Aufgabe 1: Raytracing

Teilaufgabe 1a

Raytracing nach Whitted, wie Sie es in der Vorlesung kennengelernt haben, folgt den Gesetzen der geometrischen Optik. Ergänzen Sie die folgende Liste um die 3 weiteren Strahltypen, die bei diesem Raytracing-Verfahren vorkommen!

(1) Primärstrahlen (2) Reflektionsstrahlen (rekursiv) (3) Transmissionsstrahlen (rekursiv) (4) Schattenstrahlen

Teilaufgabe 1b

Die folgenden Skizzen zeigen zwei Lichtstrahlen mit unterschiedlichem Einfallswinkel die an einer spekularen Glasoberfläche reflektiert werden (der Vektor N ist die Oberflächennormale).

In Bild 2, da dort der Winkel des Strahls auf die Oberfläche flacher ist.

Teilaufgabe 1c

Wie nennt man das physikalische Gesetz oder Prinzip, welches den Zusammenhang zwischen Einfallswinkel und Reflektivität beschreibt?

Snelliussches Brechungsgesetz. Es lautet

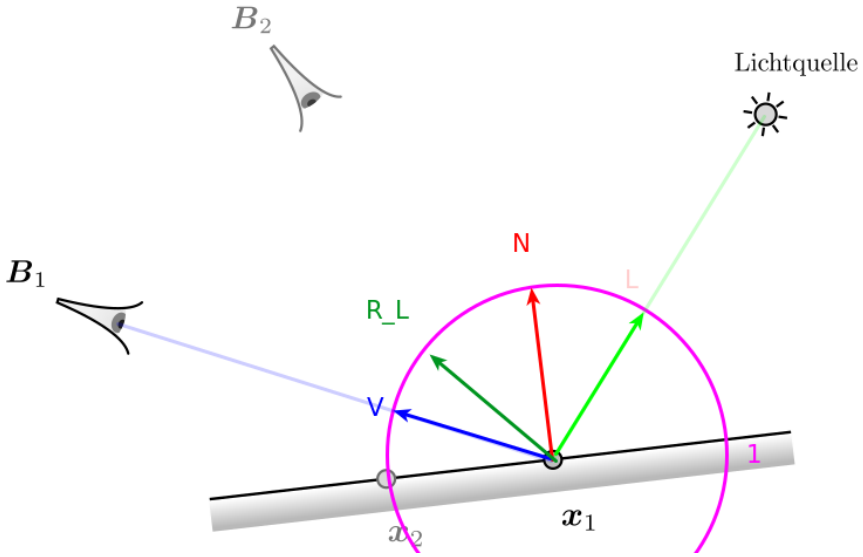
$$n_1 \cdot \sin(\theta_1) = n_1 \cdot \sin(\theta_2)$$

wobei die Winkel von der Oberflächennormale aus gemessen werden. n_1, n_2 sind Materialkonstanten.

Aufgabe 2: Beleuchtung, Licht und Wahrnehmung

Teilaufgabe 2a

Ergänzen Sie die Skizze und zeichnen Sie die 4 Vektoren ein, die im Phong-Beleuchtungsmodell für die Beleuchtungsberechnung benötigt werden! Verwenden Sie für die Skizze die Betrachterposition B_1 und den Oberflächenpunkt x_1



Die 4 Vektoren sind:

- View-Vektor V
- Normale N,
- Licht-Vektor L und
- Reflektionsvektor R_L

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

Teilaufgabe 2b

Der Wert welcher Komponente(n) des Phong-Beleuchtungsmodells verändert bzw. verändern sich, wenn in der obigen Situation...

(i) *... der Punkt x_2 statt x_1 betrachtet wird?*
L, R_L, V: Spekular und diffus

(ii) *... die Szene aus der Position B_2 statt B_1 betrachtet wird?*
V: diffus

Vektor	Punkt	Richtung	Kartesische Koordinaten
$(1, 2, 3, 1)$	☑	☐	$(1, 2, 3)$
$(1, 2, 3, 0.1)$	☑	☐	$(1, 2, 30)$
$(1, 2, 3, 0)$	☐	☑	$(1, 2, 3)$

Teilaufgabe 2c

In welcher Komponente taucht der sogenannte Phong-Exponent auf und welchen Einfluss hat er auf die Erscheinung einer Oberfläche? Wie ändert sich das Aussehen, wenn der Phong-Exponent größer gewählt wird?

Diffuse Komponente

Ein großes n führt dazu, dass **Glanzlichter** kleiner, aber intensiver werden. Die reflektion wird „perfekter“.

Teilaufgabe 2d

#	Aussage	Wahr	Falsch
1	Zu drei gewählten Primärfarben gibt es immer Spektralfarben, die durch die Kombination dieser drei Farben nicht realisierbar sind.	x	
2	Menschen können geringe Helligkeitsunterschiede im Bereich niedriger Lichtintensität besser wahrnehmen als im Bereich hoher Lichtintensität.	x	
3	Es gibt keinen linearen Zusammenhang zwischem dem CIE-XYZ- und dem RGB-Modell.		x
4	Gammakorrektur ist nur bei Röhrenmonitoren notwendig.		x

Aufgabe 3: Transformationen

Teilaufgabe 3a

Gegeben sind Vektoren in homogenen Koordinaten. Kreuzen Sie jeweils an, ob es sich um einen Punkt oder eine Richtung handelt. Geben Sie außerdem die dazugehörigen kartesischen Koordinaten an.

Ein Punkt hat als letzte Komponente einen Wert $\neq 0$, eine Richtung hat dort $= 0$.

Teilaufgabe 3b

Korrekt sind:

1. dreht die x-Achse in Richtung y-Achse
2. Scherung um Faktor a
3. erhält die Parallelität von Linien (bei den anderen beiden ist die Scherung ein Gegenbeispiel)

Aufgabe 4: Texturen und Texture-Mapping

Teilaufgabe 4a

Was versteht man unter Mip-Mapping? Welches Problem beim Texture Mapping soll damit gelöst werden und wann tritt dieses Problem auf? Wie erzeugt man Mip-Maps?

Mip-Mapping ist eine Vorverarbeitung der Textur, um das **Aliasing**-Problem bei **Minification** zu behandeln.

Es werden kleiner skalierte, vorverarbeitete Versionen der Textur erstellt (Stufe i: um Faktor 2^i pro Achse kleiner).

Teilaufgabe 4b

Was versteht man unter einer Environment Map? Nennen Sie eine Anwendung von Environment Mapping. Wie wird auf die Environment Map zugegriffen und welche vereinfachende Annahme wird dabei gemacht?

Was: Eine Environment-Map ist eine Textur zur Darstellung der Umgebung.

Anwendung: Durch eine Environment-Map kann die Reflektion/Beleuchtung eines Objekts bestimmt werden, ohne aufwendiges Ray-Tracing zu betreiben.

Konkret: Spiegelungen auf „flüssigem“ Terminator (vgl. Folien)

Annahme: Die Umgebung ist weit genug entfernt, sodass die Position keine Rolle spielt. Es wird nur die Blickrichtung verwendet. Man bestimmt für Cube-Maps also die Richtung (die Fläche des Würfels) und greift nur entsprechend dieser auf die Textur der betreffenden Würfelfläche zu.

Aufgabe 5

Teilaufgabe 5a

TODO

Teilaufgabe 5a

TODO

Aufgabe 6

Teilaufgabe 6a

TODO

Teilaufgabe 6b

TODO

Teilaufgabe 6c

spheretracing.frag

```
in vec3 A; // Ursprung des Strahls.
in vec3 D; // Die normalisierte Richtung des Strahls.
in float tMax; // Abbruchkriterium: maximale Suchdistanz.
uniform float epsilon; // Toleranz

// Distanzfunktion. Liefert den Abstand von x zur nächsten Fläche.
float DF( vec3 x ) { ... }

// Implementieren Sie Sphere Tracing in dieser Funktion.
bool sphereTrace( out vec3 pos, out int steps ) {
    pos = A;
    steps = 0;
    float t = 0.;
    while (t < tMax) {
        float d = DF(pos);
        pos += d * D;
        if (abs(d) < epsilon) {
```

```
18              return true;
19          }
20      }
21      return false;
22  }
```

Aufgabe 7

Teilaufgabe 7a

TODO

Teilaufgabe 7b

TODO

Aufgabe 8

Teilaufgabe 8a

TODO

Teilaufgabe 8b

TODO

Teilaufgabe 8c

TODO

Aufgabe 9

keyframing.vert

```
1 in vec4 p; // Position des Vertex in Objektkoordinaten.
2 uniform float t; // Aktueller Zeitpunkt.
3 uniform float t1; // Die Zeitpunkte der drei Keyframes.
4 uniform float t2;
```

```
uniform float t3;
uniform mat4 M1; // Die drei Transformationsmatrizen (Objekt->Welt).
uniform mat4 M2;
uniform mat4 M3;
uniform mat4 VP; // Die View-Projection-Matrix.

void main() {
    vec4 pWorld;
    if (t < t2) {
        pWorld = mix(M1 * p, M2 * p, (t - t1) / (t2 - t1));
    } else {
        pWorld = mix(M2 * p, M3 * p, (t - t2) / (t3 - t2));
    }

    gl_Position = VP * pWorld;
}
```

Aufgabe 10

Teilaufgabe 10a

shader.frag

```
void renderScene() {
    // Setup vor dem Löschen von Frame- und Tiefenpuffer
    glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    // Zeichnen der Szene ab hier

    //TODO
}
```

Teilaufgabe 10b

TODO

Teilaufgabe 10c

TODO

Aufgabe 11: Bézierkurven

Teilaufgabe 11a

TODO

Teilaufgabe 11b

TODO