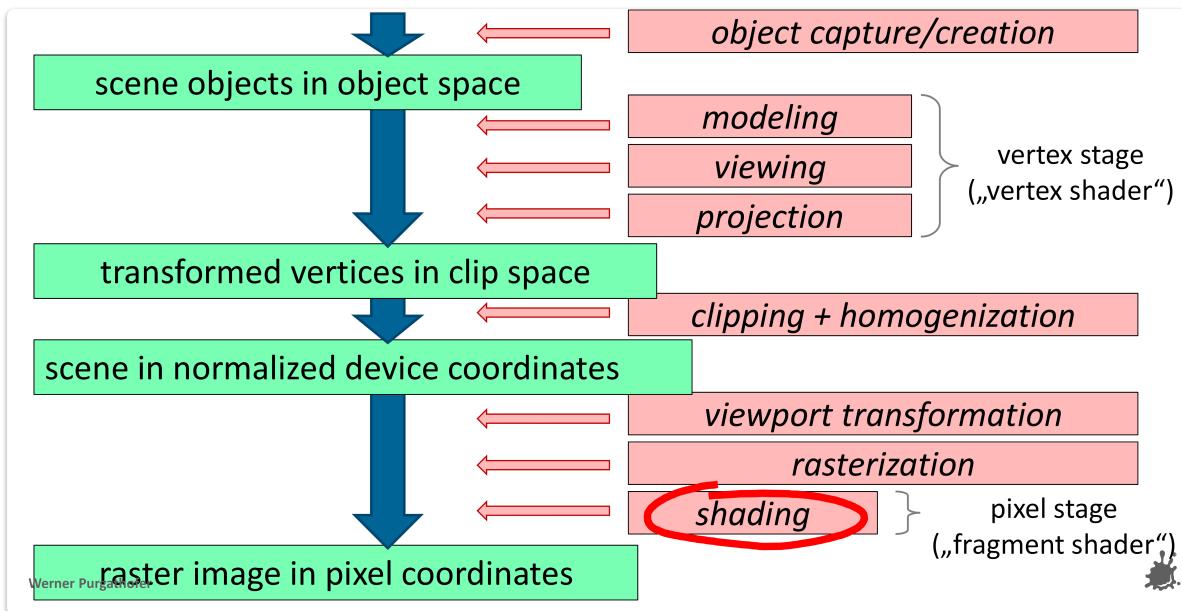


9. Beleuchtung und Schattierung

EVC_Skriptum_CG, p.35

- **Definition:** Ein Beleuchtungsmodell (oder Schattierungsmodell, Illumination/Lighting/Shading Model) berechnet die wahrgenommene Farbe/Helligkeit eines Objekts für den Betrachter basierend auf:
 - Lichtverhältnissen in der Szene.
 - Oberflächeneigenschaften des Objekts.
- **Ziel:** Bestimmung der Farbe, die das entsprechende Pixel im Bild erhalten soll.
- **Bedeutung:** Zusammen mit der perspektivischen Projektion der wichtigste Faktor für realistisch aussehende Computergraphik-Bilder.
- **Vereinfachung:** Die folgenden Betrachtungen und Formeln konzentrieren sich zunächst auf die **Helligkeit** der Beleuchtung.
- **Farbbehandlung:** Um Farben zu berücksichtigen, müssen die Berechnungen für verschiedene Wellenlängen durchgeführt werden (im einfachsten Fall für Rot, Grün und Blau).



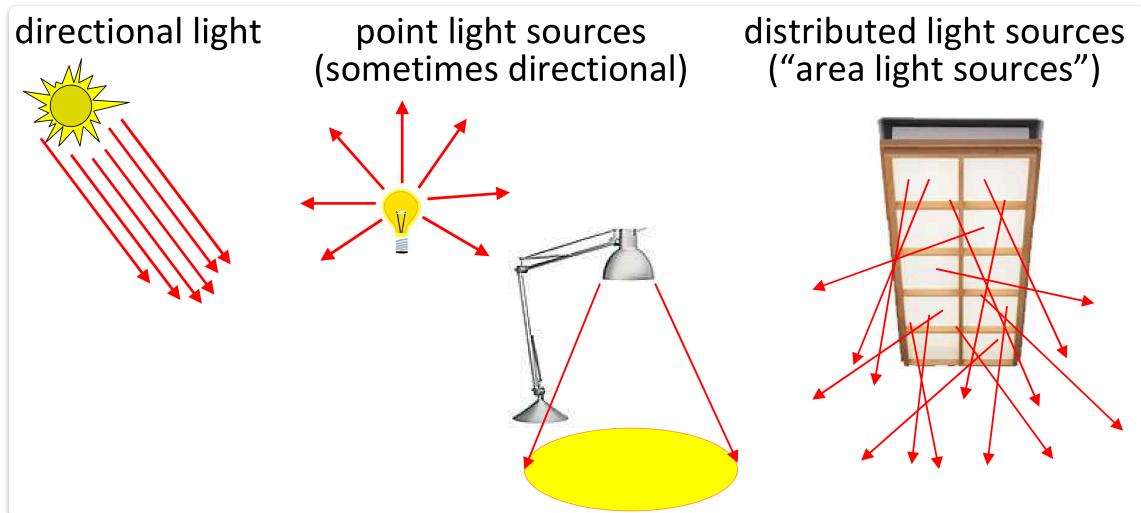
Lichtquellen und Oberflächen

Lichtquellen

EVC_Skriptum_CG, p.35

- **Notwendigkeit:** Voraussetzung zur Berechnung von Beleuchtungseffekten in einer Szene.
- **Merkmale von Lichtquellen:**
 - **Form:**

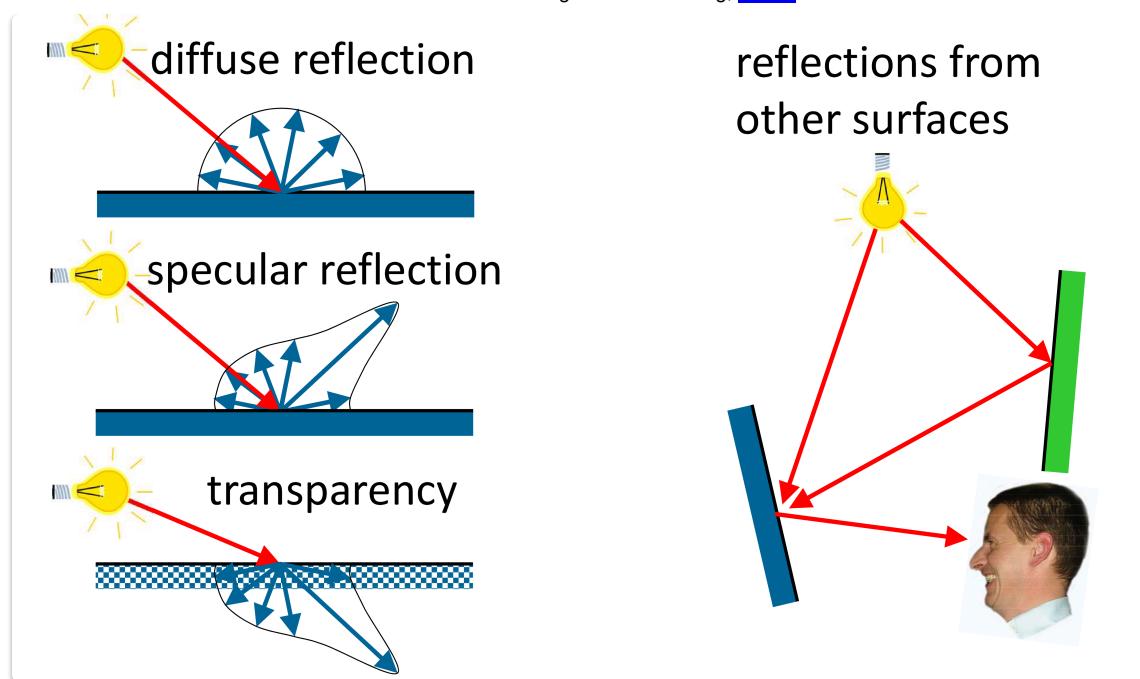
- Lichtrichtung (z.B. paralleles Sonnenlicht)
- Punktlichtquellen (strahlen Licht von einem einzelnen Punkt in alle Richtungen)
- Gerichtete Punktlichtquellen (kombinieren Punktlichtquelle mit einer kegelförmigen oder rechteckigen Lichtverteilung)
- Flächige Lichtquellen (emittieren Licht von einer ausgedehnten Oberfläche)
- ... (weitere Formen sind möglich)
- **Eigenschaften:**
 - Helligkeit (Intensität des Lichts)
 - Farbe (Spektrale Zusammensetzung des Lichts)
 - Entfernung (bei einigen Lichtquellen relevant für die Intensitätsabnahme)
 - ... (weitere Eigenschaften können definiert werden)



Objektoberflächen

EVC_Skriptum_CG, p.35

- **Wechselwirkung mit einfallendem Licht:** Oberflächen können Licht auf verschiedene Weisen beeinflussen:
 - **Diffuse Reflexion:** Licht wird in alle Richtungen gleichmäßig reflektiert (Beispiele: Papier, Kreide).
 - **Spiegelnde Reflexion:** Licht wird bevorzugt in die Spiegelungsrichtung reflektiert (Beispiele: Lack, Metall).
 - **Transparenz:** Licht durchdringt die Oberfläche und tritt auf der anderen Seite wieder aus (Beispiele: Glas, Wasser).
- **Realität:** Die meisten Oberflächen weisen eine Kombination dieser Eigenschaften auf.
- **Indirekte Beleuchtung:** Es ist wichtig zu beachten, dass Licht nicht nur direkt von Lichtquellen auf Oberflächen trifft, sondern auch von anderen Oberflächen reflektiert wird und somit zur Beleuchtung beiträgt.



Ein einfaches Beleuchtungsmodell

EVC_Skriptum_CG, p.35

- **Hintergrund:** Die physikalisch genaue Simulation von Licht und seiner Interaktion mit Oberflächen ist sehr aufwendig.
- **Ansatz in der Praxis:** Verwendung vereinfachter, empirischer Beleuchtungsmodelle.
- **Grundstruktur (ungefähre Darstellung):** (Die genaue Struktur wird in den folgenden Abschnitten detaillierter erläutert.)
- **Ziel:** Eine visuell plausible Beleuchtung mit überschaubarem Rechenaufwand zu erzielen.

Hintergrundlicht (Ambientes Licht)

EVC_Skriptum_CG, p.35

- **Realitätsbezug:** Objekte strahlen einen Teil des auf sie treffenden Lichts ab, wodurch es auch in Bereichen ohne direkte Beleuchtung nicht vollständig dunkel ist.
- **Definition:** Dieses überall vorhandene, indirekte Basislicht wird als **ambientes Licht** oder **Hintergrundlicht** bezeichnet.
- **Implementierung in einfachen Beleuchtungsmodellen:** Ein konstanter Helligkeitswert (I_a) wird zu jeder Beleuchtungsberechnung addiert, um diesen globalen Lichtanteil zu approximieren.

- ambient light (background light) I_a

- constant over a surface
- independent of viewing direction
- diffuse-reflection coefficient k_d ($0 \leq k_d \leq 1$)
- approximation of global diffuse lighting effects

$$L_{\text{ambdiff}} = k_d I_a$$



Lambert'sches Gesetz (Diffuse Reflexion)

[EVC_Skriptum_CG, p.35](#), [EVC_Skriptum_CG, p.36](#)

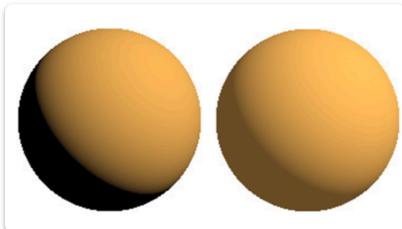
- **Kernaussage:** Die Helligkeit einer diffus reflektierenden Oberfläche ist proportional zum Kosinus des Winkels zwischen der Oberflächennormale und der Richtung zur Lichtquelle. Flacher Lichteinfall führt zu dunkleren Oberflächen.
- **Bedeutung:** Erzeugt den Eindruck räumlicher Form durch Helligkeitsvariationen.
- **Formel für die resultierende Helligkeit (L) durch diffuse Reflexion:**

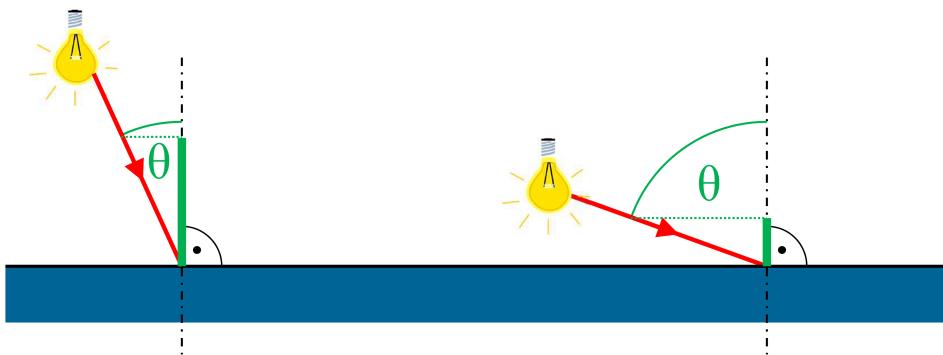
$$L = k_d \cdot I \cdot \cos \theta$$

oder in Vektorform:

$$L = k_d \cdot I \cdot (\mathbf{n} \cdot \mathbf{l})$$

- I : Helligkeit der relevanten Lichtquelle.
- k_d : Diffuser Reflexionskoeffizient der Oberfläche ($0 \leq k_d \leq 1$), gibt den Anteil des einfallenden Lichts an, der diffus reflektiert wird.
- θ : Winkel zwischen der Oberflächennormale (\mathbf{n}) und der Richtung zur Lichtquelle (\mathbf{l}).
- $\mathbf{n} \cdot \mathbf{l}$: Skalarprodukt zwischen dem Normalenvektor und dem Lichtrichtungsvektor.
- **Kombination mit ambientem Licht:**
 - Gesamte Helligkeit = Beitrag durch diffuse Reflexion + Beitrag durch ambientes Licht (I_a).
 - Führt bereits zu einer ansprechenden Darstellung (siehe Beispiel der Kugeln)





$$L = I \cdot \cos \theta$$

when considering
the material:

I ... light source intensity

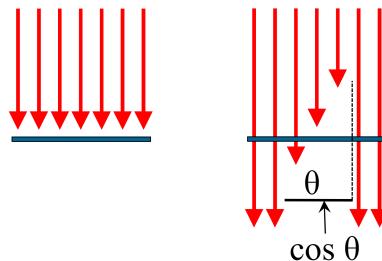
L ... pixel color

k_d ... diffuse coefficient

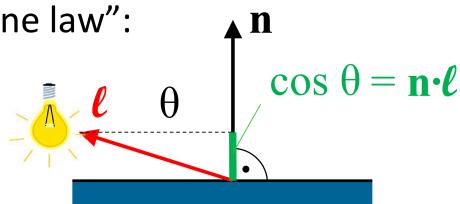
$$L = k_d \cdot I \cdot \cos \theta$$

for ideal diffuse reflectors (Lambertian reflectors)

brightness depends on
orientation of the surface:



"Lambert's cosine law":



$$L_{\text{diff}} = k_d \cdot I \cdot (\mathbf{n} \cdot \mathbf{l})$$

total diffuse reflection:

$$L_{\text{diff}} = k_a I_a + k_d I(\mathbf{n} \cdot \mathbf{l})$$

$$k_d=0 \quad k_d=0.3 \quad k_d=0.7 \quad k_d=1$$



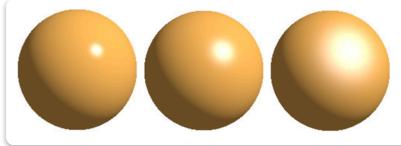
(sometimes extra k_a
for ambient light)

Werner Purgathofer

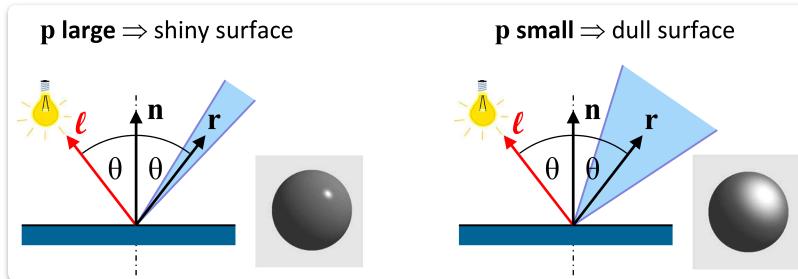
Glanzpunkte (Specular Highlights)

EVC_Skriptum_CG, p.36

- Fast jede Oberfläche ist auch etwas spiegelnd. Wenn man diesen Aspekt nicht mitmodelliert, dann wirken alle Materialien gleich stumpf.



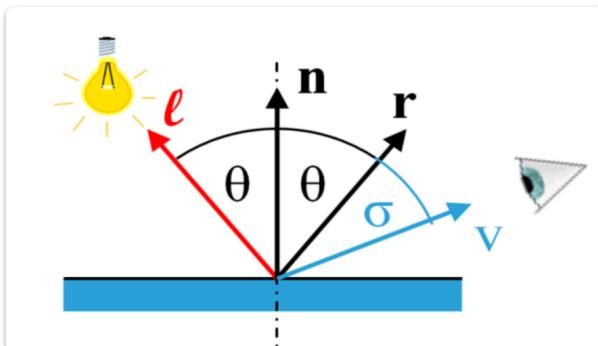
- Da die exakte spiegelnde Reflexion äußerst kompliziert zu berechnen ist, behilft man sich mit einer einfachen Funktion, die einen ähnlichen Verlauf hat wie das Highlight: $\cos^p(\alpha)$.
- Mit dem freien Parameter p lässt sich dabei die „Poliertheit“ der Oberfläche steuern:
 - Je größer p ist, desto kleiner wird der Glanzpunkt und desto glatter wirkt die Oberfläche (linke Kugel im Bild).
 - Je kleiner p ist, desto matter wirkt die Oberfläche (rechte Kugel im Bild).



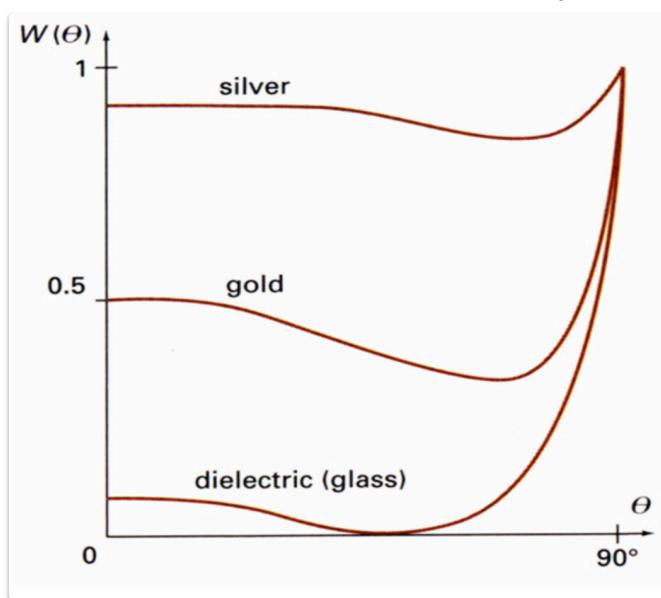
- Um diesen Effekt im richtigen Ausmaß zur Beleuchtung hinzufügen zu können, wird noch ein weiterer Faktor eingeführt, der spiegelnde Reflexionskoeffizient k_s .
- Der Glanz berechnet sich dann nach diesem sogenannten **Phong-Beleuchtungsmodell** so:

$$I_{spec} = k_s * I_L * \cos^p(\alpha)$$

- I_L : Intensität des Lichts.
- α : Winkel zwischen dem exakten Reflexionsstrahl r und der Richtung zum Auge v .
- Etwas näher an der Wahrheit ist die Verwendung des **Fresnel'schen Reflexionsgesetzes**, das beschreibt, dass der Spiegelungsgrad auch vom Lichteinfallswinkel θ abhängt.
- Also ist der Koeffizient k_s eigentlich eine Funktion $W(\theta)$ der Lichteinfallsrichtung l .
- Für die meisten Materialien ist dieser Wert aber fast konstant. Daher wird auf diesen Aufwand verzichtet, wenn man nicht gerade ein Material darstellen will, bei dem der Effekt auffällt.

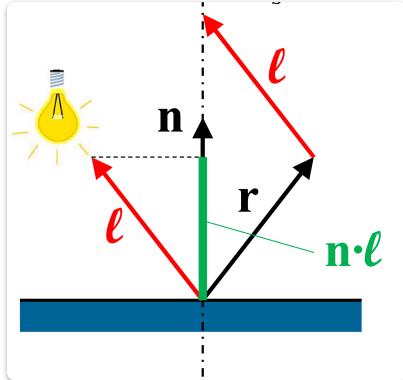


- Das Bild zeigt die Abhängigkeit dieser Funktion $W(\theta)$ vom Winkel zwischen Lichteinfall und Normale auf der Oberfläche für drei verschiedene Materialien (Silber, Gold, dielektrisches Material).



- Bei der Berechnung des Reflexionsvektors r muss man noch bedenken, dass es sich hier um **Vektoren im 3D-Raum** handelt, wobei l (Lichtrichtung), n (Oberflächennormale) und r in einer Ebene liegen müssen und alle Länge eins haben sollen.
 - r ergibt sich zu:

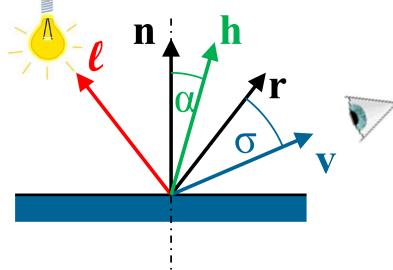
$$r = (2n \cdot l)n - l$$



- Weil die Glanzfunktion sowieso nur eine grobe Näherung ist, verwendet man auch häufig eine einfachere Formel, in der $r \cdot v$ (Winkel zwischen Reflexionsrichtung und Blickrichtung) durch $n \cdot h$ ersetzt wird.
 - h : Halbierungsvektor zwischen l und v .

simplified Phong model with halfway vector \mathbf{h}

$$L_{\text{spec}} = k_s \cdot I \cdot (v \cdot r)^p \quad \rightarrow \quad L_{\text{spec}} = k_s \cdot I \cdot (n \cdot h)^p$$



$$\mathbf{h} = \frac{\ell + \mathbf{v}}{\|\ell + \mathbf{v}\|}$$

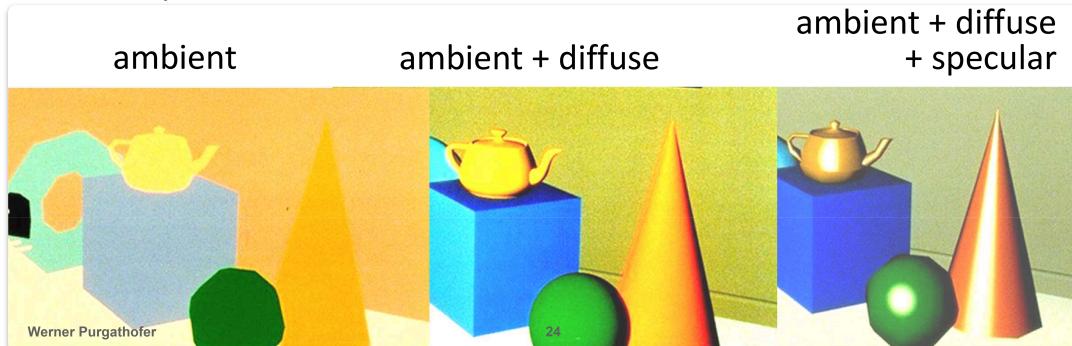
this revised model is called
“Blinn-Phong Shading”

- Der Winkel zwischen n und h ist oft sehr ähnlich dem Winkel zwischen r und v .

- Das resultierende Modell nennt man **Blinn-Phong-Beleuchtungsmodell**.
- Wenn wir alle bisherigen Komponenten zusammensetzen, erhalten wir ein einfaches komplettes Beleuchtungsmodell:

$$L = k_a * I_a + \sum_{i=1,\dots,N} (k_d * I_i * (n \cdot l_i) + k_s * I_i * (n \cdot h_i)^p)$$

- L : Gesamte Beleuchtung.
- k_a : Ambienter Reflexionskoeffizient.
- I_a : Intensität des ambienten Lichts.
- N : Anzahl der Lichtquellen.
- k_d : Diffuser Reflexionskoeffizient.
- I_i : Intensität der i -ten Lichtquelle.
- n : Oberflächennormale.
- l_i : Richtung zur i -ten Lichtquelle.
- k_s : Spekularer Reflexionskoeffizient.
- h_i : Halbierungsvektor zwischen l_i und der Blickrichtung v .
- p : Glanz-Exponent (Polierheit).



- Es gibt noch viele weitere Aspekte, die man berücksichtigen muss, um der Realität näher zu kommen, aber diese werden hier nicht näher beschrieben: Farbverschiebungen in Abhängigkeit der Blickrichtung, Einfluss der Entfernung der Lichtquelle, anisotrope Oberflächen und Lichtquellen, Transparenz, atmosphärische Effekte, Schatten und so weiter.

Schattierung von Polygonen

Flat-Shading

[EVC_Skriptum_CG, p.37](#)

- Beim Schattieren eines Polygons hat klarerweise jeder Punkt die gleichen Oberflächeneigenschaften, vor allem auch den gleichen Normalvektor.
- Beim einfachen Ausfüllen jedes Polygons mit einer Farbe werden die Grenzen zwischen den Polygone deutlich störend erkennbar.
- Der sogenannte **Mach-Band-Effekt**, ein kantenverstärkender Mechanismus des Auges, macht das Problem dabei noch ärger als es ist.

- Dieser Effekt lässt uns Kanten die dunklere Seite dunkler wahrnehmen als sie ist, und die hellere Seite heller als sie ist.
- Die einfachste Lösung dieses Problems ist das Interpolieren der Schattierung zwischen den Polygonen. Dazu sind zwei Verfahren üblich: **Gouraud-Schattierung** und **Phong-Schattierung**.



Gouraud-Schattierung

[EVC_Skriptum_CG, p.37](#)

Die Gouraud-Schattierung interpoliert die berechneten Helligkeitswerte über die Polygonflächen. Dazu werden an den Eckpunkten der Polygone Helligkeitswerte berechnet und von diesen aus durch lineare Interpolation jedes Polygon gefüllt.

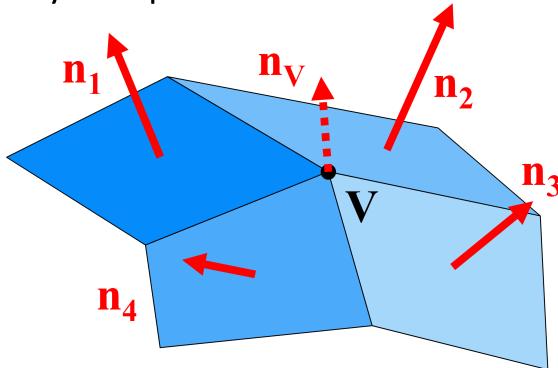


Konkret geht das so:

1. **Berechnung der Eckennormalen:** An jedem Eckpunkt wird eine Normale als Mittelwert der Normalen aller angrenzenden Polygone berechnet. Dies ist natürlich nur ein Näherungswert der Normale der echten zugrundeliegenden Fläche.
2. **Berechnung der Eckpunktintensitäten:** Aus den Eigenschaften der Oberfläche, der Normale (der gemittelten Eckennormalen) und der Lichteinfallsrichtung wird für jeden Eckpunkt ein Helligkeitswert („Schattierung“) berechnet. Beachte, dass dadurch angrenzende Polygone an diesen Eckpunkten alle die gleichen Werte erhalten.

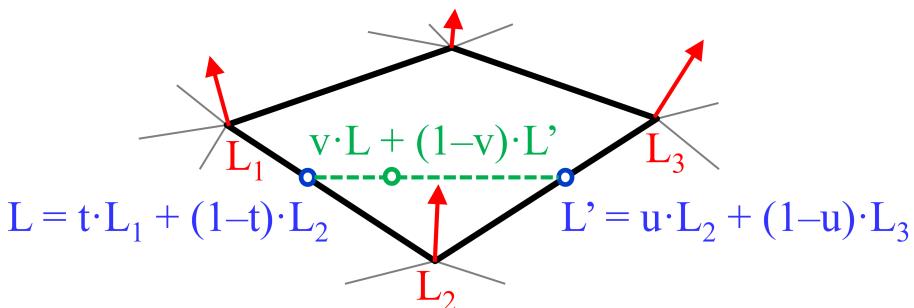
intensity-interpolation:

- determine average unit normal vector at each polygon vertex
- apply illumination model to each vertex
- linearly interpolate vertex intensities



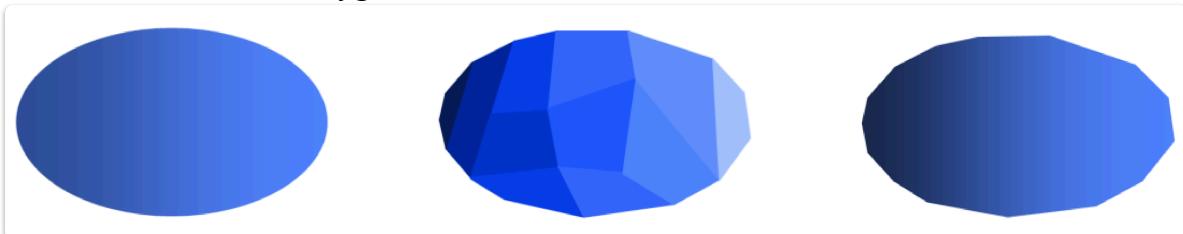
$$\mathbf{n}_v = \frac{\sum_{k=1}^N \mathbf{n}_k}{\left\| \sum_{k=1}^N \mathbf{n}_k \right\|}$$

- Interpolation entlang der Polygonkanten:** Entlang der Polygonkanten werden die Helligkeitswerte linear interpoliert, d.h. es wird für jeden Schnittpunkt mit einer Scanline ein Wert ermittelt. Beachte, dass dadurch für aneinandergrenzende Polygone entlang der gemeinsamen Kante die gleichen Werte entstehen.
- Interpolation entlang der Scanlines:** Entlang jeder Scanline wird von der linken bis zur rechten Polygongrenze wieder linear interpoliert. Dadurch haben nebeneinander liegende Pixel immer eine sehr ähnliche Helligkeit und es kommt zu keinen sichtbaren Kanten (im Idealfall).

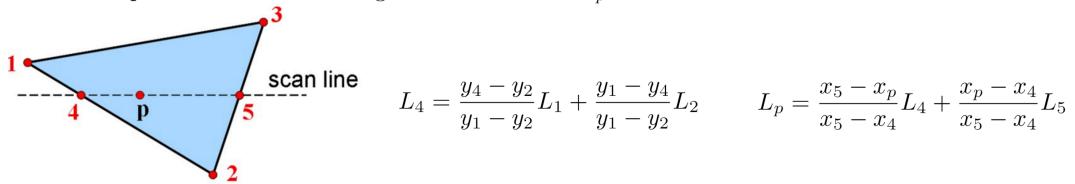


- find normal vectors at corners and calculate shading (intensities) there: L_i
- interpolate intensities along edges linearly: L, L'
- interpolate intensities along scanlines linearly: L_p

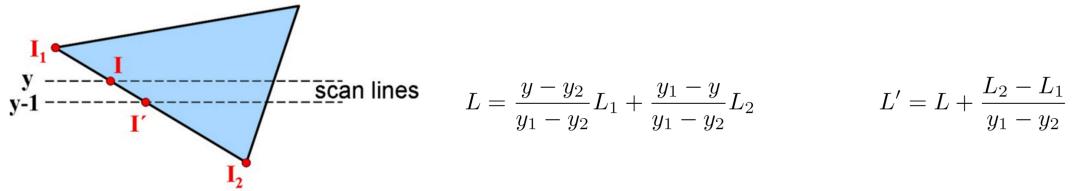
Dennoch verbleiben **Fehlerquellen**. So wird die Silhouette natürlich nicht verändert, dadurch verbleiben störende Polygonkanten sichtbar:



Einfache lineare Interpolation zur Berechnung eines Pixelwertes L_p :



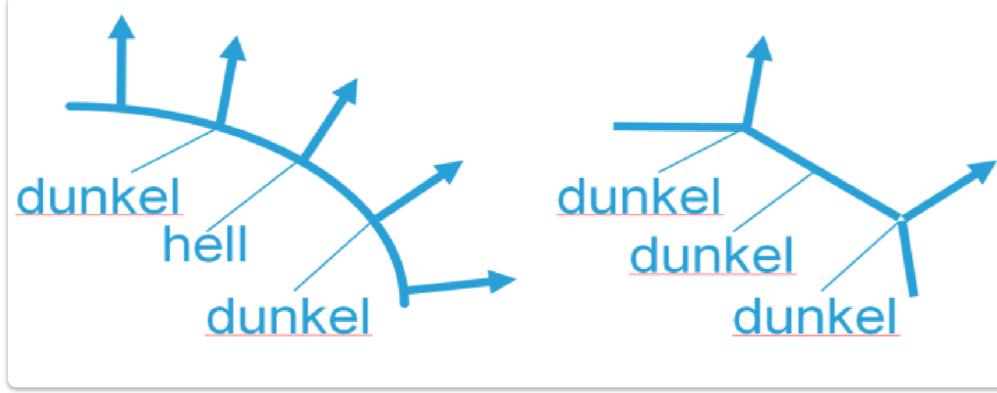
Die *lineare (!) Interpolation der Intensitäten* kann natürlich wieder inkrementell erfolgen, z.B.:



Probleme bei Gouraud-Schattierung (Glanzpunkte)

[EVC_Skriptum_CG, p.37](#)

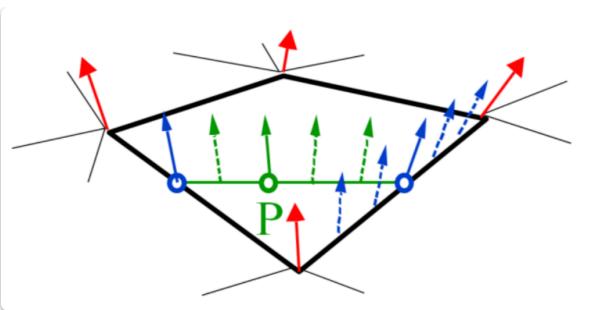
- Zufällige Interpolationsergebnisse bei Glanzpunkten möglich.
- Abhängig davon, ob Eckennormale zufällig Glanzpunkt erzeugt.
- Störend bei bewegten Objekten (Glanzpunkt "wandert").



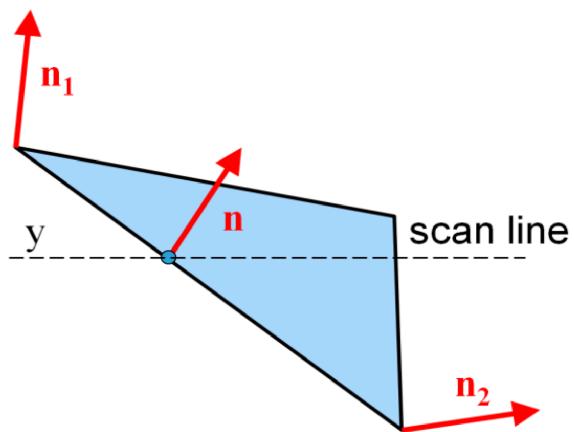
Phong-Schattierung

[EVC_Skriptum_CG, p.38](#)

- Alternative zu Gouraud-Schattierung für konsistenter Glanzpunkte.
- **Ablauf:**
 1. Normalen an Polygon-Eckpunkten berechnen.
 2. Diese Normalen entlang Polygonkanten interpolieren.
 3. Entlang Scanlines interpolierte Normalen weiter interpolieren (pro Pixel).
 4. Pro Pixel mit interpolierter Normalen Helligkeit nach Beleuchtungsmodell berechnen.
- **Unterschied:** Wir drehen Punkt 2 und 3 um. Also ich interpoliere die Vektoren und schattiere dann.
- **Vorteil:** Konsistenter Glanzpunkte.
- **Nachteil:** Höherer Aufwand (Beleuchtung pro Pixel).



Normalvektorinterpolation:



$$n = \frac{y - y_2}{y_1 - y_2} n_1 + \frac{y_1 - y}{y_1 - y_2} n_2$$

ZUR BEACHTUNG:

- Phong-Beleuchtungsmodell und Phong-Schattierung sind zwei unabhängige Konzepte!

Vergleich zwischen Gouraud und Phong:

comparison to Gouraud shading

- better highlights
- less Mach banding
- more costly
- wrong silhouette stays!

