

# Interaktive Computergrafik



**Prof. Dr. Frank Steinicke**  
Human-Computer Interaction  
Department of Computer Science  
University of Hamburg

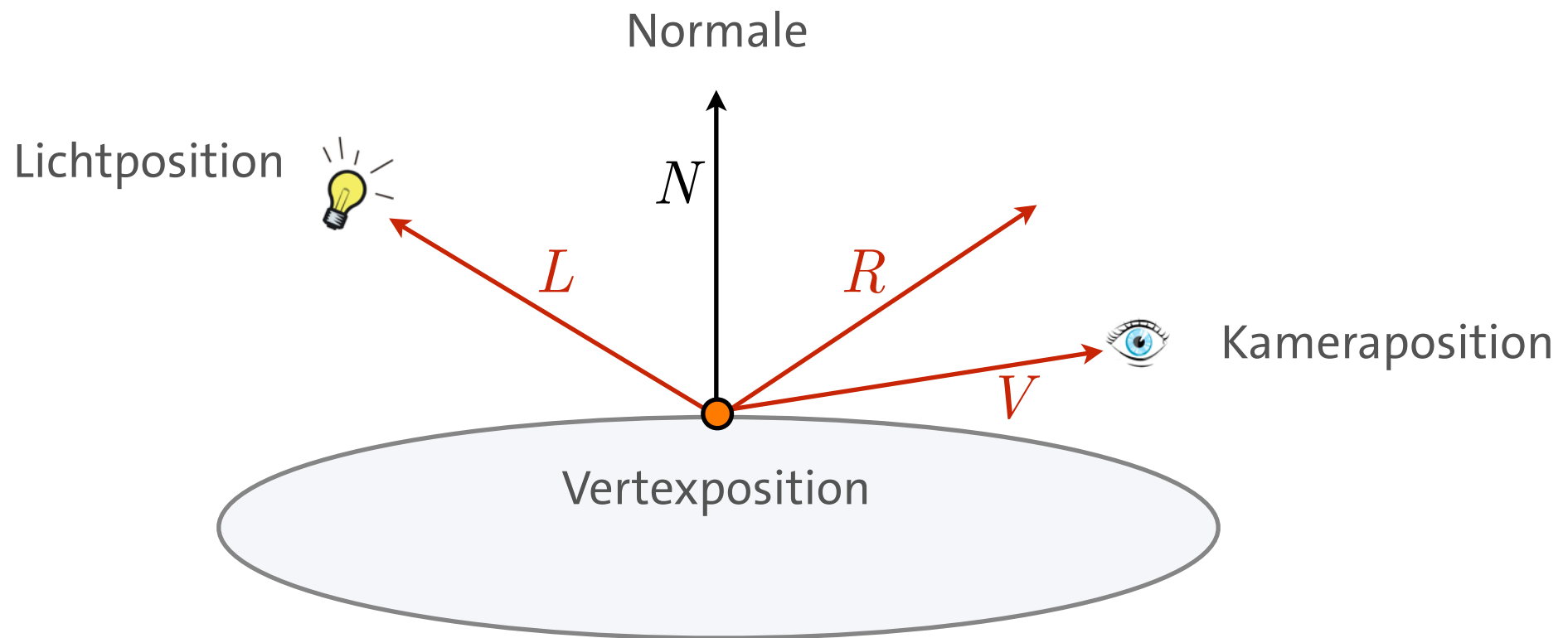


# Interaktive Computergrafik

## Übung - Woche 7

# Beleuchtungsgleichung

$$I = \underbrace{I_a \cdot k_a}_{\text{ambient}} + \sum_{i=1}^m f_{att_i} \cdot \underbrace{(I_{d_i} \cdot k_d \cdot \max(0, N \cdot L_i))}_{\text{diffus}} + \underbrace{I_{s_i} \cdot k_s \cdot \max(0, R_i \cdot V)^n}_{\text{spekular}}$$



# Beleuchtungsgleichung

## Implementierung

$$I = I_a \cdot k_a + \sum_{i=1}^m f_{att_i} \cdot (I_{d_i} \cdot k_d \cdot \max(0, N \cdot L_i) + I_{s_i} \cdot k_s \cdot \max(0, R_i \cdot V)^n)$$

- Schritt 1: Alle **Koeffizienten** als **Uniform-Variablen** an Shader übergeben.
- Schritt 2: Für jeden Vertex **Normale** definieren und in VBO an Shader übergeben.

# Beleuchtungsgleichung

## Implementierung

$$I = I_a \cdot k_a + \sum_{i=1}^m f_{att_i} \cdot (I_{d_i} \cdot k_d \cdot \max(0, N \cdot L_i) + I_{s_i} \cdot k_s \cdot \max(0, R_i \cdot V)^n)$$

- Schritt 3: Überführung der gegebenen Variablen in **Kamerakoordinaten**.
- Schritt 4: Berechnung der **normalisierten Vektoren N, L, R und V** aus gegebenen Variablen in Kamerakoordinaten.

# Hinweise

## Kamerakoordinaten

- Ausgangskoordinatensystem beachten  
→ unterschiedlich für z.B. Vertexpositionen bzw. -normalen (Objektkoordinaten) und Lichtposition (Weltkoordinaten)
- Überführung in Kamerakoordinaten durch Multiplikation mit passender Matrix (Model- und/oder View-Matrix)  
→ siehe Online-Diskussion zur Lektion 7

# Hinweise

## Punkt- vs. Richtungsvektoren

- Licht-, Vertex- und Kameraposition sind 3D-Punkte
- Phong-Beleuchtungsgleichung arbeitet mit Richtungsvektoren  $L$ ,  $N$ ,  $V$  und  $R$   
→ Umrechnung notwendig

# Hinweise

## Normalentransformation

- Vertexpositionen und -normalen müssen zwar jeweils von Objekt- in Kamerakoordinaten transformiert werden, trotzdem unterscheiden sich Transformationsmatrizen  
→ siehe Online-Diskussion zur Lektion 7
- $V_i' = T \cdot V_i$
- $N_i' = (T^T)^{-1} \cdot N_i$



# Hinweise

## Abschwächungsfaktor

$$f_{att} = \min \left( \frac{1}{c_1 + c_2 \cdot d + c_3 \cdot d^2}, 1 \right)$$

- $c_1$ ,  $c_2$  und  $c_3$  sind Konstanten, mit denen Stärke der Lichtabschwächung gesteuert werden kann  
→ geeignete Werte können aus Aufgabe 1 des Übungszettels übernommen werden

# Hinweise

## Reflexionsvektor

- GLSL stellt Built-In-Funktion für Reflexion an einer Normalen zur Verfügung:  
`reflect(Einfallsvektor, Normale)`
- Vorsicht:  
L zeigt von Vertex zu Lichtquelle, bei `reflect` ist jedoch **Einfallsvektor** gefordert

# Hinweise

## Weitere GLSL-Hilfsfunktionen

- `max(x, y), pow(x, y)` → *beide Parameter müssen selben Typ haben!*
- `normalize(Vektor)`  
(= Normalisierung des Vektors auf Länge 1)
- `dot(Vektor1, Vektor2)`  
(= Skalarprodukt)
- `inverse(Matrix), transpose(Matrix)`

*Können nur in Shadern verwendet werden!  
(nicht in JavaScript-Anwendungsprogramm)*

# Hinweise

## Dimensionalität I

- Alle Materialkoeffizienten  $k_x$  sowie Intensitäten  $I_x$  sind vec3 mit Werten für R, G und B (keine skalaren Werte!)

# Hinweise


## Dimensionalität II

- Vektoren N, L, R und V sind entweder vec3, oder vec4 mit homogener Komponente = 0 (d.h. Richtungsvektoren)
- Beispiel:
  - ✓  $(1, 0, 0)^T$  wird zu  $(1, 0, 0, 0)^T \xrightarrow{\text{Normalisierung}} (1, 0, 0, 0)^T$
  - ✗  $(1, 0, 0)^T$  wird zu  $(1, 0, 0, 1)^T \xrightarrow{\text{Normalisierung}} (\frac{1}{\sqrt{2}}, 0, 0, \frac{1}{\sqrt{2}})^T$
- Fehler durch inkorrekte Homogenisierung

# Hinweise

## Multiplikationen

$$I = I_a \cdot k_a + \sum_{i=1}^m f_{att_i} \cdot (I_{d_i} \cdot k_d \cdot \max(0, N \cdot L_i) + I_{s_i} \cdot k_s \cdot \max(0, R_i \cdot V)^n)$$



- $I_x \cdot k_x$  sind **komponentenweise Produkte**
- $N \cdot L_i$  und  $R_i \cdot V$  sind **Skalarprodukte**

