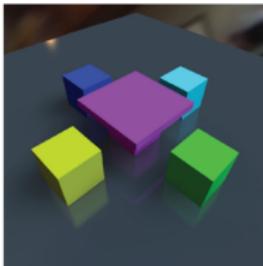
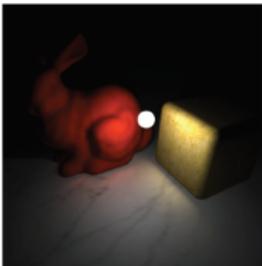
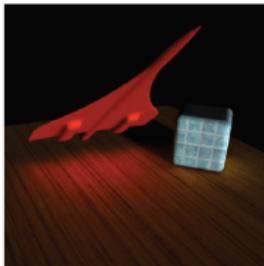
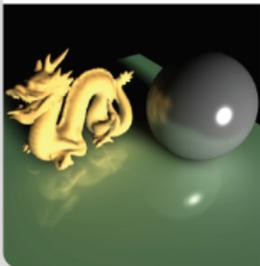


Glossy Reflections

Kai Westerkamp

Lehrstuhl für Computergrafik



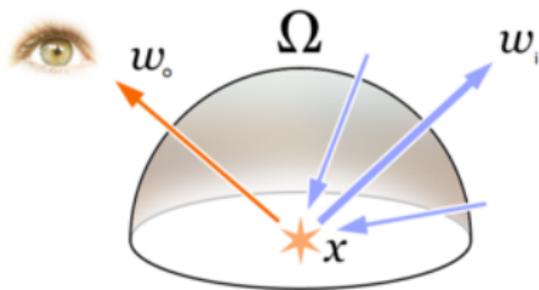
Einleitung

Einleitung// Inhalt?

Virtual Point Lights

$$L(x, o) = L_e(x, o) + \int_{\Omega^+} f_r(i, x, o) L_i(x, i) \max(0, i \cdot n) di$$

- x Oberflächenpunkt
- i Einfall Richtung (incoming)
- o Ausfall Richtung (outgoing)
- Ω^+ Positive Hemisphäre
- $f_r(i, x, o)$ BRDF



$$L_i(x, i) = L(y, -i)$$

$$y = \text{ray}(x, i)$$

$$G(\mathbf{v}; \rho, \lambda, c) = c \cdot e^{\lambda(\mathbf{v} \cdot \rho - 1)}$$

- ρ Mittelachse
- λ Sharpness
- c Skalar

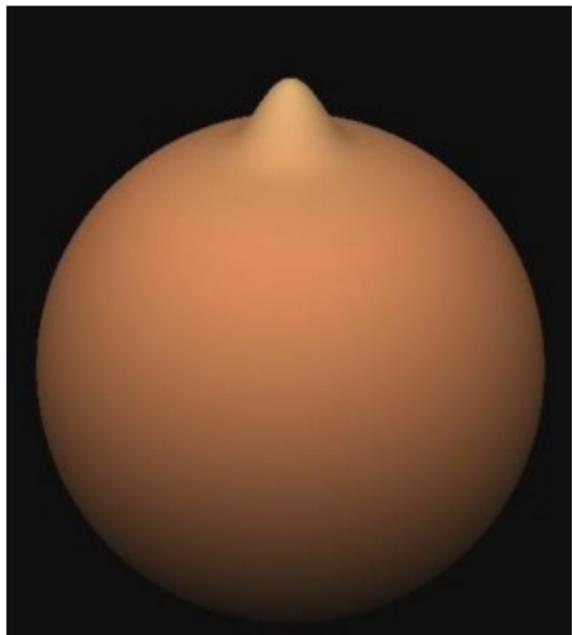
Schreibweisen:

$$G_l(\mathbf{v}) = G(\mathbf{v}; \rho_l, \lambda_l) = G(\mathbf{v}; \rho_l, \lambda_l, 1)$$

Eigenschaften:

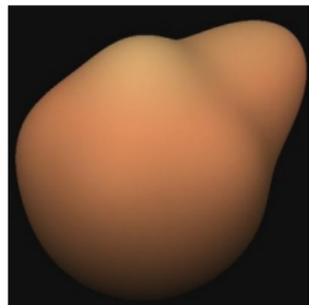
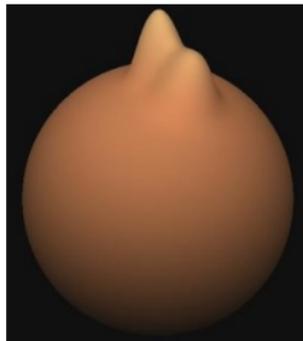
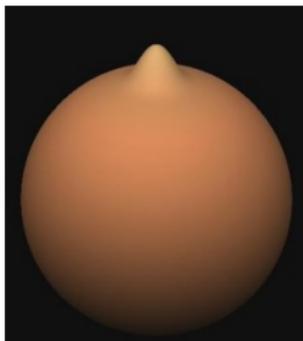
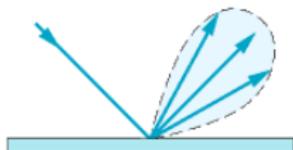
$$G_1(\mathbf{v}) \cdot G_2(\mathbf{v}) = G_3(\mathbf{v})$$

$$\int_{\Omega} G_1(\mathbf{v}) \cdot G_2(\mathbf{v}) d\mathbf{v} \approx c(\mathbf{v}) G_3(\mathbf{v})$$



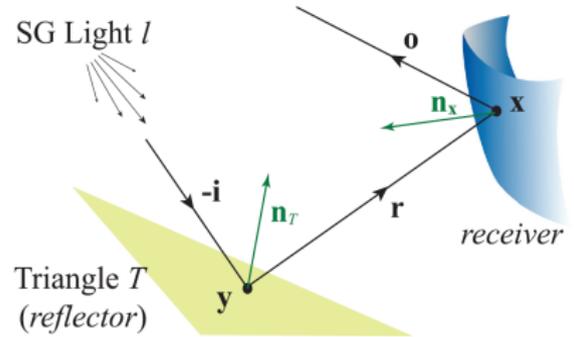
$$G(v; p, \lambda, c) = c \cdot e^{\lambda(v \cdot p - 1)}$$

- Summe an SG eignen sich zum Approximieren von BRDF
- Diffuse: SG mit $\lambda = 0$;
- Spekular: 1-9 SG
- Lichtquellen als SG

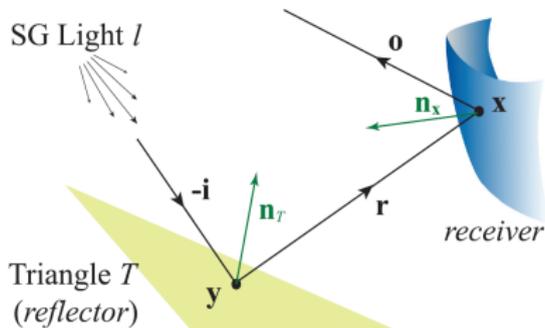


Algorithmus Übersicht

- One Bounce Interreflection
- Baumstruktur
- Sichtbarkeit



One Bounce Interreflection



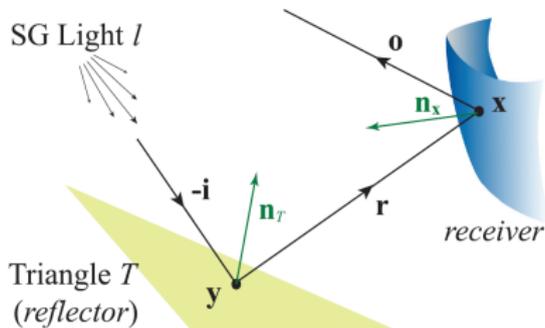
Receiver

$$L(x, o) = \int_{\Omega_T} f_r(-r, x, o) L_i(x, -r) \max(0, -r \cdot n_x) dr$$

Reflector

$$L(y, r) = \int_{\Omega} f_r(i, y, r) G_l(i) \max(0, i \cdot n_T) di$$

One Bounce Interreflection



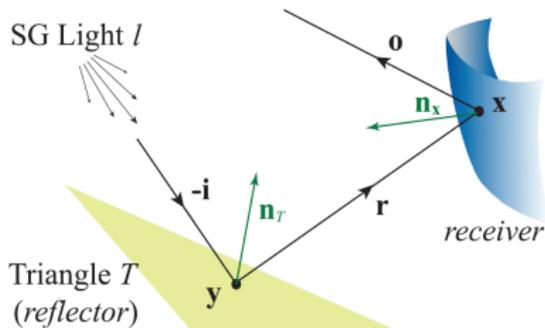
Reflector

$$L(y, r) = \int_{\Omega} f_r(i, y, r) G_l(i) \max(0, i \cdot n_T) di$$

BRDF als Summe von SG
Cos Term ist "wirklich glatt"

$$L(y, r) \approx F(r) G(r; i_r, \lambda_r)$$

One Bounce Interreflection

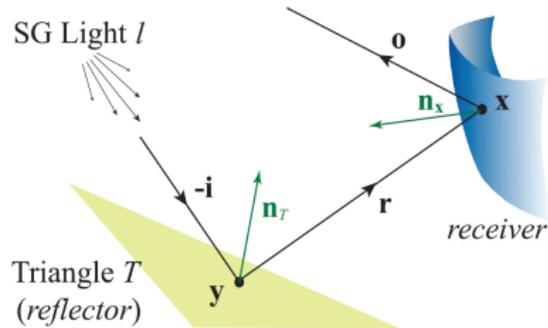


Receiver

$$L(x, o) = \int_{\Omega^T} f_r(-r, x, o) L_i(x, -r) \max(0, -r \cdot n_x) dr$$

BRDF als Summe von SG
Cos Term ist "wirklich glatt"
 L_i ist das reflektierte Licht

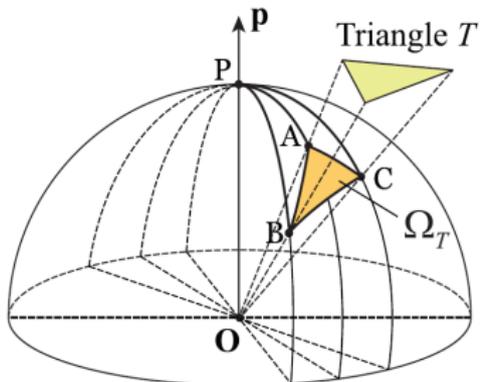
One Bounce Interreflection



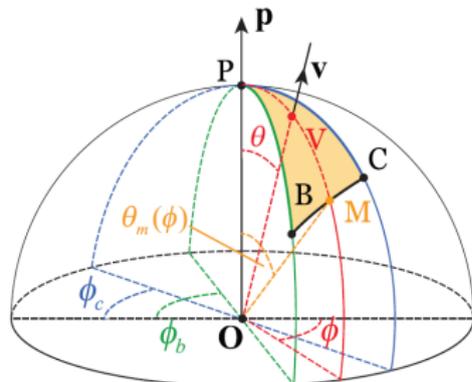
$$L(x, o) \approx H(r'_h) \int_{\Omega_T} G(r; r_h, \lambda_h, c_h) dr$$

One Bounce Interreflection

$$\int_{\Omega_T} G(\mathbf{v}; \mathbf{p}, \lambda) d\mathbf{v}$$



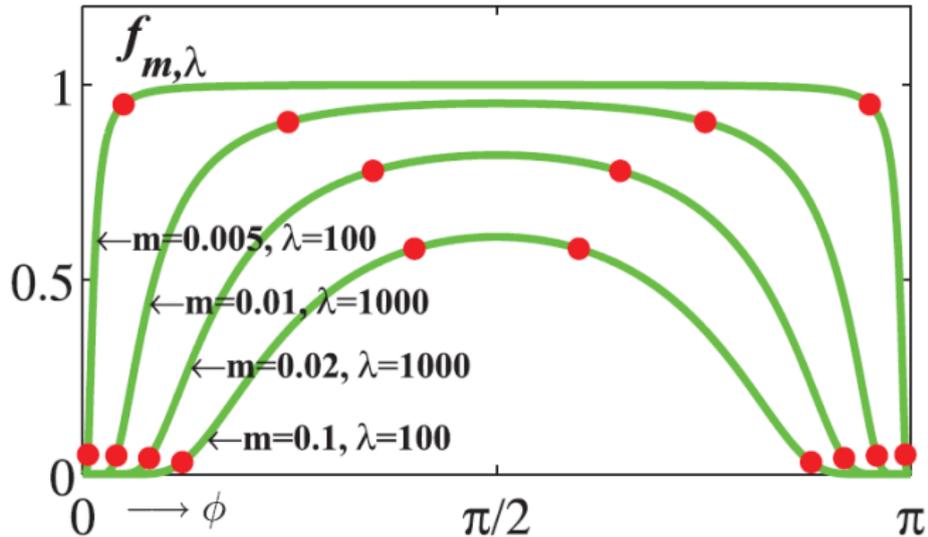
(a)



(b)

$$\Omega_t = \Omega_{\triangle ABC} = \Omega_{\triangle PBC} - \Omega_{\triangle PAB} - \Omega_{\triangle PCA}$$

One Bounce Interreflection



Struktur

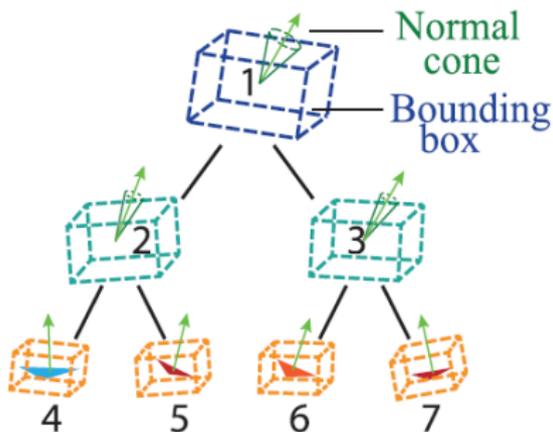
- Blätter sind die Dreiecke
- Knoten haben 2 Kinder

Berechnung der Reflexion

- Mittelpunkt
- Durchschnitts Normale
- die Durchschnittsfarbe
- Dreiecksfläche

Berechnung des Fehlers

- Bounding Box
- Normalenverteilung
- größte und kleinste Farbwert



Erstellen des Baums

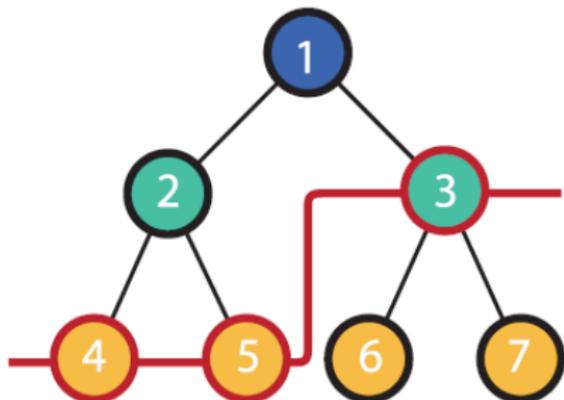
- Start: Wurzel mit allen Dreiecken
- Split Kriterium aus Dreiecks Mittelpunkt und Normale
- Principal component analysis (PCA)
- Split in 2 Kinder entlang der größte Varianz

Schnitt bestimmen

- Starte mit der Wurzel
- finde Knoten mit größte Fehler
- ersetze ihn durch beide Kinder

Abbruch bei:

- Fehler <1%
- 1000 Knoten

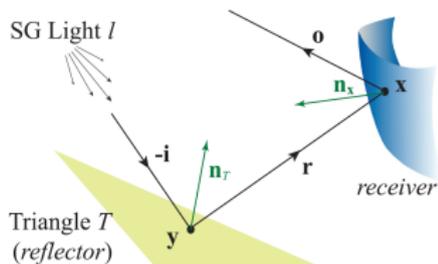


Licht und Reflektor

- Variance Shadow Maps

Reflektor Receiver

- 200 Virtual Point Lights mit Imperfect Shadowmaps
- Pro Knoten speichere die 3 nächsten VPL
- Projiziere Knoten auf die Ebene der VPL
- interpolieren (baryzentrische Koordinaten)

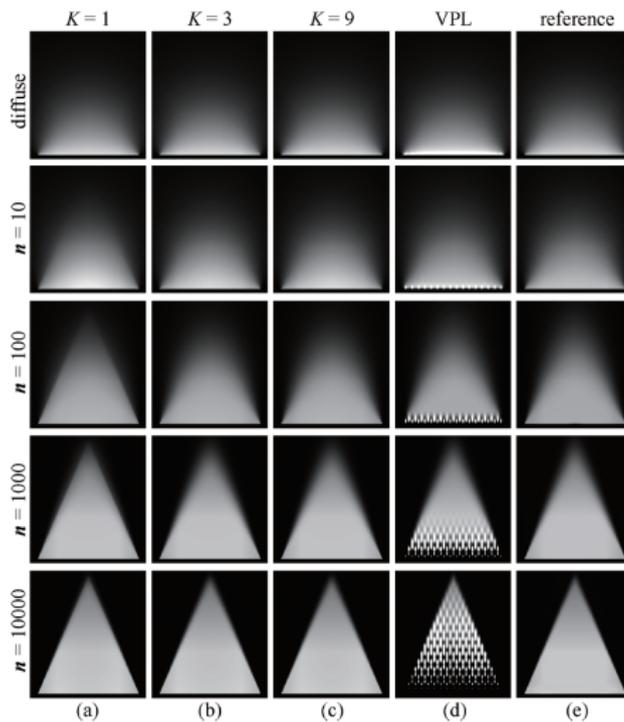


Setup

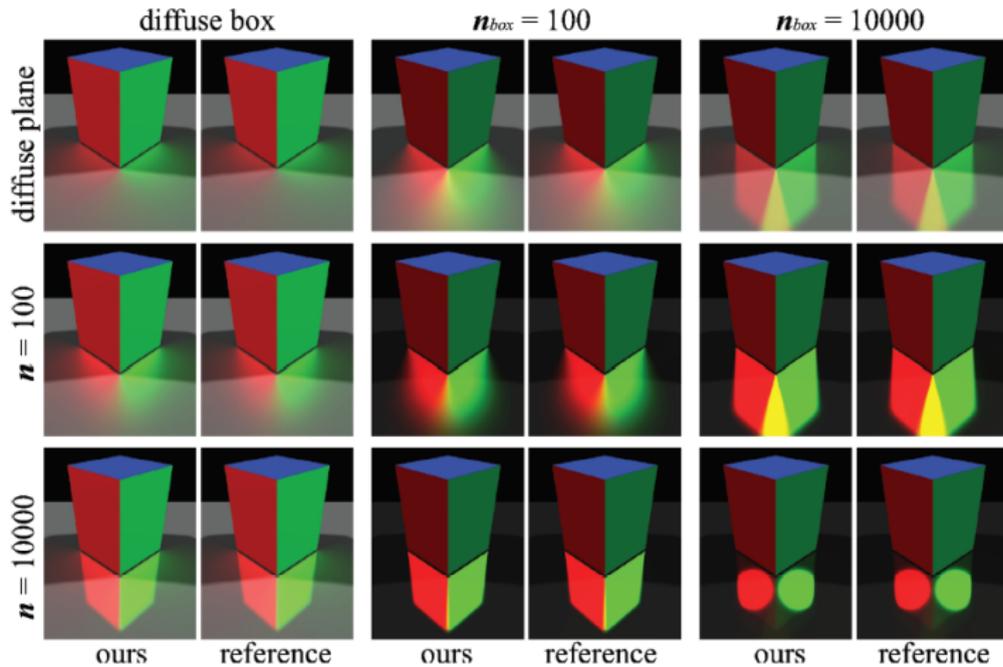
- Erstellen des Baums ab 200 Dreiecken

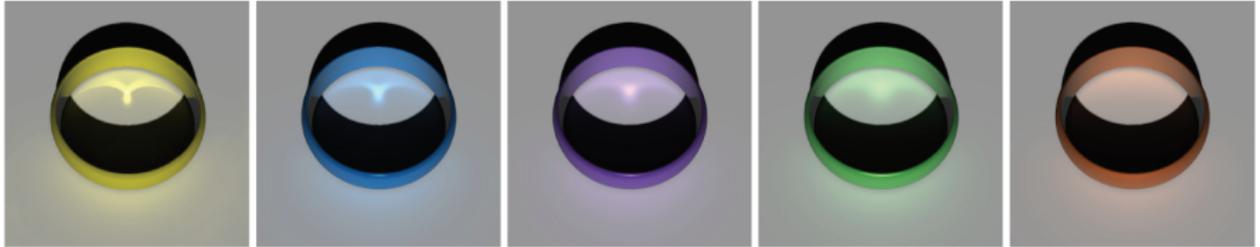
Pro Pixel

- Direkte Beleuchtung ausrechnen
- Schnitt Selektion
 - Für Vertices mit CUDA
 - "interpolieren" für Fragmente
- Berechnung der Reflektion

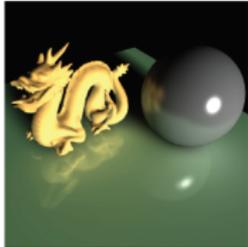


Evaluation

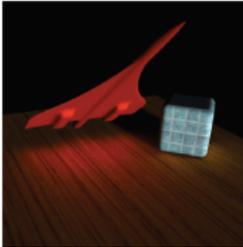




(a) ring



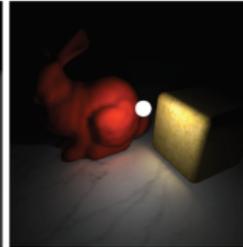
(b) dragon



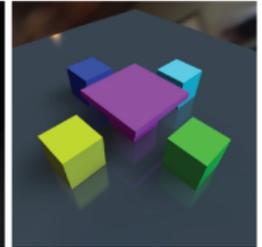
(c) airplane



(d) magic cube



(e) bunny



(f) table

Fragen?

Quellen

TODO