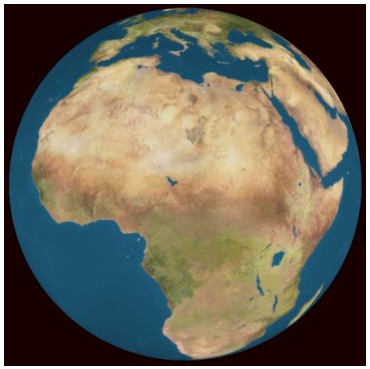
Computergrafik

Universität Osnabrück, Henning Wenke, 2012-06-19

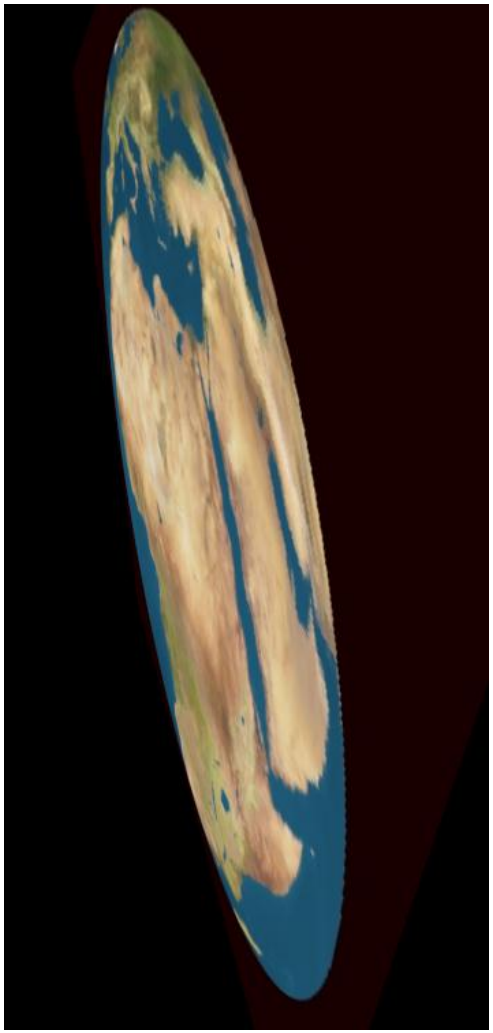
Die Erde ist eine Scheibe!*

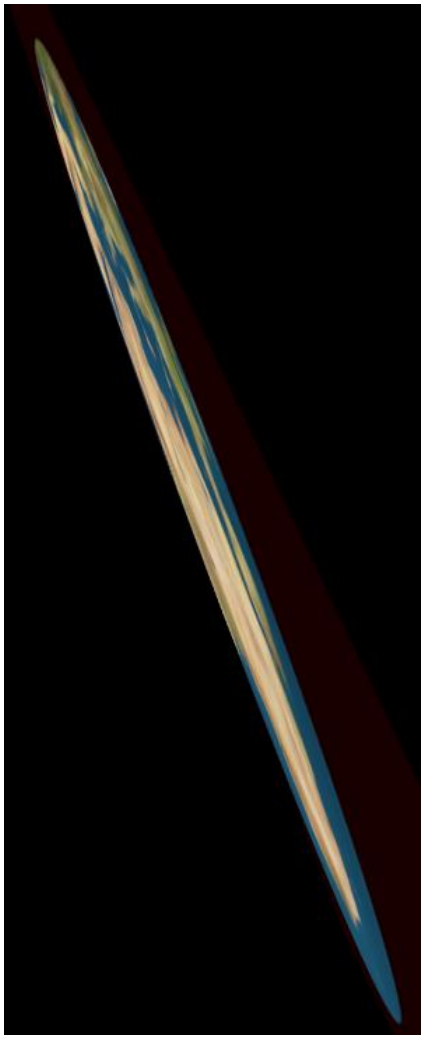
Die Erde ist eine Kugel!†



*2012-06-19, H.W., Wörtlich

†1992-11-02, Papst J.P. II. Sinngemäß.
Kirchengeschichtlicher Meilenstein





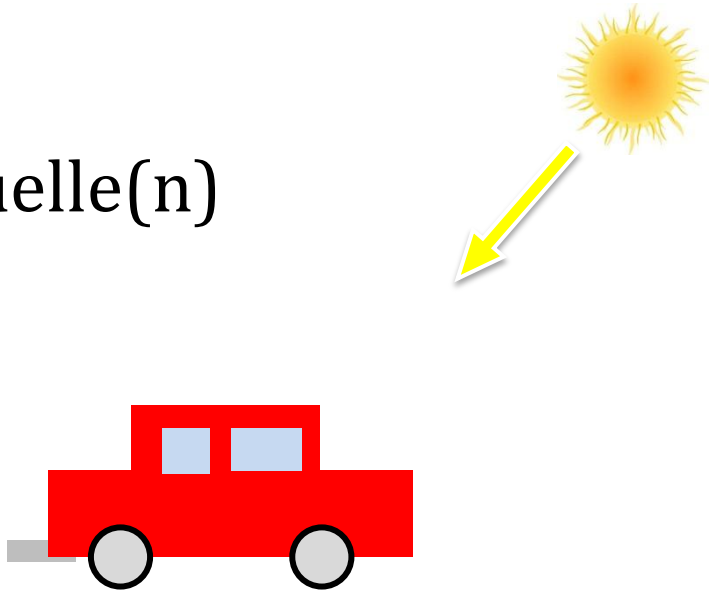
Kapitel XIV:

Lokale Beleuchtung

Unbeleuchtet / Beleuchtet

Lokale Beleuchtung

- Beleuchtet jeweils…
- …einen Punkt einer Oberfläche, repräsentiert durch:
  - Fragment
  - Vertex
- Es gehen nur dessen Eigenschaften ein
  - Material: Holz, Metall, Glas, …
  - Normale
- Und die der jeweiligen Lichtquelle(n)
  - Position
  - Leuchtrichtung
  - Entfernung
  - …

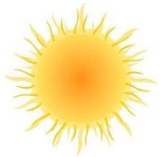


14.1

Lichtquellentypen

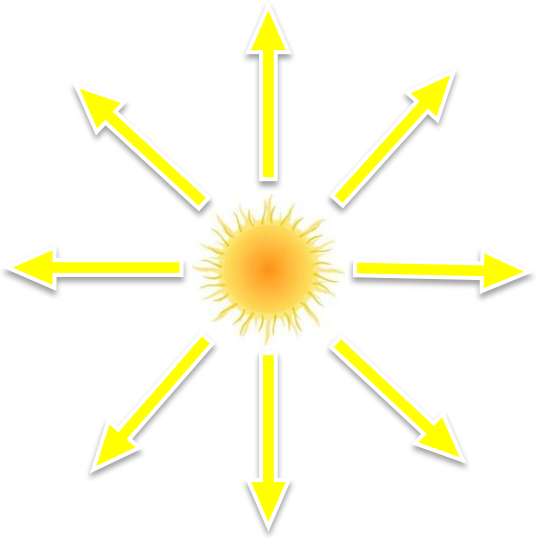
Directed Light

- Repräsentiert eine Lichtquelle im unendlichen, z.B. Sonne
- Maximale Intensität: $I_d = (I_{d,r}, I_{d,g}, I_{d,b})$
- Richtung: $\boldsymbol{l_d} = (x, y, z, 0)^T$
- Beispiel: Rotes Licht in z-Richtung
  - $I_d = (1,0,0)$
  - $\boldsymbol{l_d} = (0,0,1,0)^T$
- Hinweis: Wir hatten bisher gerichtete, weiße Lichtquelle



Point Light

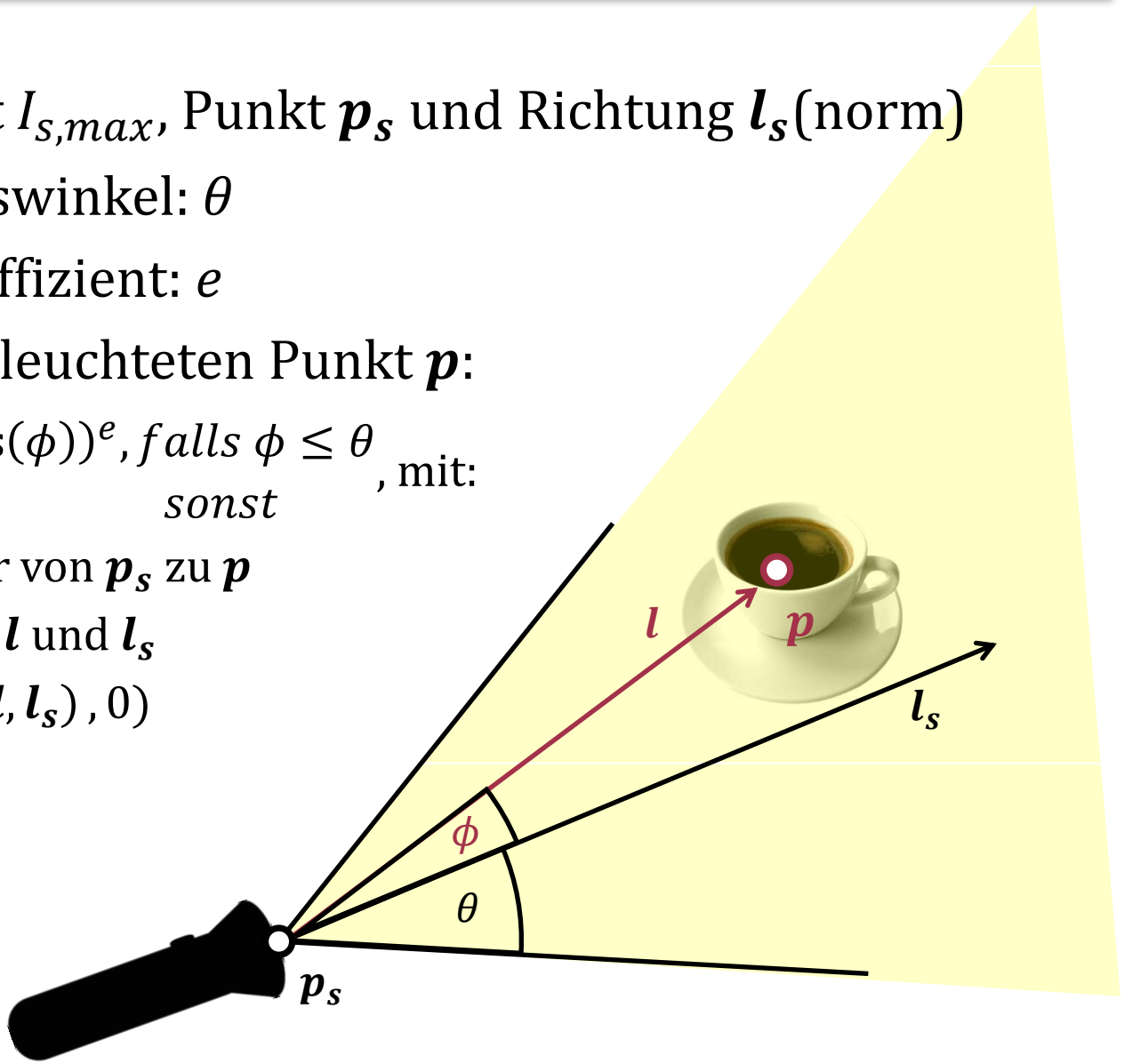
- Besitzt keine physikalische Entsprechung
- Besteht aus Position $\boldsymbol{p_p} = (x, y, z, 1)^T$
- Und maximaler Intensität (in $\boldsymbol{p_p}$): $I_{p,max}$
- Intensität nimmt quadratisch mit Abstand ab:
  - $I_p(p) = \frac{1}{|\boldsymbol{p}-\boldsymbol{p_p}|^2} I_{p,max}$ (Kugeloberfläche)
- Aus praktischen Gründen oft:
  - $I_p(p) = \frac{1}{a+b\cdot(\boldsymbol{p}-\boldsymbol{p_p})+(\boldsymbol{p}-\boldsymbol{p_p})^2} I_{p,max}$



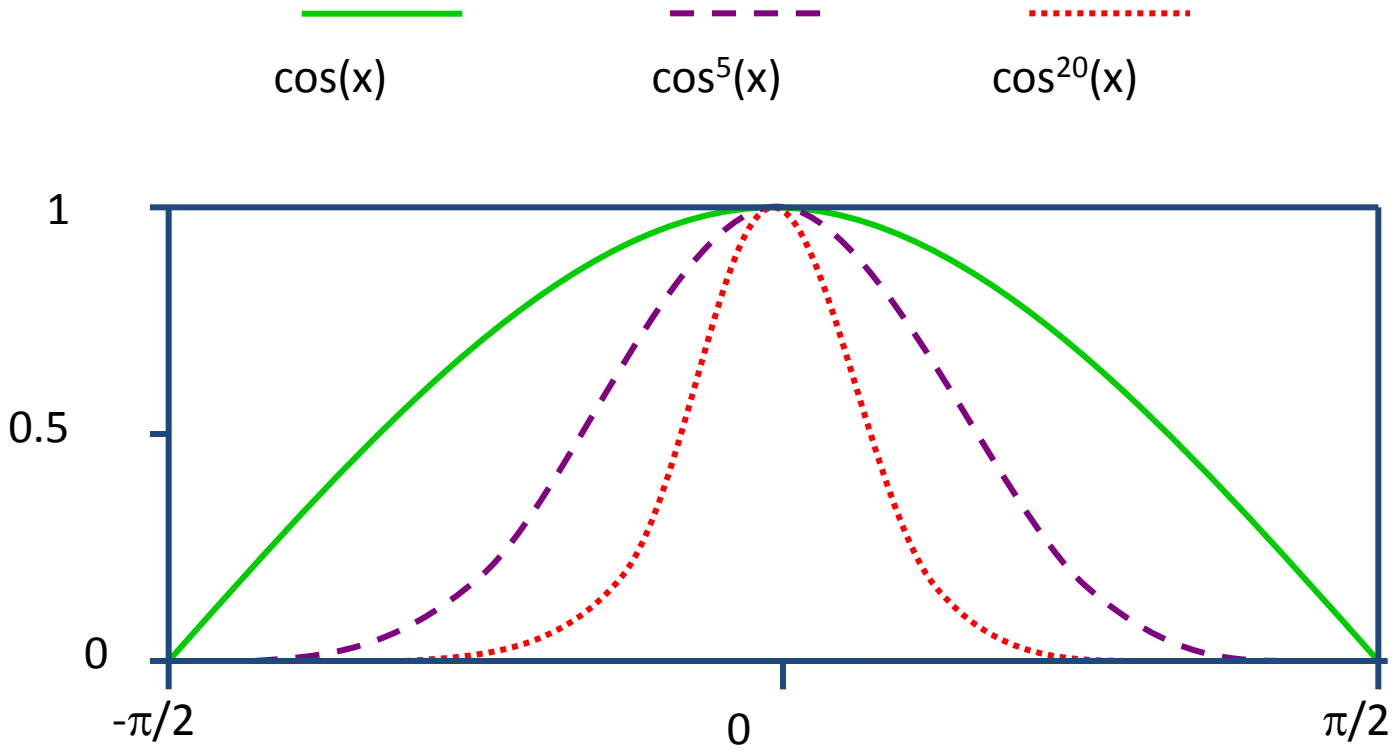

125
100
Recycling Facility Boarding

Spotlight II

- Maximale Intensität $I_{s,max}$, Punkt $\boldsymbol{p_s}$ und Richtung $\boldsymbol{l_s}$(norm)
- Maximaler Öffnungswinkel: θ
- Abschwächungskoeffizient: e
- Dann gilt für den beleuchteten Punkt $\boldsymbol{p}$:
  - $I_S(\boldsymbol{l}) = \begin{cases} I_{s,max} \cdot (\cos(\phi))^e, falls\ \phi \leq \theta \\ 0, \qquad\qquad sonst \end{cases}$, mit:
  - $\boldsymbol{l}$: Normierter Vektor von $\boldsymbol{p_s}$ zu $\boldsymbol{p}$
  - ϕ: Winkel zwischen $\boldsymbol{l}$ und $\boldsymbol{l_s}$
  - $\cos(\phi) = max(do\,t(\boldsymbol{l}, \boldsymbol{l_s}), 0)$



Spotlight III



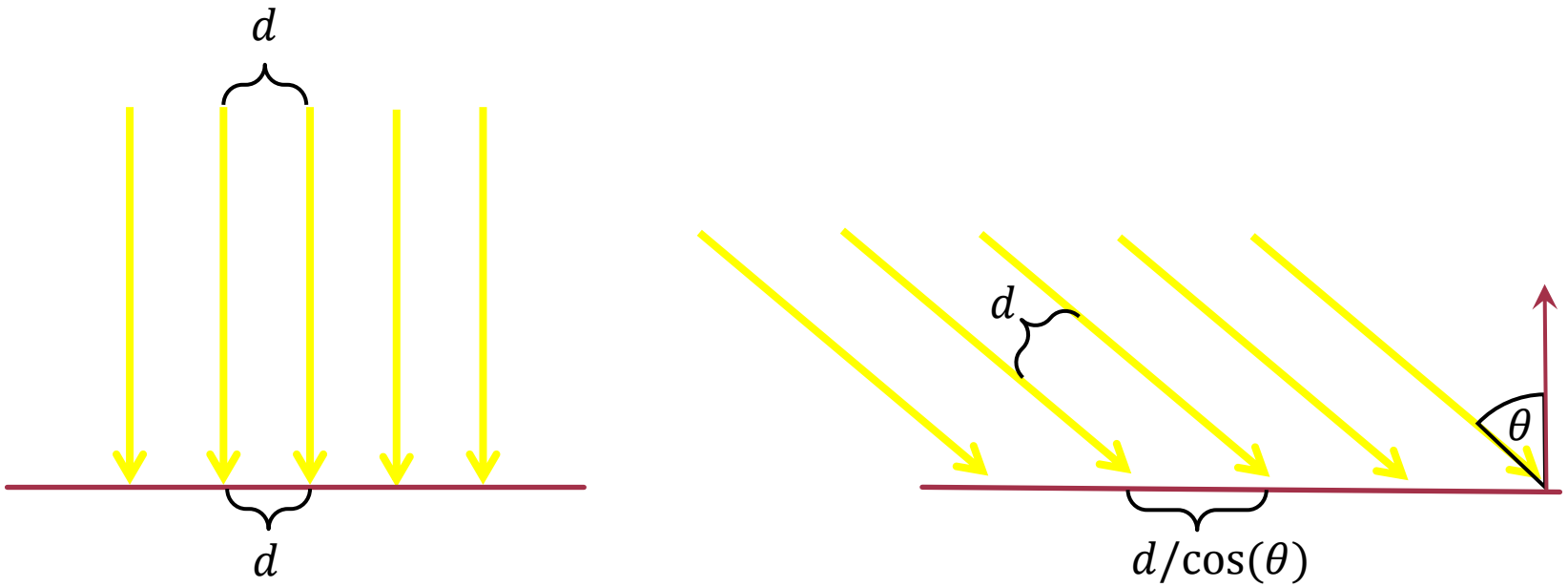
Ambient Light

- Bislang Szene ausschließlich direkt beleuchtet
- Rest unsichtbar
- Ambienter Term: Global konstante Lichtquelle
  - $I_a = (I_{a,r}, I_{a,g}, I_{a,b})$
- Keine Richtung und Position
- Sehr grobe Näherung an indirektes Licht

14.2

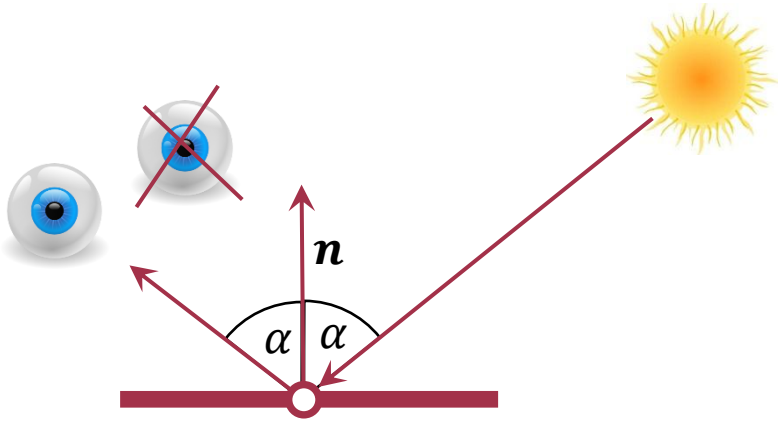
Grundlagen Reflexion & BRDF

Einfallendes Licht

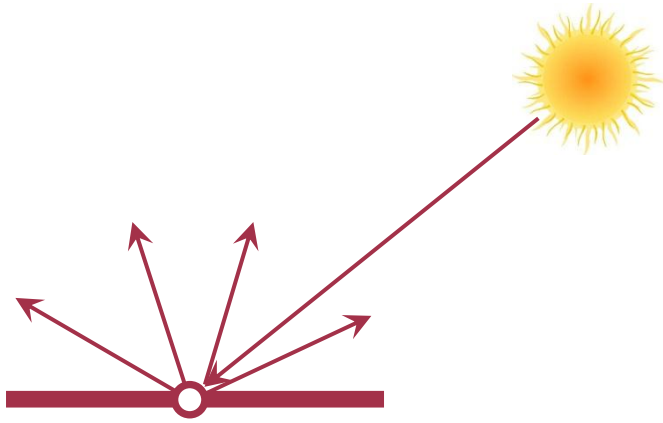


Einfallendes Licht $\propto \cos(\theta)$

Reflexion



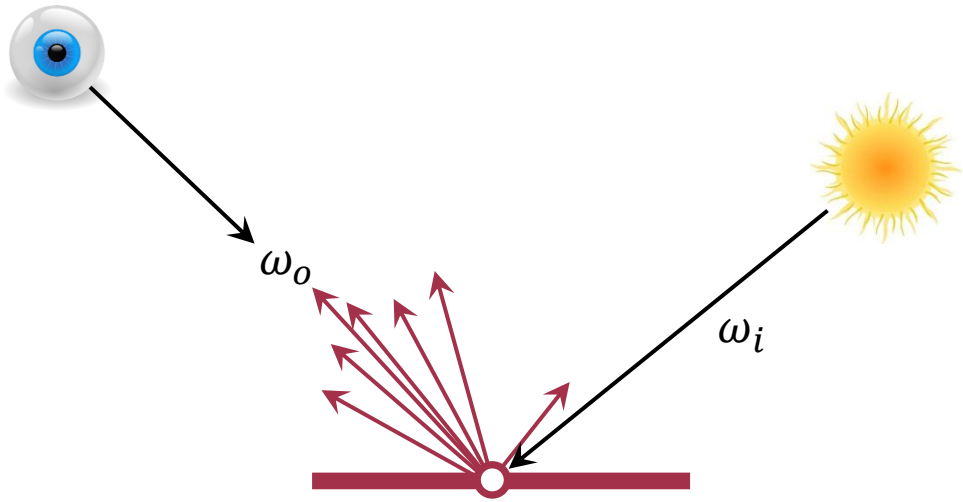
- Perfekter Spiegel
- Einfalls- = Ausfallsrichtung
- Andere Richtungen: Nichts

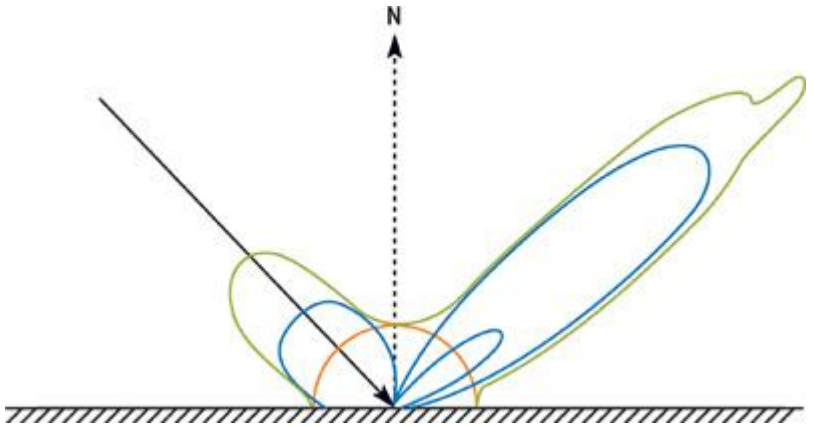


- Perfekte diffuse Oberfläche
- In alle Richtungen exakt gleich viel reflektiert
- Begründung: Oberfläche aus winzigen, zufällig orientierten Spiegeln

Reflexion realer Materialien

- Zwischen den Extremen vollständig diffus und spiegelnd
- Bestimmung in der Praxis:
  - Bestrahle Materie aus allen Winkeln ω_i
  - Miss Reflexion aus allen Winkeln ω_o
  - Ergibt Näherung der Bidirection Reflectance Distribution Function (BRDF): $f(\omega_i, \omega_o)$ des Materials



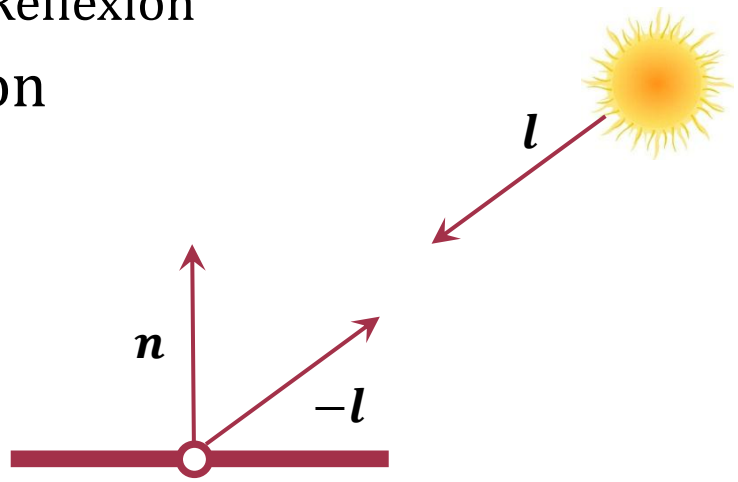


14.3

Phong BRDF

Diffuse Reflexionskomponente

- $R_{dif} = I_e \cdot k_{dif} \cdot C_{m,dif} \cdot \cos(-\boldsymbol{l}, \boldsymbol{n})$
  $= I_e \cdot k_{dif} \cdot C_{m,dif} \cdot max(do\,t(-\boldsymbol{l}, \boldsymbol{n}), 0)$, mit:
  - **l** Lichtrichtung (normiert). Nicht unbedingt die der Lichtquelle
  - **n** Normale des beleuchteten Oberflächenpunkts (normiert)
  - I_e: Einfallendes Licht, z.B. gerichtetes Licht einer Lichtquelle, Folie 8
  - $C_{m,dif}$ Beleuchtungsunabhängige Materialfarbe, z.B. Wert aus Farbtextur, für diffuse Reflexion
  - k_{dif} Gewicht für diffuse Reflexion
- Kennen wir, mit $I_e = I_d$, schon



FS: Diffuse Reflexion / 1 gerichtete LQ

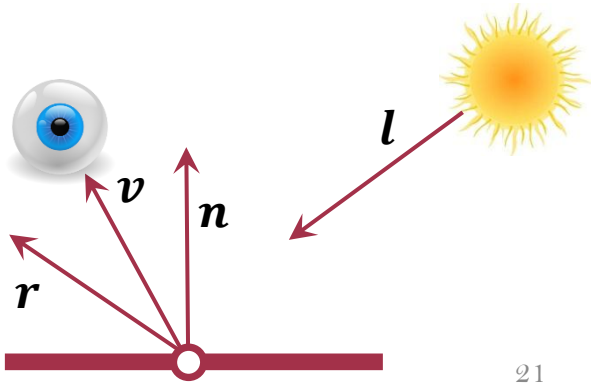
```
#version 150 core
in vec3 normalWC;
out vec4 phongColor;
vec4 phongDiff(vec4 I_e, vec3 l_in, vec3 n){
  // Materialeigenschaften für diffuse Reflexion
  float k_dif = 1;                    // Gewicht für diffuse Reflexion
  vec4 C_m_dif = vec4(0,1,0,1);       // Diffuse Farbe

  // Berechne diffuse Beleuchtung
  vec4 R_dif = I_e * k_dif * C_m_dif * max(dot(-l_in, n), 0.0);
  return R_dif;
}
void main() {
  // Eigenschaften des gerichteten Lichts
  vec4 I_d = vec4(1,1,1,1);           // Farbe bzw. Intensität
  vec3 l_d = vec3(-1,0,0);            // Richtung

  vec3 n = normalize(normalWC);
  phongColor = phongDiff(I_d, l_d, n);
}
```

Spekulare Reflexionskomponente

- $R_{spec} = I_e \cdot k_{spec} \cdot C_{m,spec} \cdot \cos(reflect(\boldsymbol{l}, \boldsymbol{n}), \boldsymbol{v})^{es}$

  $= I_e \cdot k_{spec} \cdot C_{m,spec} \cdot \cos(\boldsymbol{r}, \boldsymbol{v})^{es}$

  $= I_e \cdot k_{spec} \cdot C_{m,spec} \cdot max(dot(\boldsymbol{r}, \boldsymbol{v})^{es}, 0)$, mit:
  - $\boldsymbol{l}$ Lichtrichtung
  - $\mathbf{n}$ Normale des beleuchteten Oberflächenpunkts
  - $\boldsymbol{r}$ An $\boldsymbol{n}$ reflektierte Lichtrichtung
  - $\boldsymbol{v}$ Vektor zum Betrachter
  - I_e: Einfallendes Licht, z.B. gerichtetes Licht einer Lichtquelle, Folie 8
  - $C_{m,spec}$ Beleuchtungsunabhängige Materialfarbe, z.B. Wert aus Farbtextur, für spekulare Reflexion
  - k_{spec} Gewicht für spekulare Reflexion
  - es Spekularer Exponent
- Hinweis: Alle Vektoren normiert



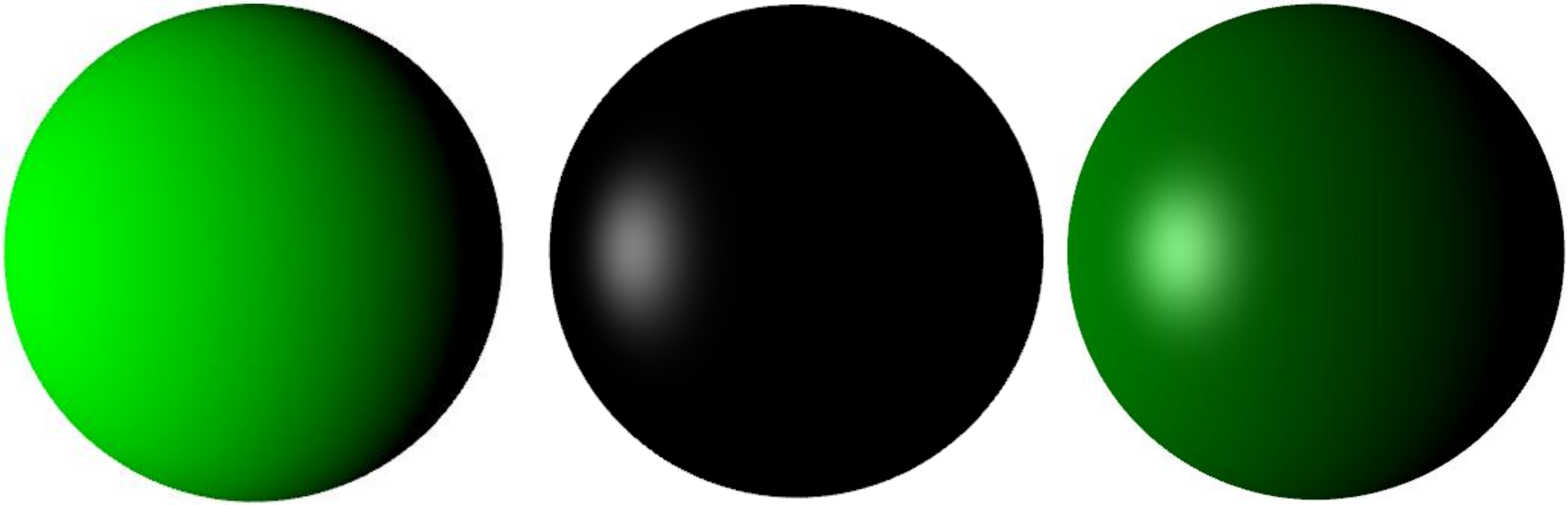
FS: Spekulare Reflexion / 1 gerichtete LQ

```
#version 150 core
in vec3 normalWC, positionWC;
uniform vec3 eyePosition;
out vec4 phongColor;

vec4 phongSpec(vec4 I_e, vec3 l_in, vec3 n, vec3 posWC, vec3 eye){
  float k_spec = 1.0;              // Gewicht für spekulare Reflexion
  vec4 C_m_spec = vec4(1,1,1,1); // Spekulare Farbe
  float es = 12.0;                 // Spekularer Exponent
  vec3 r = reflect(l_in, n);
  vec3 v = normalize(eye - posWC);
  return I_e * k_spec * C_m_spec * max(pow(dot(r, v), es), 0.0);
}

void main() {
  vec4 I_d = vec4(1,1,1,1);        // Lichtfarbe bzw. Intensität
  vec3 l_d = vec3(-1,0,0);         // Lichtrichtung
  vec3 n = normalize(normalWC);
  phongColor =  phongSpec(I_d, l_d, n, positionWC, eyePosition);
}
```

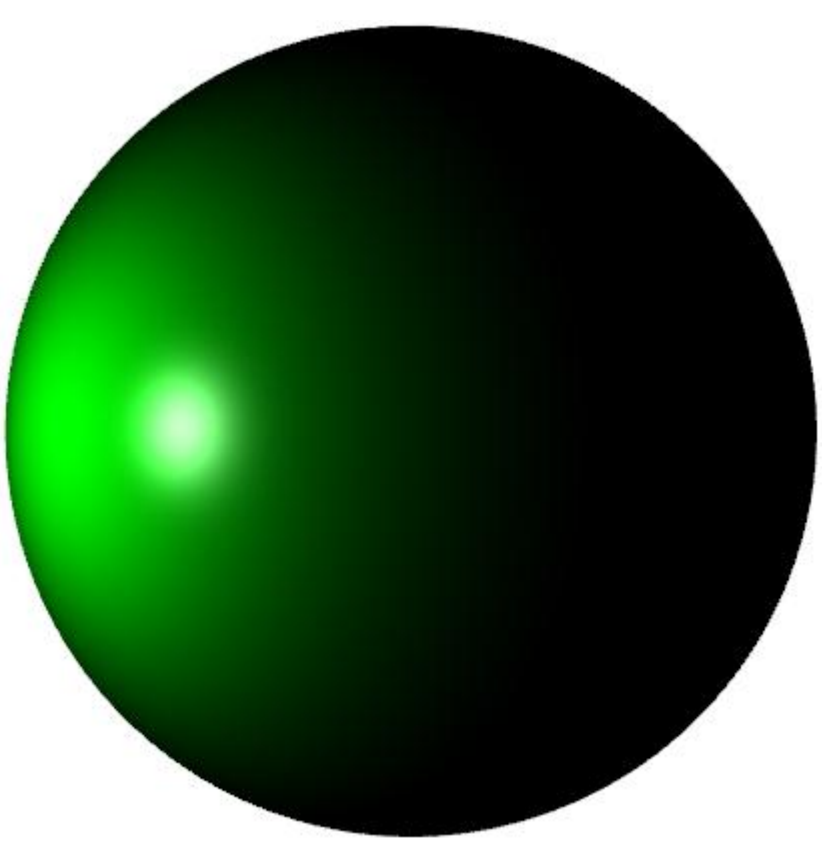
FS: Spek. & diff Reflexion / 1 ger. LQ

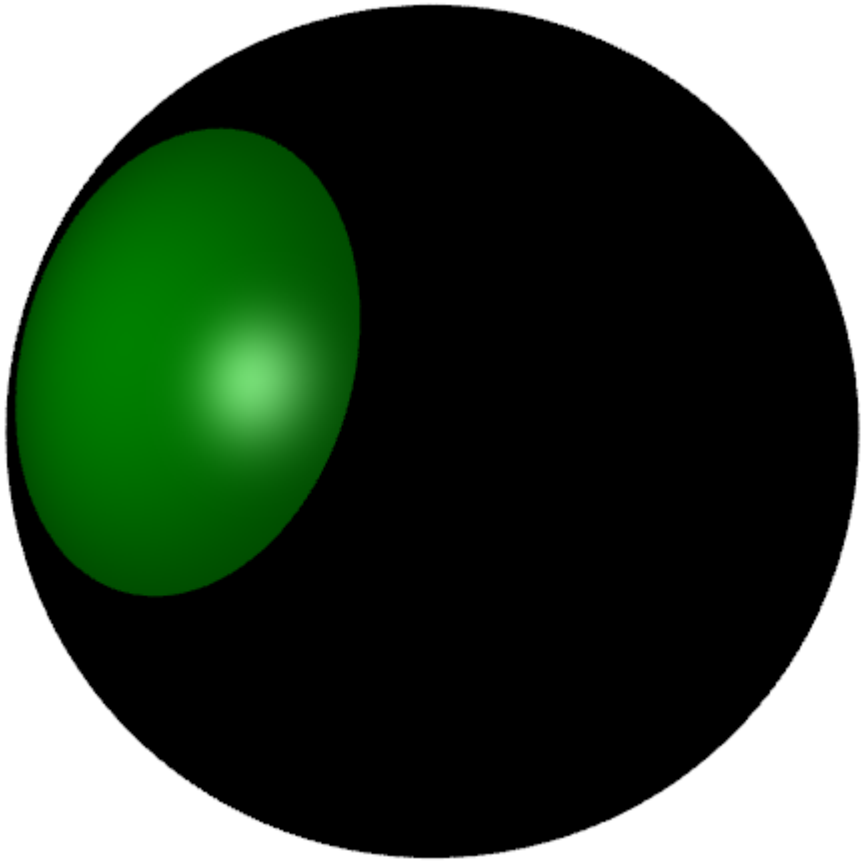
```
#version 150 core
in vec3 normalWC, positionWC;
uniform vec3 eyePosition;
out vec4 phongColor;

vec4 phongSpec(...){...} // Folie 20 Aber mit k_spec = 0.5;

vec4 phongDiff(...){...} // Folie 22 Aber mit k_dif = 0.5;

void main() {
  vec4 I_d = vec4(1,1,1,1);       // Lichtfarbe bzw. Intensität
  vec3 l_d = vec3(-1,0,0);        // Lichtrichtung
  vec3 n = normalize(normalWC);

  phongColor =  phongDiff(I_d, l_d, n)
              +phongSpec(I_d, l_d, n, positionWC, eyePosition);
}
// Andere Lichtquelle, z.B. PointLight?
// Zunächst Parameter I_e und l_in der Methoden ausrechen.
```

Beispiele mit gerichtetem Licht



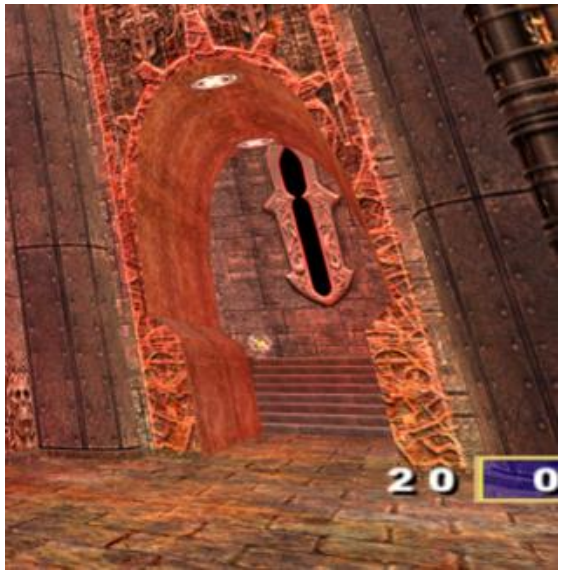
Pointlight & Spotlight



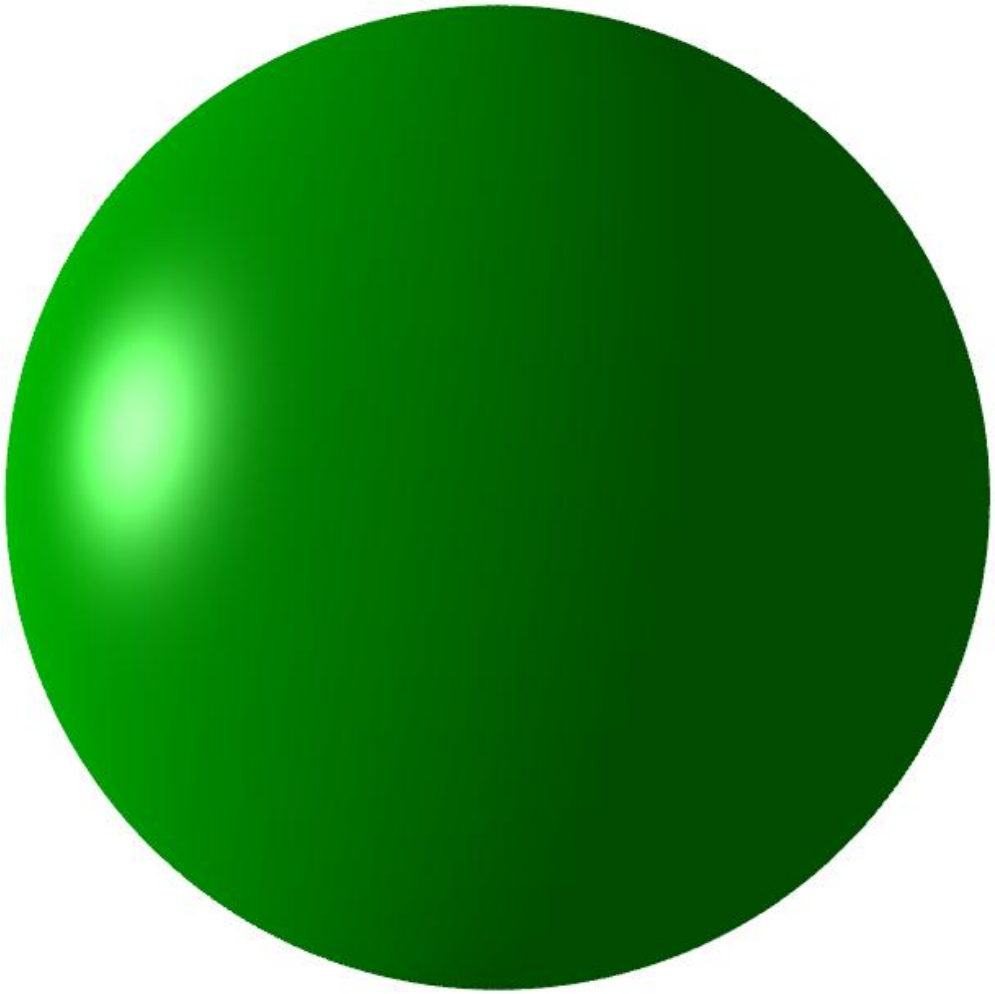


Ambiente Reflexion

- “Grundhelligkeit“ zum „Stopfen“
- $R_a = I_a \cdot k_a \cdot C_m$, mit:
  - I_a Ambientes Licht, Folie 13
  - C_m Beleuchtungsunabhängige Materialfarbe, z.B. Wert aus Farbtextur
  - k_a Gewicht für ambiente Reflexion



Beispiel: Diffus + Spekular + Ambient



Materialeigenschaften

- Bedingung: Nicht mehr Reflektieren als empfangen
  - $0 \leq k_{spec}, k_{dif}, k_a \leq 1$
  - $k_{spec} + k_{dif} + k_a \leq 1$
- Modellierung, z.B.:
  - Matt: $k_{dif} \gg k_{spec}$
  - Spiegelnd: $k_{spec} \gg k_{dif}$
  - Kontrastarm, “unräumlich“: $k_a \gg k_{dif}, k_{spec}$
- Finden brauchbarer Parameter, etwa für Holz:
  - Eher Matt…
  - …Google, etc.

Gesamtberechnung

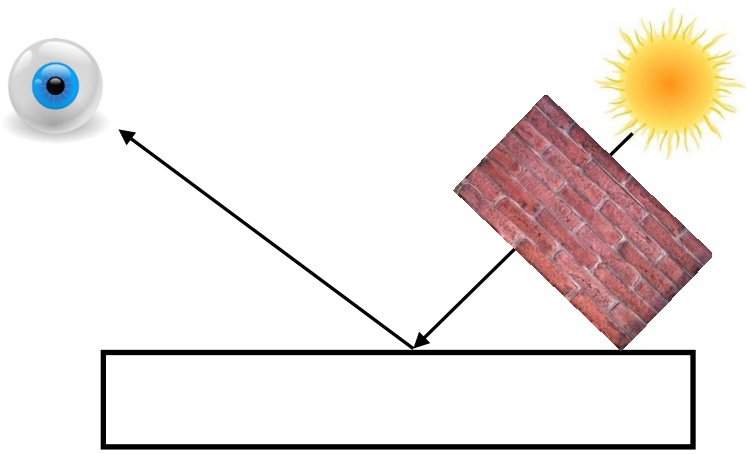
- Das gesamte reflektierte Licht ergibt sich bei n Lichtquellen:
  - $R_{phong} = R_a$ Ein ambienter Term
    $+ \sum_0^{n-1} R_{dif}(I_{e,n}, \boldsymbol{l_n})$ Diffuse Beiträge der n Lichtquellen
    $+ \sum_0^{n-1} R_{spec}(I_{e,n}, \boldsymbol{l_n})$ Spekulare Beiträge der n Lichtquellen
- Diese Gleichung muss für jeden Oberflächenpunkt, etwa Fragment oder Vertex, ausgewertet werden

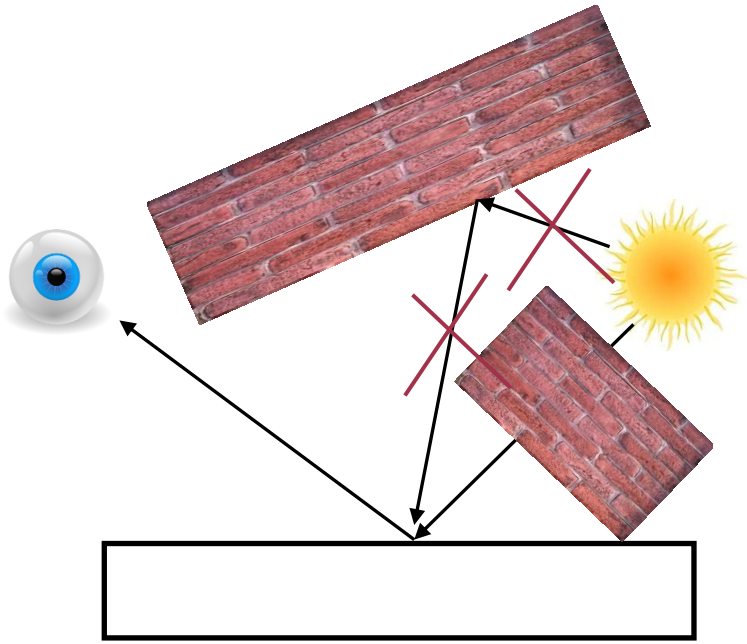
Deferred Shading

- Typische Szene:
  1. Viele Fragments erst beleuchtet
  2. Dann in Framebuffer geschrieben
  3. Später durch andere, näher liegende, Geometrie überschrieben
  - Unnötig beleuchtet!
- Ideen?
  1. Schreibe erst alle Fragments mit für Berechnung nötige Daten(Normale, Objektfarbe, etc.) in FB
  2. Dann liegen lediglich Daten sichtbaren Fragments vor
  3. Führe Berechnungen dann auf diesem 2D-Raster aus
- Problem: Transparenz

Grenzen

- Schatten
- Indirektes Licht
- Ausgedehnte Lichtquelle





Toon Shader