Aufgabe 1: Raytracing

Teilaufgabe 1a

Phong-Beleuchtungsmodell anwenden:

- Berechnung des ambienten Anteils (indirekte Beleuchtung, Licht von anderen Oberflächen)
- Berechnung der Reflektion
  - Berechnung des spekularen Reflektion. Diese findet nur in Richtung $R_L = 2N \cdot (N \cdot L)$ statt. In alle andren Richtungen fällt sie stark ab:

    $$I_s = k_s \cdot I_L \cdot \cos^n \alpha = k_s \cdot I_L (R_L \cdot V)^n$$

    wobei n der Phong-Exponent ist.
  - Berechnung der diffusen (Lambertschen) Reflektion:

    $$I_d = k_d \cdot I_L \cdot \cos\theta = k_d \cdot I_L \cdot (N \cdot L)$$

Also:

$$I = \underbrace{k_a \cdot I_L}_{\text{ambient}} + \underbrace{k_d \cdot I_L \cdot (N \cdot L)}_{\text{diffus}} + \underbrace{k_s \cdot I_L (R_L \cdot V)^n}_{\text{spekular}}$$

Auch:

- Berechnung von Schattenstrahlen
- Weitere, Strahlen rekursive verschießen

Teilaufgabe 1b

TODO

Aufgabe 2: Farben und Spektren

Teilaufgabe 2a

- RGB: LCD/CRT-Displays
- CMYK: Drucker
- HSV: TODO
- HSI: TODO
- XYZ Color Space: Farbraum für Konversion zwischen Farbräumen
- Lab-Farbraum: TODO

Teilaufgabe 2b

TODO

Teilaufgabe 2c

Metamerismus ist das Phänomen, dass unterschiedliche Spektren gleich aussehen können. Dies ist wichtig für Monitore, da sie aufgrund dieses Phänomens mit nur 3 Farben den gleichen Farbeindruck erwecken können wie mit einem komplexeren Spektrum.

Aufgabe 3: Transformationen

Teilaufgabe 3a

Transformationen mit homogenen Koordianten laufen Grundsätzlich nach folgendem Schema ab:

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{pmatrix} \leftarrow T \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Die Transformationsmatrix T für die Translation von homogenen Koordinaten ist von der Form

$$T = \begin{pmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{pmatrix}$$

Die Transformationsmatrix R für eine Rotation um den Punkt $c = (c_x, c_y)$ um den Winkel α ist

$$R_{\alpha,c} = \begin{pmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{pmatrix}$$

Die Idee ist nun, zuerst eine Rotation um 90° gegen den Urzeigersinn um $(0, 0)$ zu machen (Matrix R). Dann wird das Rechteck in Richtung der x-Achse um die hälfte gestaucht (Matrix S) und schließlich um 0.5 nach links verschoben (Matrix T):

$$R = \begin{pmatrix} \cos 90 & -\sin 90 & 0 \\ \sin 90 & \cos 90 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{1}$$

$$S = \begin{pmatrix} 0.5 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

$$T = \begin{pmatrix} 1 & 0 & -0.5 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

$$M = T \cdot S \cdot R \tag{4}$$

$$= \begin{pmatrix} 0 & -0.5 & -0.5 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5}$$

Zur Kontrolle:

$$M \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.5 \\ 0 \\ 1 \end{pmatrix} \qquad M \cdot \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.5 \\ 1 \\ 1 \end{pmatrix} \tag{6}$$

$$M \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix} \qquad M \cdot \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \tag{7}$$

Teilaufgabe 3b

TODO

Teilaufgabe 3c

Die Transformation von Welt- in Kamerakoordinaten wird auch *Kameratransformation* genannt. Die virtuelle Kamera ist durch die Position $\mathbf{e}$ und die negative Blickrichtung $\mathbf{w}$ definiert. Mithilfe des Up-Vektors $\mathbf{up}$ ergibt sich

$$\mathbf{u} = \mathbf{up} \times \mathbf{w} \qquad \mathbf{v} = \mathbf{w} \times \mathbf{u}$$

Es wird zuerst eine Translation um $-\mathbf{e}$ durchgeführt und dann eine Transformation in das Kamera-Koordiantensystem durchgeführt. Die Basis des Kamera-Koordinatensystems ist $\mathbf{u}, \mathbf{v}, \mathbf{w}$.

Das Verschieben ist einfach die Matrix

$$T = \begin{pmatrix} 1 & 0 & 0 & e_x \\ 0 & 1 & 0 & e_y \\ 0 & 0 & 1 & e_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

nun muss noch rotiert werden:

$$R = \begin{pmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ w_x & w_y & w_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Die Gesamttrasformationsmatrix ist also $V = R \cdot T$.

(vgl. `03_ Transformationen und homogene Koordinaten.pdf`, Folie 50)

Aufgabe 4: Phong-Beleuchtungsmodell

Teilaufgabe 4a

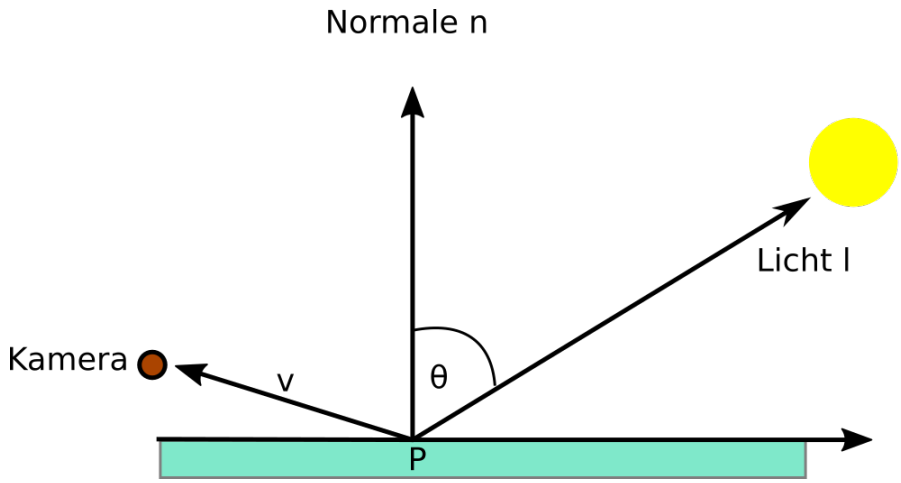

Abbildung 1: Situationsskizze. r_l ist senkrecht auf L in P.

$$\mathbf{r}_l = 2 \cdot \mathbf{n} \cdot (\mathbf{n} \cdot \mathbf{l}) - \mathbf{l}$$

Kaptiel 2, Folie 96.

Teilaufgabe 4b

Die spekulare Komponente im Phong Beleuchtungsmodell lautet

$$I_s = k_s \cdot I_L \cdot \cos^n \alpha = k_s \cdot I_L(\mathbf{r}_l \cdot \mathbf{v})^n$$

Teilaufgabe 4c

(i) Wie verändert sich das Glanzlicht, wenn e größer wird?
⇒ TODO
(ii) Wie verändert sich das Glanzlicht, wenn die Kugel um eine beliebige Achse rotiert?
⇒ TODO

Aufgabe 5: Dreiecke und Schattierung

Teilaufgabe 5a

Drei Punkte P_1, P_2, P_3 definieren eine Ebene. Diese Ebene ist eine Menge von Punkten

$$\{x \in \mathbb{R}^3 | x \text{ liegt auf der Ebene von } P_1, P_2, P_3\} = \{x \in \mathbb{R}^3 = n \cdot x = d\}$$

Es gilt also folgendes Gleichungssystem zu lösen:

$$\|n\| = 1 \tag{8}$$
$$n_x \cdot P_{1,x} + n_y \cdot P_{1,y} + n_z \cdot P_{1,z} = d \tag{9}$$
$$n_x \cdot P_{2,x} + n_y \cdot P_{2,y} + n_z \cdot P_{2,z} = d \tag{10}$$
$$n_x \cdot P_{3,x} + n_y \cdot P_{3,y} + n_z \cdot P_{3,z} = d \tag{11}$$

Teilaufgabe 5b

Die Normalen der Vertices eines Dreiecksnetzes können wie folgt berechnet werden:

1. Berechne Normale jedes Dreiecks
2. Für jeden Vertex wird nun die Summe der Normalen der angrenzenden Dreiecke gebildet.
3. Die Vertex-Normalen werden normalisiert, indem sie durch ihre Länge geteilt werden.

Teilaufgabe 5c

TODO

Teilaufgabe 5d

Gouraud-Shading im Vergleich zu Phong-Shading

- Vorteil: Schnellere Berechnung
- Nachteil: Schlechtere Ergebnisse

Aufgabe 6: Texturen und Texturfilterung

Teilaufgabe 6a

vgl. geometrische Interpretation (Kaptiel 2, Folie 43)

$$\lambda_1 = \frac{A_\Delta(S, P_2, P_3)}{A_\Delta(P_1, P_2, P_3)} \tag{12}$$

$$\lambda_2 = \frac{A_\Delta(P_1, S, P_3)}{A_\Delta(P_1, P_2, P_3)} \tag{13}$$

$$\lambda_3 = \frac{A_\Delta(P1, P_2, S)}{A_\Delta(P_1, P_2, P_3)} \tag{14}$$

Teilaufgabe 6b

$$T_S = \sum_{i=1}^{3} \lambda_i T_i$$

Teilaufgabe 6c

- Aliasing-Artefakte durch Unterabtastung (Kapitel 4, Folie 45): Beispiel TODO
- TODO: Beispiel TODO

Teilaufgabe 6d

TODO

Teilaufgabe 6e

Unter trilinearer Texturfilterung versteht man eine bilineare Interpolation der Stufe n, eine bilineare Interpolation der Stufe $n+1$ und dann eine lineare Interpolation dieser beiden Farben.

Teilaufgabe 6f

- `GL_TEXTURE_MAG_FILTER`: The texture magnification function is used when the pixel being textured maps to an area less than or equal to one texture element. It sets the texture magnification function to either `GL_NEAREST` or `GL_LINEAR`. `GL_NEAREST` is generally faster than `GL_LINEAR`, but it can produce textured images with sharper edges because the transition between texture elements is not as smooth. The initial value of `GL_TEXTURE_MAG_FILTER` is `GL_LINEAR`.
  Quelle: www.talisman.org/opengl-1.1/Reference/glTexParameter.html
- `GL_TEXTURE_MIN_FILTER`: The texture minifying function is used whenever the pixel being textured maps to an area greater than one texture element. There are six defined minifying functions. Two of them use the nearest one or nearest four texture elements to compute the texture value. The other four use mipmaps. [...]
  Quelle: www.talisman.org/opengl-1.1/Reference/glTexParameter.html

Bilineare Filterung (Kaptiel 4, Folie 38):

$$a = \frac{3}{8} - \frac{1}{4} = \frac{1}{8} \tag{15}$$

$$b = \frac{1}{2} - \frac{1}{4} = \frac{1}{4} \tag{16}$$

$$t_{12} = t_1 + a(t_2 - t_1) = 16 + \frac{1}{8} \cdot (-16) = 14 \tag{17}$$

$$t_{34} = (1-a)t_3 + at_4 = \frac{7}{8} \cdot 0 + \frac{1}{8} \cdot 32 = 4 \tag{18}$$

$$t = (1-b)t_{12} + bt_{34} = \frac{3}{4} \cdot 14 + \frac{1}{4} \cdot 4 = 10.5 + 1 = 11.5 \tag{19}$$

Aufgabe 7: Projektionen

TODO

Aufgabe 8: Beschleunigungsstrukturen

Teilaufgabe 8a

TODO

Teilaufgabe 8b

1 Der Baum einer Hüllkörperhierarchie ist immer balanciert.
⇒ TODO
2 Der Speicherbedarf für ein reguläres Gitter ist unabhängig von der Anzahl der Primitive.
⇒ TODO
3 Ein kD-Baum hat immer achsenparallele Split-Ebenen.
⇒ TODO
4 Ein kD-Baum braucht spezielle Vorkehrungen, um redundante Schnitttests mit demselben Dreieck auszuschliessen.
⇒ TODO
5 Ein Verfahren zur Erzeugung eines kD-Baumes erzeugt auch gültige BSP-Bäume.
⇒ TODO
6 Reguläre uniforme Gitter leiden nicht unter dem Teapot-in-a-Stadium Problem.
⇒ TODO
7 Die Komplexität der Bestimmung eines Schnittpunktes in einem BSP-Baum mit n Primitiven liegt im Optimalfall in $\mathcal{O}(\log n)$.
⇒ TODO
8 Das Traversieren einer Hüllkörperhierarchie kann abgebrochen werden sobald ein Schnittpunkt gefunden wurde.
⇒ Falsch. Es könnte einen Schnitt geben, der näher zur Kamera ist (TODO: Beispiel in Folien raussuchen.)

Aufgabe 9: Instancing (GLSL)

TODO

Aufgabe 10: Normal Mapping in Objektkoordinaten (GLSL)

TODO

Aufgabe 11: Bézier-Kurven und Bézier-Splines

Teilaufgabe 11a

TODO

Teilaufgabe 11b

TODO