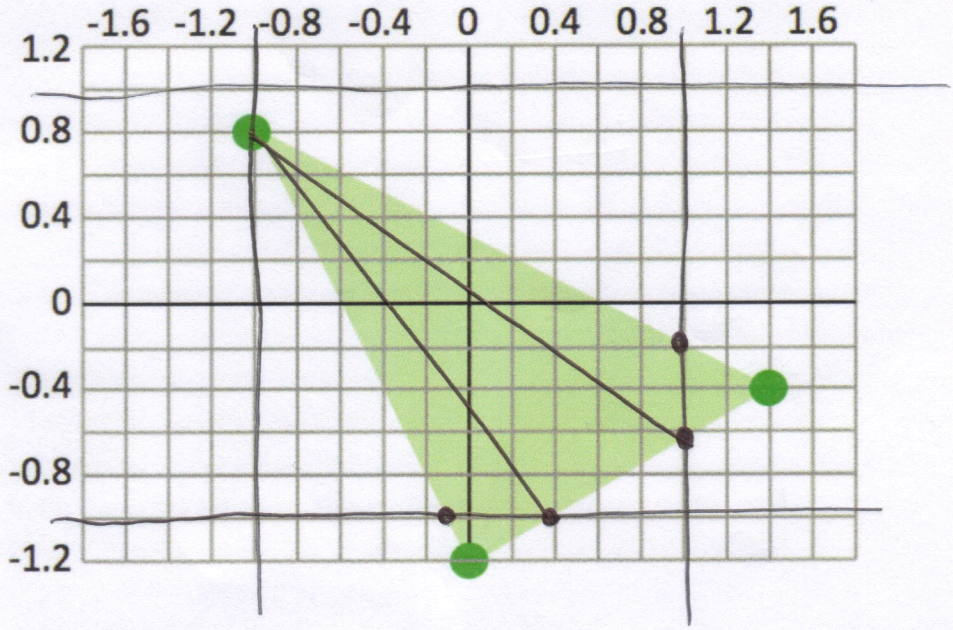
Attribut-Buffer mit Normalen + Texturen — was damit tun?

Normalen: 1 (3 pro Vertex), Texturen: 2

10. Clipping

5 Punkte

Zeichnen Sie in die untenstehende Grafik die beim Clipping des Dreiecks entstehenden Vertices ein. Zeichnen Sie weiterhin eine Möglichkeit ein, wie die Kanten der geclippten Dreiecke verlaufen könnten.



11. Rasterisierung

~~10~~ **Punkte** max 5.

3P

Was sind Ein- und Ausgabe der Rasterisierung, und wie wird diese Ausgabe berechnet?

Ein: geclipptes Primitiv
Aus: Fragmente
- Pixelraster wird auf Bild gelegt; überdeckt das Primitiv einen Pixel, wird ein Fragment erzeugt (Fragmente werden für jedes Primitiv einzeln erzeugt ⇒ mehrere Fragmente können auf ein Pixel fallen)

Wie viele Fragmente können pro Pixel bei der Rasterisierung eines Primitivs entstehen, (ohne Multisampling)?

- [ ] 0
- [x] 1
- [ ] 2
- [x] beliebig viele

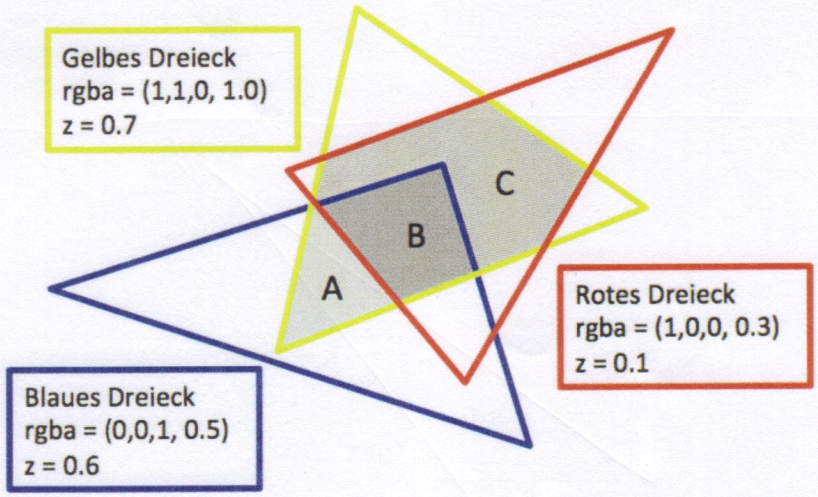
-2

12. Framebuffer-Operationen: Z und Alpha

5 Punkte

4P

In dieser Aufgabe vollziehen Sie die Arbeit und das **Zusammenspiel** der wichtigsten beiden Framebuffer-Operationen nach: Z-Buffer und Alpha-Blending.



Betrachten Sie die drei Dreiecke in der obigen Abbildung. Die angegebenen Werte **rgba** (Farbe+Alpha) und **z** (Tiefe) gelten jeweils auch für das Innere der Dreiecke; lediglich in der Skizze sind zur besseren Übersicht nur ihre Umrandungen dargestellt. Je größer der z-Wert, desto weiter hinten in der Szene befindet sich das jeweilige Dreieck.

Zunächst sei Alpha-Blending ausgeschaltet, der übliche Z-Test eingeschaltet, und die Dreiecke werden in der Reihenfolge gelb-rot-blau gezeichnet.

Welche Farbe hat Region **A**? R = 0 G = 0 B = 1 ✓

Welche Farbe hat Region **B**? R = 1 G = 0 B = 0 ✓

Nun löschen wir das Bild wieder und zeichnen zunächst nur das gelbe Dreieck. Danach schalten wir auch das Alpha-Blending ein. Dabei wird folgende Blending-Funktion verwendet: $\mathbf{C} = \mathbf{C_F} * \text{alpha}_F + \mathbf{C_P} * (1 - \text{alpha}_F)$. Dabei ist **C** die neue Farbe des Pixels nach dem Alpha-Blending, $\mathbf{C_P}$ die bisherige Farbe des Pixels, $\mathbf{C_F}$ die Farbe des neuen Fragments, und alpha_F der Alpha-Wert des Fragments.

Rot * 0.3 + Gelb * (1 - 0.3)

Nun wird als nächstes das rote Dreieck gezeichnet.

Welche Farbe hat Region **C** nun? R = 0.7 + 0.3 → 1 G = 0.7 B = 0 ✓

Abschließend wird das blaue Dreieck gezeichnet.

Blau * 0.5 + Gelb * (1-0.5) ✓

Welche Farbe hat Region **A**? R = 0.5 G = 0.5 B = 0.5

Welche Farbe hat Region **B**? R = 0.3 + 0.35 → 0.65 G = 0.35 B = 0.35 (−1)

0.3 * Rot + (0.5, 0.5, 0.5) * (1 - 0.3)

13. Texturfilterung

5 Punkte (2P)

In welcher Situation kann eine Mip-Map sinnvoll eingesetzt werden?

Anwendungen mit verschiedenen „Zoom"-Stufen
→ je näher rangezoomt wird, desto höher aufgelöst wird das Bild

-3

Warum muss eine Mip-Map erzeugt werden - kann die gleiche Berechnung nicht einfach direkt im Shader erfolgen?

- Rechenintensiv ✓

14. Bump- vs. Displacement Map

5 Punkte

Welcher Eigenschaft/Größe/Variable im Shadercode wird durch eine Bump Map verändert, und welcher durch eine Displacement Map?

Bump Map: Normalenrichtung ✓ Displacement Map: Geometrie ✓

In welchem Shadertyp implementiert man eine Bump Map, und in welchem Shadertyp eine Displacement Map?

Bump Map: [redacted] Fragment ✓ Displacement Map: [redacted] Vertex ✓

Wie können Sie am Umriss eines Objekts erkennen, ob es sich wahrscheinlich eher um eine Bump- oder Displacement Map handelt?

Antwort:

Bei der BumpMap wird die Geometrie nicht verändert, d.h. erscheint ein Objekt als hätte es Beulen o.Ä. kann man das Objekt von der Seite betrachten und [redacted] man wird keine Beulen sehen können.
Beim Displacement Mapping hingegen wird die Geometrie verändert → Beulen sind aus jeder Perspektive erkennbar. ✓