Aufgabe 1: Wahrnehmung, Farbe und Rasterbilder

Teilaufgabe 1a

Was versteht man unter Metamerie beim Farbsehen des Menschen?

Metamerie ist das Phänomen, dass verschiedene Spektren den selben Farbeindruck erzeugen können.

Teilaufgabe 1b

Was versteht man unter Schwarzkörperstrahlung und Farbtemperatur?

Ein Schwarzkörper ist eine idealisierte thermische Strahlungsquelle. Die idealisierung besteht darin, dass der Körper die komplette auftretende Strahlung vollständig absorbiert. Gleichzeitig sendet er Wärmestrahlung (Schwarzkörperstrahlung) aus, welche nur von seiner Temperatur abhängig ist.

Die Farbtemperatur ist ein Maß, um einen jeweiligen Farbeindruck einer Lichtquelle zu bestimmen.

Teilaufgabe 1c

Siehe martin-thoma.com/html5/graphic-filters zum ausprobieren.

(A) Hervorheben von horizontalen Kanten.
(B) Unschärfe / Weichzeichnen
(C) Hervorheben aller Kanten (Schärfen; vgl. Übung 02_Bildoperationen, Folie 19)[1]
(D) Hervorheben aller Kanten (Invertierter Laplace-Filter), entfernen vom Rest

Teilaufgabe 1d

Was versteht man unter einem normalisierten Filterkernel?

Ein normalisierter Filterkernel hat als Summe der Element den Wert 1.

Welche globale Eigenschaft eines Bildes ändert sich, wenn ein Filterkernel nicht normalisiert ist?

Die Gesamthelligkeit des Bildes ändert sich nicht (vgl. Übungsfolie `02_Bildoperationen` Folie 19)

[1] Siehe martin-thoma.com/html5/graphic-filters/graphic-filters.htm

Aufgabe 2: Prozedurale Modelle

Teilaufgabe 2a

Was sind Turbulenzfunktionen und wie können Sie aus Noise-Funktionen gebildet werden?

Eine Turbulenzfunktion summiert k Oktaven mehrerer Noise-Funktionen n auf:

$$\text{turbulence}(x) = \sum_k \left(\frac{1}{2}\right)^k \cdot n(2^k \cdot x)$$

Einsatzgebiete:

- Natürliche Oberflächen
- Feuer

Teilaufgabe 2b: Kontext-Freie Lindenmayer-Systeme

$$\begin{array}{llllll}
(1) & F & \rightarrow & F[+F][-F] & \rightarrow & F[+F[+F][-F]][-F[+F][-F]] \\
(2) & F & \rightarrow & F[+F] & \rightarrow & F[+F][+F[+F]] \\
(3) & F & \rightarrow & F[f-F]fF & \rightarrow & F[f-F]fF[f-F[f-F]fF]fF[f-F]fF
\end{array}$$

Teilaufgabe 2c: Turtle-Grafiken

- Die Grafik links oben ist (1)
- Die Grafik in der Mitte, unten ist (2)
- Die Grafik rechts unten ist (3)

Aufgabe 3: Supersampling und Baryzentrische Koordinaten

Teilaufgabe 3a

Was ist adaptives Supersampling?

Beim adaptiven Supersampling wird durch zwei benachbarte Pixel jeweils ein Strahl geschossen. Ist die Differenz der Pixelwerte über einem Schwellwert, so schießt man weitere Strahlen zwischen den beiden Pixeln. Dies wiederholt man so lange, bis man unter dem Schwellwert ist.

Was ist stochastisches Supersampling mit Stratifikation?

Beim stochastischen Supersampling wird jeder Pixel in ein Gitter unterteilt und durch jeden Gitterpunkt wird mit einer gewissen Wahrscheinlichkeit ein Strahl geschossen.

Was sind die Unterschiede zwischen adaptivem Supersampling und stochastischem Supersampling mit Stratifikation?

Adaptives Supersampling schießt nur bei Bedarf weitere Strahlen. Allerdings kann man Fälle konstruieren, wo adaptives Supersampling immer fehlschlägt.

Teilaufgabe 3b

$$\lambda_A = \frac{A_\Delta(P,B,C)}{A_\Delta(A,B,C)} = \frac{2}{7.5} = \frac{4}{15} \tag{1}$$

$$\lambda_B = \frac{A_\Delta(P,A,C)}{A_\Delta(A,B,C)} = \frac{2.5}{7.5} = \frac{5}{15} \tag{2}$$

$$\lambda_C = \frac{A_\Delta(P,A,B)}{A_\Delta(A,B,C)} = \frac{3}{7.5} = \frac{6}{15} \tag{3}$$

Aufgabe 4: Texturen

Teilaufgabe 4a

Wie wird aus einer Textur eine Mip-Map-Pyramide erstellt?

Man erzeugt Mip-Maps durch „zusammenfassen“ von jeweilse 2×2 Texeln. Es werden solange neue Mip-Map-Stufen generiert, bis man eine Mip-Map erzeugt hat, die nur noch aus einem Texel besteht. In Stufe 0 wird die ursprüngliche Textur gespeichert, in Stufe 1 dann eine Textur die in beiden Dimensionen halbiert wurde usw.

Wie hoch ist der zusätzliche Speicherbedarf?

Der zustätzliche Speicherbedarf z ist $\approx 1/3$ der ursprünglichen Texturgröße.

Beweis:

Es gilt:

$$z = \sum_{i=1}^{\infty} \left(\frac{1}{4}\right)^i$$

Dies ist eine geometrische Reihe. Die Summe der ersten n Terme ist

$$\frac{1-(1/4)^n}{1-1/4}$$

daher gilt:

$$\lim_{n\to\infty} = 1 - \frac{1}{1-1/4} = 1/3$$

Teilaufgabe 4b

Welche Probleme bei der Texturfilterung im Fall der Verkleinerung (Texture Minification) löst Mip-Mapping?

Mip-Mapping verringert Aliasing-Effekte.

Welche Probleme bei der Texturfilterung im Fall der Verkleinerung löst Mip-Mapping nicht?

Verwaschenes aussehen bei länglichem Footprint, da Mip-Map isotrop (TODO: erklären, ausformulieren).

Teilaufgabe 4c

Wofür verwendet man Environment Mapping?

Darstellung reflektierender Objekte mit Spiegelung von Umgebung ohne geometrische Repräsentation.

Was speichert eine Environment Map?

Beleuchtungsinformationen

Welche vereinfachenden Annahmen werden bei der Anwendung getroffen?

Der Betrachter ist sehr weit von der Umgebung entfernt, sodass die Position keine Rolle spielt und ausschließlich die Blickrichtung wichtig ist.

Aufgabe 5: α-Clipping

Teilaufgabe 5a

$$\mathrm{WEC}_{x_{\min}}(P_0) = -3 - 0 = -3 \qquad \mathrm{WEC}_{x_{\min}}(P_1) = 6 - 0 = 6 \tag{4}$$
$$\mathrm{WEC}_{x_{\max}}(P_0) = 10 + 3 = 13 \qquad \mathrm{WEC}_{x_{\max}}(P_1) = 10 - 6 = 4 \tag{5}$$
$$\mathrm{WEC}_{y_{\min}}(P_0) = 4 - 0 = 4 \qquad \mathrm{WEC}_{y_{\min}}(P_1) = -2 - 0 = -2 \tag{6}$$
$$\mathrm{WEC}_{y_{\max}}(P_0) = 6 - 4 = 2 \qquad \mathrm{WEC}_{y_{\max}}(P_1) = 6 + 2 = 8 \tag{7}$$

Teilaufgabe 5b

Wenn Sie alleine die WEC betrachten, welche Kanten des Viewports werden dann potenziell geschnitten? Begründen Sie Ihre Antwort mit Hilfe der Outcodes!
Da für $\mathrm{WEC}_{x_{\min}}(P_0) < 0$ und $\mathrm{WEC}_{y_{\min}}(P_1) < 0$ ist der Outcode $x_{\min}(P_0) = 1$ und

Outcode $y_{\min}(P_1) = 1$. Daraus folgt, dass $x_{\min}$ und $y_{\min}$ auf Schnitt getestet werden müssen. (TODO: Stimmt das?)

Teilaufgabe 5c

$$\alpha_{max} = 1, \quad \alpha_{min} = 0 \tag{8}$$
$$\alpha_{min} = \max(\alpha_{min}, \alpha_s) = 1/3 \tag{9}$$
$$\alpha_s = \frac{\mathrm{WEC}_{y_{\min}}(P_1)}{\mathrm{WEC}_{y_{\min}}(P_0) - \mathrm{WEC}_{y_{\min}}(P_1)} = 4/6 = 2/3 \tag{10}$$
$$\alpha_{max} = \min(\alpha_{max}, \alpha_s) = 2/3 \tag{11}$$

Das neue Liniensegment ist somit $(P_0 + 1/3 \cdot (P_1 - P_0), P_1 - 2/3 \cdot (P_1 - P_0))$.

Aufgabe 6

Teilaufgabe 6a

Was ist der Unterschied zwischen Gouraud- und Phong-Shading bei der Schattierung eines Dreiecks mit einem Rasterisierungsverfahren?

- **Gouraud:** Beleuchtungsberechnung nur an den Eckpunkten mit gemittelter Normale der angrenzenden Facetten
- **Phong:** Beleuchtungsberechnung pro Fragment mit interpolierter Normale

Teilaufgabe 6b

Was versteht man unter einem Accumulation-Buffer, wie Sie ihn von OpenGL kennen?

Der Accumulation-Buffer ist ein Zwischenspeicher zur Kombination mehrerer Redinering-schritte.

Nennen Sie zwei Beispiele für Effekte, die sich mit einem Accumulation-Buffer erreichen lassen!

- Bewegungsunschärfe
- Tiefenunschärfe

Teilaufgabe 6c

Aussage	Wahr	Falsch	Begründung
T-Vertices können bei Phong-Shading Artefakte verursachen.	☑	☐	TODO: Wirklich?
Z-Fighting kann durch die Repräsentation der Tiefenwerte mit beschränkter Genauigkeit im Tiefenpuffer entstehen.	☑	☐	TODO: Wirklich?
Je feiner eine Oberfläche tesselliert wird, umso geringer werden die Unterschiede zwischen Gouraud- und Phong-Shading.	☑	☐	TODO: Wirklich?
Bei der Rasterisierung ist eine perspektivisch korrekte Abbildung der Textur aufwendiger als eine affine, da pro Pixel eine zusätzliche Division benötigt wird.	☐	☐	TODO
Der Scissor-Test dient dazu, durchsichtige Teile einer Oberfläche gemäß einer Textur wegzuschneiden.	☐	☑	Dient dazu Teile auserhalb des Rechtecks wegzuschneiden (vgl. OpenGL Teil 2, Folie 103)
Screendoor-Transparency stellt transparente Objekte mittels Blending dar.	☐	☐	TODO

Aufgabe 7: OpenGL

Teilaufgabe 7a: Blending

Teilaufgabe 7a (1)

Es erfolgt kein Blending. Durch den aktivierten Tiefentest wird der rote Würfel nicht beachtet, da er hinter dem blauen Ball liegt. Daher:

`color(P) = (0.0, 0.0, 1.0, 0.5)`

siehe „OpenGL Teil 2 und 3“, Folie 88.

Teilaufgabe 7a (2)

$$c' = 1 \cdot (0, 0, 1, 0.5) + 0.5 \cdot (0, 0, 0, 0) = (0, 0, 1, 0.5) \tag{12}$$
$$c = 1 \cdot (1, 0, 0, 0.5) + 0.5 \cdot (0, 0, 1, 0.5) = (1, 0, 0.5, 0.75) \tag{13}$$

Teilaufgabe 7b

Bringen Sie die folgende Operationen in die richtige Reihenfolge, wie sie in der Fixed-Function-Pipeline von OpenGL ausgeführt werden:

Die Fixed Function Pipeline kann nur Gouraud oder Flat Shading. Das heißt, die Beleuctung wird pro Vertex in Kamera Koordinaten berechnet. Dementsprechend:

1. Model-View-Transformation anwenden (1)
2. Projektionstransformation anwenden (4)
3. Beleuchtungsberechnung (5)
4. Texturierung (3)
5. Tiefentest (2)

Leider sind wir uns bei dieser Lösung ziemlich unsicher (siehe GitHub issue(s)).

- Beleuchtungsberechnung in Kamerakoordinaten nach der Modelview-Transformation (Folie 39 ⇒ Erst (1), dann (5))

Teilaufgabe 7c

Zwischen 2 und 3, da die Rasterisierung nur für Dreiecke erfolgen soll, die im Viewfrustum liegen.

Aufgabe 8

Teilaufgabe 8a

Operation	Vertex-Shader	Fragment-Shader	Weder noch
Model-View-Transformation anwenden	☑	☐	☐
Tiefentest	☐	☐	☑
Texturierung	☐	☑	☐
Projektionstransformation anwenden	☑	☐	☐
Beleuchtungsberechnung	☐	☑	☐
Clipping	☐	☐	☑

Teilaufgabe 8b

- `attribute`: Attribut eines Vertex; nur für Vertex-Shader; z.B. Farbe oder Normale
- `uniform`: Bei jedem Shader-Aufruf gleich (also insbesondere für jeden Vertex gleich); read-only; z.B. Transformationsmatrix

- `varying`: weitergegebene/interpolierte Werte (schreiben in einem Shader, lesen im darauffolgenden Shader)

Teilaufgabe 8c

shader.vert

```
varying vec3 L, N;

void main() {
    gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
    vec3 P = vec3(gl_ModelViewMatrix * gl_Vertex);
    L = gl_LightSource[0].position - P;
    N = gl_NormalMatrix * gl_Normal;
}
```

shader.frag

```
varying vec3 L, N;

void main() {
    float kd = max(0.0, dot(normalize(L), normalize(N)));
    gl_FragColor = vec4(kd);
}
```

Aufgabe 9: Bézier-Kurven

Teilaufgabe 9a

$$
\begin{aligned}
& b_0 \cdot (1 - 2u + u^2) + b_1(2u - 2u^2) + b_2 \cdot u^2 && (14)\\
=& b_0 - 2b_0u + b_0u^2 + 2b_1u - 2b_1u^2 + b_2u^2 && (15)\\
=& \underbrace{(b_0 - 2b_1 - b_2)}_{=a_2} u^2 + \underbrace{(2b_1 - 2b_0)}_{=a_1} u + \underbrace{b_0}_{=a_0} && (16)
\end{aligned}
$$

Teilaufgabe 9b

Was versteht man unter affiner Invarianz der Bézier-Repräsentation?

Für jede Bézierkurve $F(u)$ und jede affine Abbildung $\varphi(x) = Ax + t$ gilt:

$$\varphi(F(u)) = \sum_{i=0}^{n} B_i^n(u)\varphi(\mathbf{b}_i)$$

Das heißt es genügt bei Transformationen die Kontrollpunkte $\mathbf{b}_i$ zu transformieren.

Teilaufgabe 9c

Für C^0-Stetigkeit muss $c_0 = b_3 = (4,1)$ gelten.

Für C_1-Stetigkeit muss $b_2 - b_3 = c_0 - c_1 \Leftrightarrow (3,1) - (4,1) = (4,1) = \underbrace{(5,1)}_{=c_1}$ gelten.

Für C_2-Stetigkeit muss

$$b_2 + (b_2 - b_1) = c_1 + (c_1 - c_2) \tag{17}$$

$$\Leftrightarrow (3,1) + ((3,1) - (1,2)) = (5,1) + ((5,1) - c_2)) \tag{18}$$

$$\Leftrightarrow (5,0) = (10,2) - c_2 \tag{19}$$

$$\Leftrightarrow c_2 = (5,2) \tag{20}$$