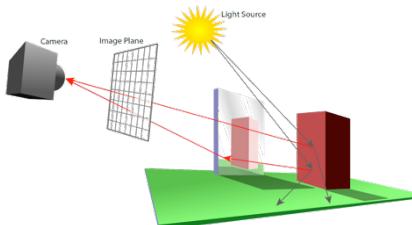
 BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Computergrafik II

Rekursives Raytracing

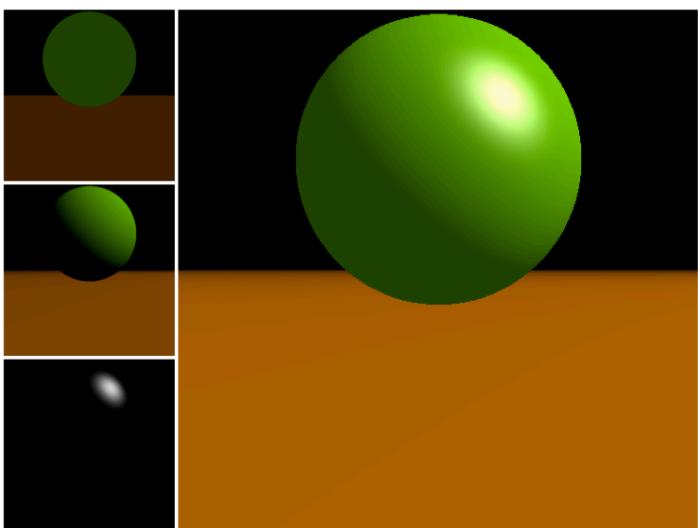


Bachelor Medieninformatik
Wintersemester 2011

Prof. Dr.-Ing. Hartmut Schirmacher
<http://schirmacher.beuth-hochschule.de>
hschirmacher@beuth-hochschule.de



Warum sieht das noch nicht so realistisch aus?



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



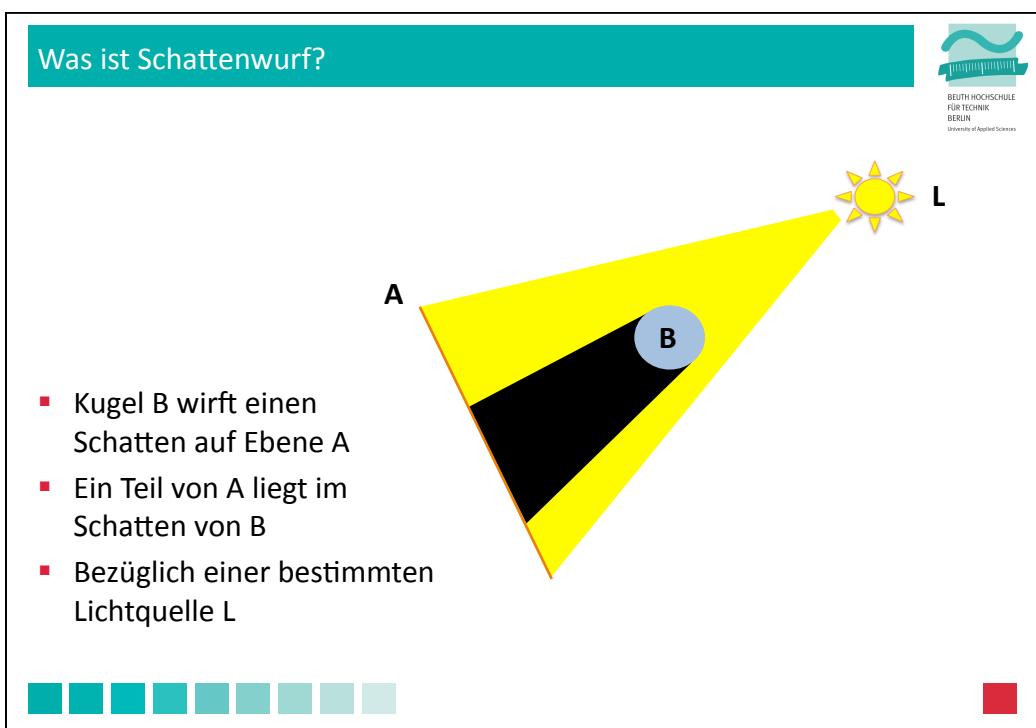
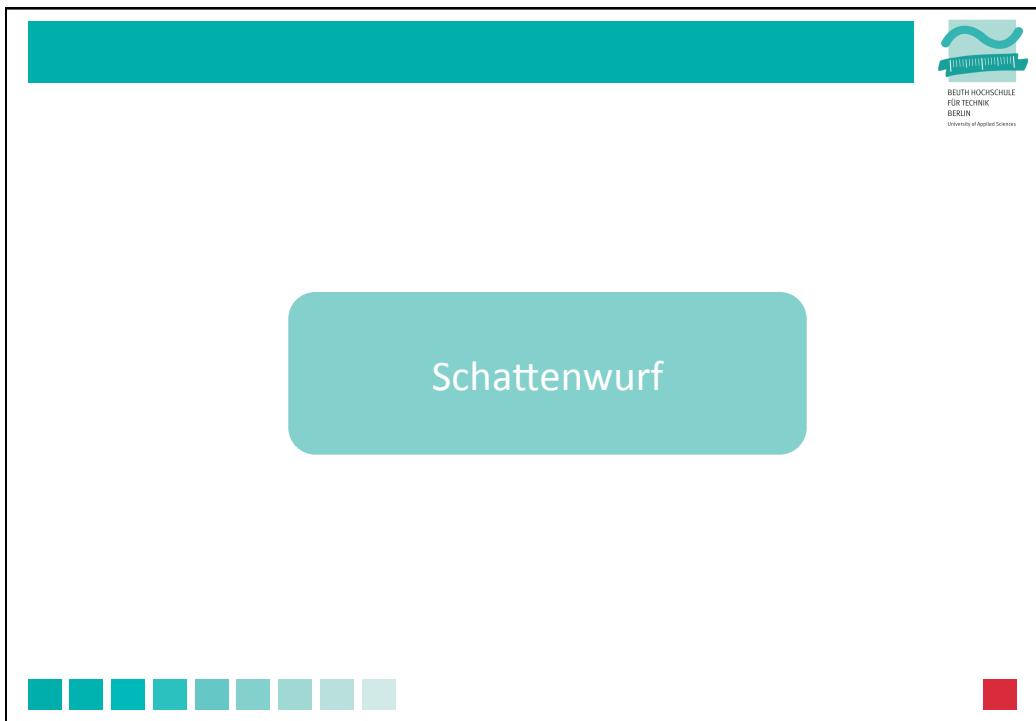
... Schatten und Interreflexionen fehlen!

BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

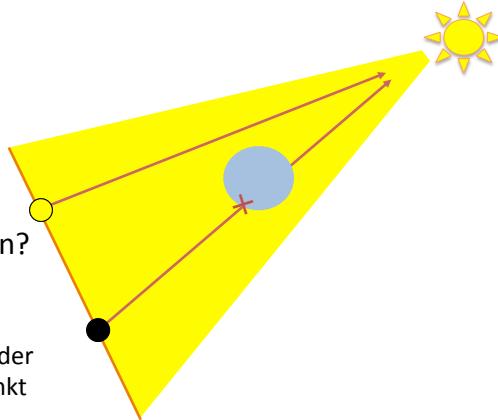
Gliederung

- Schattenberechnung
 - Schattenwurf von Punktlichtquellen und direktionalen Lichtquellen
- Reflexion und Refraktion
 - Berechnung perfekter Reflexion
 - Refraktion und Schlick-Approximation der Reflektivität
 - Weiche Schatten
- Implementierung des rekursiven Raytracers
 - Ablauf der Rekursion
 - Sonstige Implementierungshinweise
- Grenzen, Komplexität und Ausblick

BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



Was ist Schattenwurf?



- Liegt ein Punkt im Schatten?
 - Sende Strahl von Punkt zu Lichtquelle L
 - Wenn der Strahl etwas vor der Lichtquelle trifft, ist der Punkt im Schatten bezüglich L



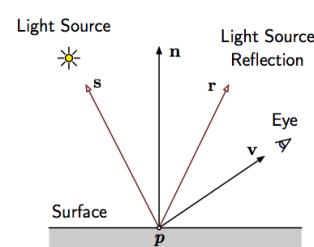
Schattenwurf im Phong-Modell



$$L(\mathbf{p}, \mathbf{v}) = k_a L^A + k_d \sum_j L_j (\mathbf{n} \cdot \mathbf{s}_j) + k_s \sum_j L_j (\mathbf{v} \cdot \mathbf{r}_j)^a$$

ambient diffus spekular

- Wie paßt die Abschattung ins Phong-Modell?
 - L_j : von Lichtquelle j eintreffendes Licht
 - Schattenwurf: die Sicht zwischen Punkt \mathbf{p} und Lichtquelle s_j ist blockiert, also ist $L_j = 0$
 - Um es noch expliziter in die Formel zu schreiben, ersetze L_j durch $S(\mathbf{p}, \mathbf{s}_j) L_j \dots$



Schattenwurf im Phong-Modell



ambient diffus spekular

$$L(\mathbf{p}, \mathbf{v}) = k_a L^A + k_d \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j (\mathbf{n} \cdot \mathbf{s}_j) + k_s \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j (\mathbf{v} \cdot \mathbf{r}_j)^a$$

- S : Sichtbarkeitsfunktion zwischen dem Oberflächenpunkt \mathbf{p} und der Lichtquelle in Richtung \mathbf{s}_j
- $S(\mathbf{p}, \mathbf{s}_j) = 1$ wenn die beiden Punkte gegenseitig sichtbar sind
- $S(\mathbf{p}, \mathbf{s}_j) = 0$ wenn die Sicht zwischen den beiden Punkten „blockiert“ ist
- $0 < S(\mathbf{p}, \mathbf{s}_j) < 1$ bei semi-transparenten Blockern (Volumen etc.)



Schattenwurf im Phong-Modell



ambient diffus spekular

$$L(\mathbf{p}, \mathbf{v}) = k_a L^A + k_d \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j (\mathbf{n} \cdot \mathbf{s}_j) + k_s \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j (\mathbf{v} \cdot \mathbf{r}_j)^a$$

$$= k_a L^A + \sum_j \underbrace{S(\mathbf{p}, \mathbf{s}_j)}_{\text{ambient}} \underbrace{L_j}_{\substack{\text{sichtbares} \\ \text{Licht von} \\ \text{Quelle } j}} \underbrace{(k_d (\mathbf{n} \cdot \mathbf{s}_j))}_{\text{diffuse Reflexion}} + \underbrace{k_s (\mathbf{v} \cdot \mathbf{r}_j)^a}_{\text{spekulare Reflexion}}$$

- S : Sichtbarkeitsfunktion zwischen dem Oberflächenpunkt \mathbf{p} und der Lichtquelle in Richtung \mathbf{s}_j
- $S(\mathbf{p}, \mathbf{s}_j) = 1$ wenn die beiden Punkte gegenseitig sichtbar sind
- $S(\mathbf{p}, \mathbf{s}_j) = 0$ wenn die Sicht zwischen den beiden Punkten „blockiert“ ist
- $0 < S(\mathbf{p}, \mathbf{s}_j) < 1$ bei semi-transparenten Blockern (Volumen etc.)



Implementierung von Schattenberechnungen



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

- Annahme:
 - Ausschließlich opake Objekte
 - Punktlichtquellen und direktionale Lichtquellen
- Dann genügt ein „Schattenstrahl“ pro Lichtquelle
 - Erzeuge Strahl von p zur Position der Lichtquelle
 - Schneide Strahl mit der Szene
 - Teste, ob Schnittpunkt zwischen Strahl und Lichtquelle liegt
 - Achtung: es könnten Objekte hinter dem Punkt p den Strahl schneiden
 - Achtung: es könnten Objekte hinter der Lichtquelle den Strahl schneiden
 - Wenn kein Objekt geschnitten wird: Sichtbarkeit = 1
 - Wenn nur opake Objekte geschnitten werden: Sichtbarkeit = 0

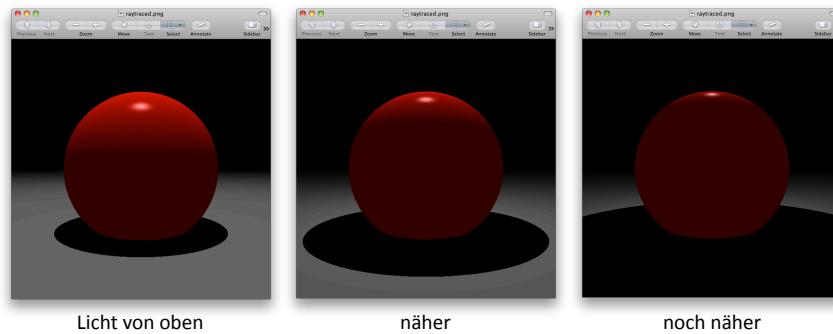
Dies erfolgt im Shader, also
z.B. in Material.Shade()



Phong mit Punktlichtquelle und Schatten



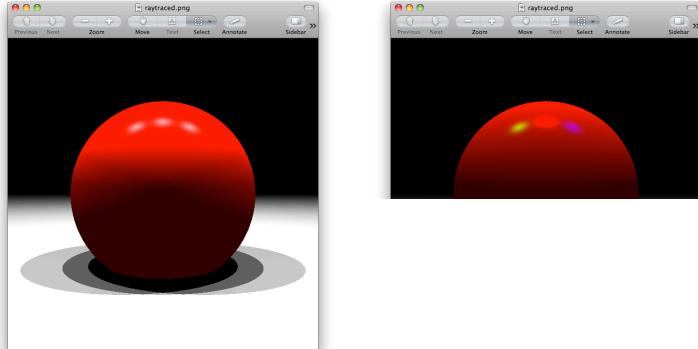
BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



- Kugel wirft Schatten auf die Ebene
- Je näher das Licht, desto größer der Schattenbereich
- Schatten ist scharf begrenzt
 - Wirkt etwas unnatürlich...



Beispiele mit mehreren Lichtquellen



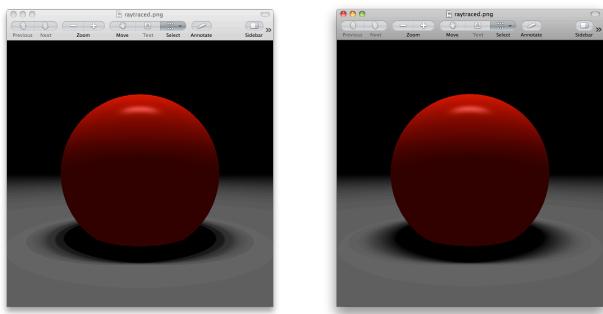
The image shows two side-by-side renderings of a red sphere on a reflective surface against a black background. The left rendering, labeled 'Drei weiße Punktlichter' (Three white point lights), shows a sharp shadow and highlights. The right rendering, labeled 'Punktlichter rot+grün+blau' (Point lights red+green+blue), shows a more complex shadow pattern with color bleeding where multiple lights overlap.

Drei weiße Punktlichter Punktlichter rot+grün+blau

- Mit mehreren Lichtern entstehen verschiedene starke Schattenbereiche
 - Keine / eine / zwei / drei Lichtquellen verdeckt
- Interessanter Fall: drei Lichtquellen rot, grün, blau
 - Dort, wo nur Licht von zwei der drei Quellen auftrifft, entstehen Mischfarben



Flächenlichtquellen und weiche Schatten (Soft Shadows)



The image shows two side-by-side renderings of a red sphere on a reflective surface against a black background. The left rendering, labeled '4 x 4 Punktlichter' (4x4 point lights), shows a sharp shadow with a distinct boundary. The right rendering, labeled '10 x 10 Punktlichter' (10x10 point lights), shows a smoother, more blurred shadow, illustrating the effect of soft shadows.

4 x 4 Punktlichter 10 x 10 Punktlichter

- Eine Punktlichtquelle erzeugt scharfe Schlagschatten
- Für weiche Schatten muß die Lichtquelle eine gewisse Ausdehnung haben
- Einfache Approximation durch mehrere Punktlichtquellen
 - Auswirkung auf die Performanz des Algorithmus?





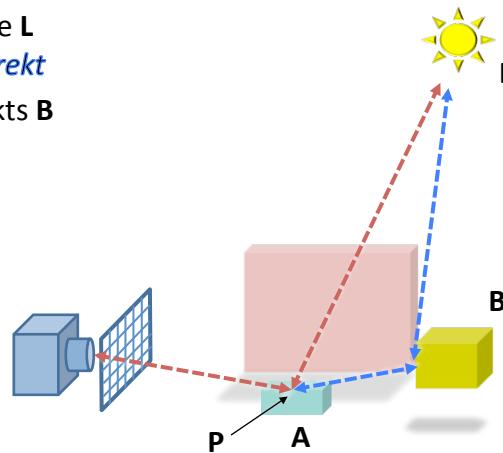
Reflexion und Refraktion



Was ist eine Interreflexion?



- Punkt **P** auf Objekt **A** wird nicht nur *direkt* von einer Lichtquelle **L** beleuchtet, sondern auch *indirekt*
- Die Farbe eines anderen Objekts **B** spiegelt sich in **P**

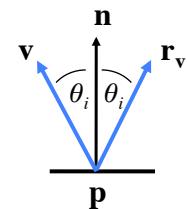


Interreflexionen im Phong-Modell



- Erweiterte Phong-Modell um perfekte Spiegelung

$$\begin{aligned}
 L(\mathbf{p}, \mathbf{v}) = & k_a L^A + \\
 & k_d \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j(\mathbf{n} \cdot \mathbf{s}_j) + \\
 & k_s \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j(\mathbf{v} \cdot \mathbf{r}_j)^a + \\
 & \textcolor{red}{k_r L(\mathbf{p}, -\mathbf{r}_v)}
 \end{aligned}$$



- k_r : Materialkoeffizient für spiegelnde Reflexionen (RGB-Vektor)
- $L(\mathbf{p}, -\mathbf{r}_v)$: eintreffendes Licht in Punkt p aus Richtung \mathbf{r}_v
- \mathbf{r}_v : Spiegelung der Richtung \mathbf{v} an der Oberflächennormalen in \mathbf{p}



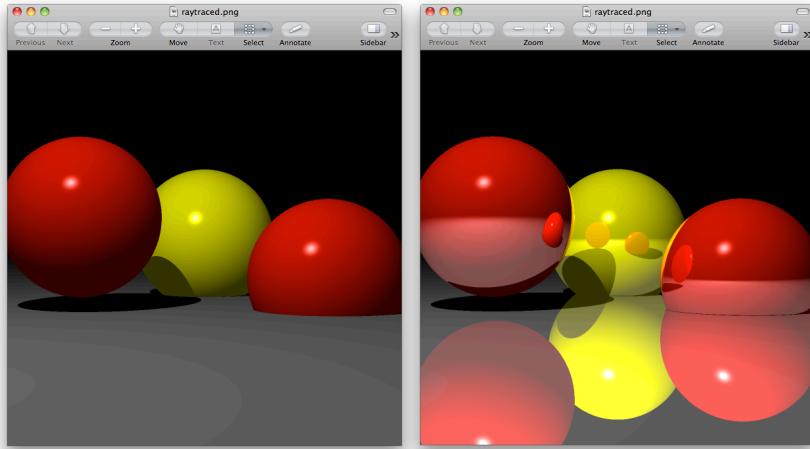
Spiegelnde Reflexionen



- Verständnisfrage:
 - Warum haben wir jetzt noch einmal spiegelnde Reflexionen eingeführt
 - Wir hatten doch schon den spekularen Term im Phong-Modell?
 - War der nicht schon für spiegelnde Reflexionen zuständig?



Beispiel mit Schatten und Interreflexion

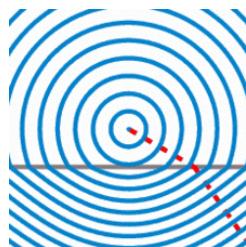


The image shows two side-by-side screenshots of a raytracer application window. Both windows have a title bar 'raytraced.png' and a toolbar with buttons for Previous, Next, Zoom, Move, Text, Select, Annotate, and Sidebar. The left screenshot shows three spheres (red, yellow, red) on a dark surface with simple shadows. The right screenshot shows the same setup but includes interreflections between the spheres, making the scene appear more complex and realistic.

BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



Was ist Refraktion / Brechung?



A diagram illustrating wave refraction. It shows concentric blue wave circles on the left moving from a medium with a higher speed to one with a lower speed. The wavelength decreases and the wave bends away from the normal line at the interface, represented by a dashed red line.

- In unterschiedlichen Medien reisen Lichtwellen mit verschiedenen Geschwindigkeiten
- Änderung der Richtung einer Welle durch Änderung der Geschwindigkeit
- Tritt beim Übergang zwischen zwei verschiedenen Medien auf (z.B. Wasser-Luft, Luft-Glas, ...)

Bild: Oleg Alexandrov
Quelle: wikipedia.de



Refraktionsindex



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

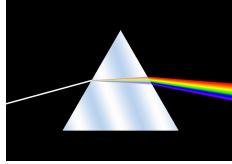
- Refraktionsindex η

- Beschreibt die Lichtgeschwindigkeit im Medium relativ zur Lichtgeschwindigkeit im Vakuum

$$\eta = \frac{c_M}{c_0}$$

- Eigentlich abhängig von der Wellenlänge, daher z.B. Dispersion am Prisma

Material	Refraktionsindex η
Vakuum	1.000
Luft	1.00029
Wasser	1.333
Glas	1.500
Diamant	2.419

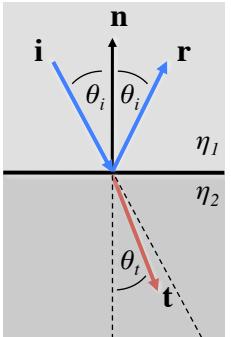


A diagram showing a triangular prism refracting a beam of white light into a spectrum of colors (rainbow) as it passes through.

Snellius'sches Gesetz (Snell's Law)



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



A diagram illustrating Snell's Law. A horizontal line represents a boundary between two media with refractive indices η_1 and η_2 . A blue ray labeled 'i' enters from the left at an angle θ_i relative to the normal 'n'. It refracts into the second medium at an angle θ_t relative to the normal. A red ray labeled 'r' represents perfect reflection within the first medium. A dashed red ray labeled 't' represents transmission into the second medium at the same angle θ_t .

- Snellius'sches Gesetz:

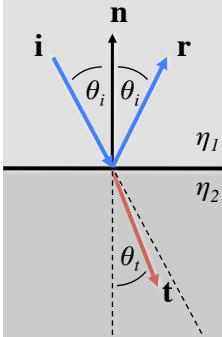
- Verhältnis von Einfallswinkel und Brechungswinkel

$$\frac{\eta_1}{\eta_2} = \frac{\sin \theta_t}{\sin \theta_i}$$


Willebrord van Roijen Snell
(a.k.a. Snellius)

- i : eintreffende (incident) Richtung
- n : Normale der Grenzfläche
- r : perfekte Reflexionsrichtung
- t : Transmissionsrichtung
- θ_i : Eintreffwinkel = Reflexionswinkel
- θ_t : Brechungs-/Transmissionswinkel
- η_1 : Refraktionsindex des Mediums vor der Grenzfläche
- η_2 : Refraktionsindex des Mediums hinter der Grenzfläche

Berechnung des Reflexionswinkels



- Reflexionsrichtung \mathbf{r}

$$\mathbf{r} = \mathbf{i} - 2(\mathbf{i} \cdot \mathbf{n})\mathbf{n}$$

- Refraktionsrichtung \mathbf{t}

$$\mathbf{t} = \frac{\eta_1}{\eta_2} \mathbf{i} + \left(\frac{\eta_1}{\eta_2} \cos \theta_i - \sqrt{1 - \sin^2 \theta_t} \right) \mathbf{n}$$

mit: $\sin^2 \theta_t = \left(\frac{\eta_1}{\eta_2} \right)^2 \sin^2 \theta_i \geq 0$

$$= \left(\frac{\eta_1}{\eta_2} \right)^2 (1 - \cos^2 \theta_i) \leq 1$$

Anschauliche Herleitung: Bram de Greve, *Reflections and Refractions in Ray Tracing*, 2004/2006, PDF Article, http://www.flipcode.com/archives/Reflections_and_Refraction_in_Raytracing.shtml

Totale interne Reflexion

- Die Gleichung hat nur dann eine reelle Lösung, wenn:

$$\left(\frac{\eta_1}{\eta_2} \right)^2 (1 - \cos^2 \theta_i) \leq 1$$

- Falls $\eta_1 > \eta_2$, kann diese Bedingung verletzt werden
 - Ab einem gewissen *kritischen Winkel* $\theta_i = \theta_C$ entsteht die sog. **totale interne Reflexion** – es findet keine Transmission statt
 - Daher zunächst die obige Bedingung testen – wenn verletzt, gibt es keine Transmission
 - Beispiel in der Natur:
 - Schaue senkrecht ins Wasser hinunter → gute Sicht unter die Oberfläche
 - Schaue von weitem unter einem geringen Winkel (at a grazing angle) auf eine Wasseroberfläche → nur Spiegelungen der Umgebung an der Oberfläche sichtbar

Beispiel totale interne Reflexion



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



Fresnel-Gleichungen

- Jetzt wissen wir, in welche Richtung gebrochenes Licht transmittiert wird
- Wieviel Licht wird reflektiert, und wieviel transmittiert?
 - Ein einzelnes Photon wird entweder reflektiert, oder transmittiert
 - Viele Photonen: Verteilung / welcher Anteil wird transmittiert?
- Fresnel-Gleichungen
 - Energieverteilung zwischen reflektiertem und transmittiertem Licht
 - Recht aufwändige Formeln, die von der Polarisation des Lichts abhängen aus maxwell'schen Gleichungen herleitbar



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences



Schlick-Approximation



- Schlick-Approximation¹ der Fresnel-Gleichungen
 - Speziell für die Verwendung in der Computergrafik
 - Einfach zu berechnen und ausreichend gut
 - R = Relektionsfähigkeit, T = Transmissionsfähigkeit

$$R = R_0 + (1 - R_0)(1 - \cos^2 \theta_i)^5,$$

$$T = 1 - R,$$

$$R_0 = \left(\frac{\eta_2 - 1}{\eta_2 + 1} \right)^2.$$

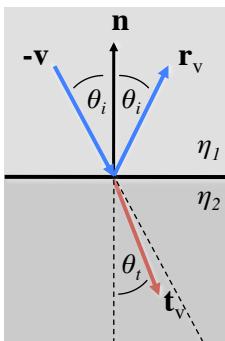
- Nochmal zur Erinnerung: $T = 0$ und $R = 1$ wenn folgende Bedingung verletzt ist:

$$\left(\frac{\eta_1}{\eta_2} \right)^2 (1 - \cos^2 \theta_i) \leq 1$$

1) Christophe Schlick, A Customizable Reflectance Model for Everyday Rendering, Fourth Eurographics Workshop on Rendering, 1993.



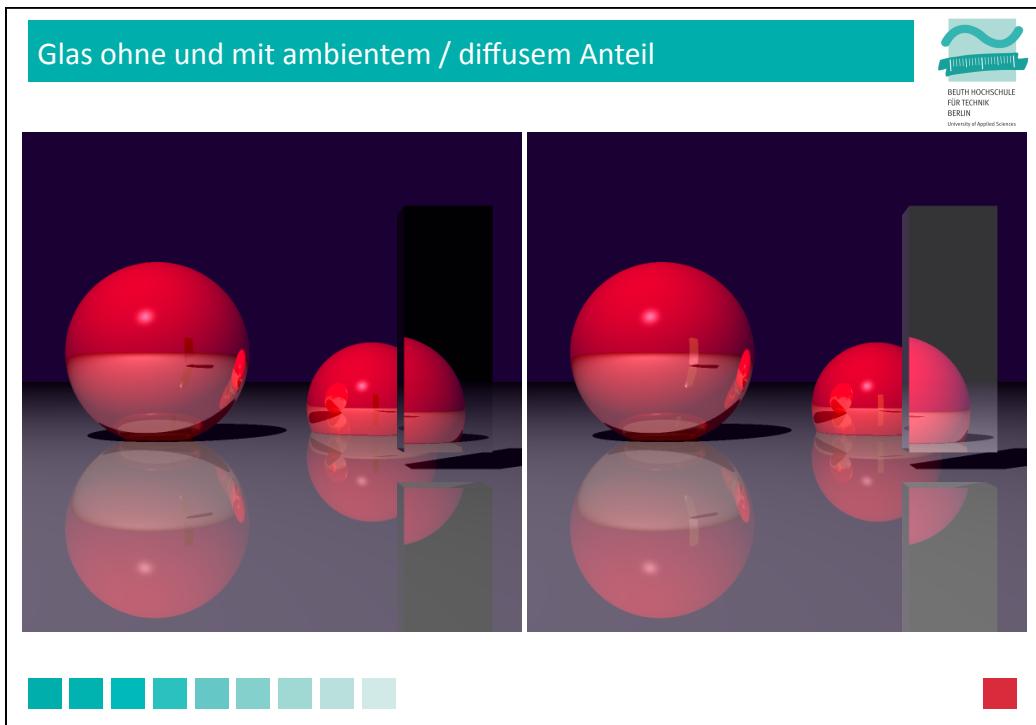
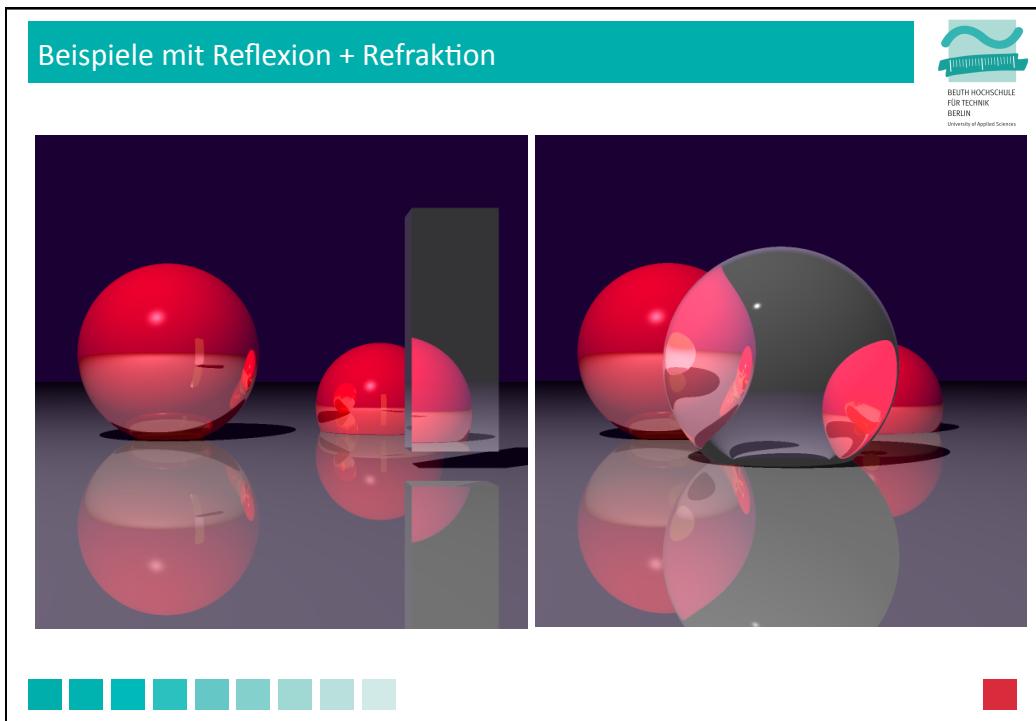
Phong-Modell mit Reflexion und Refraktion

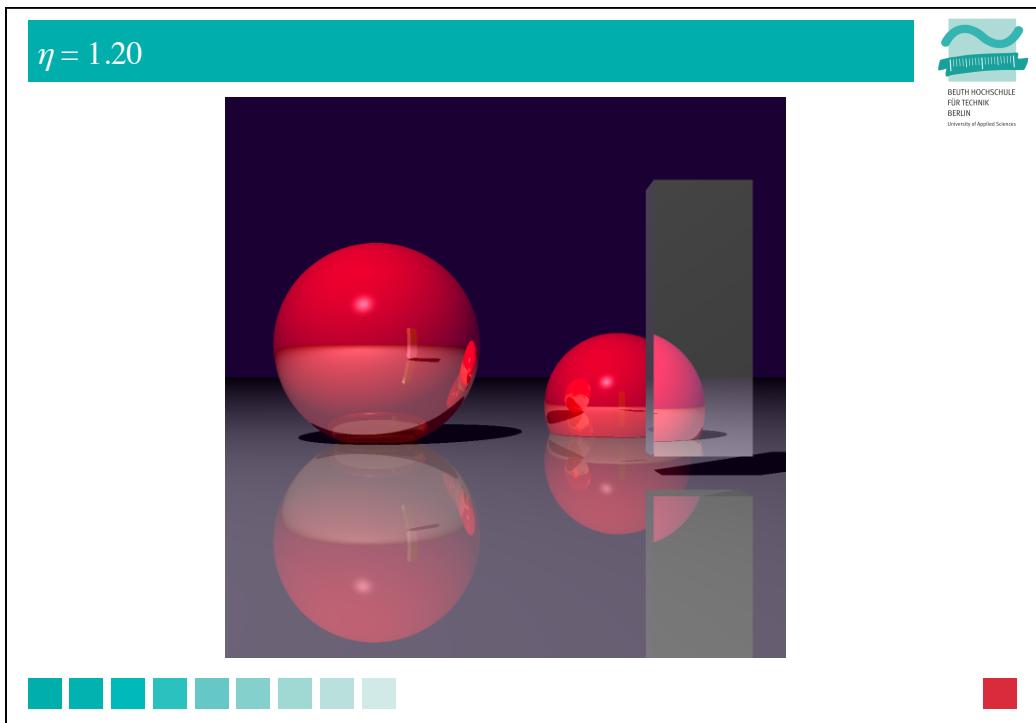
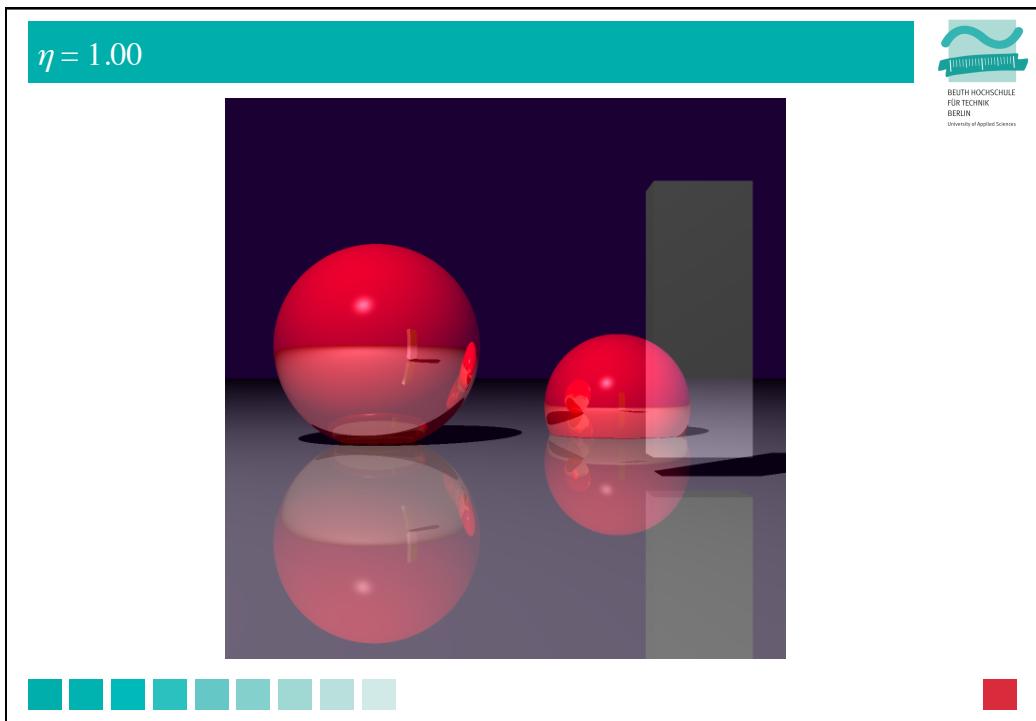


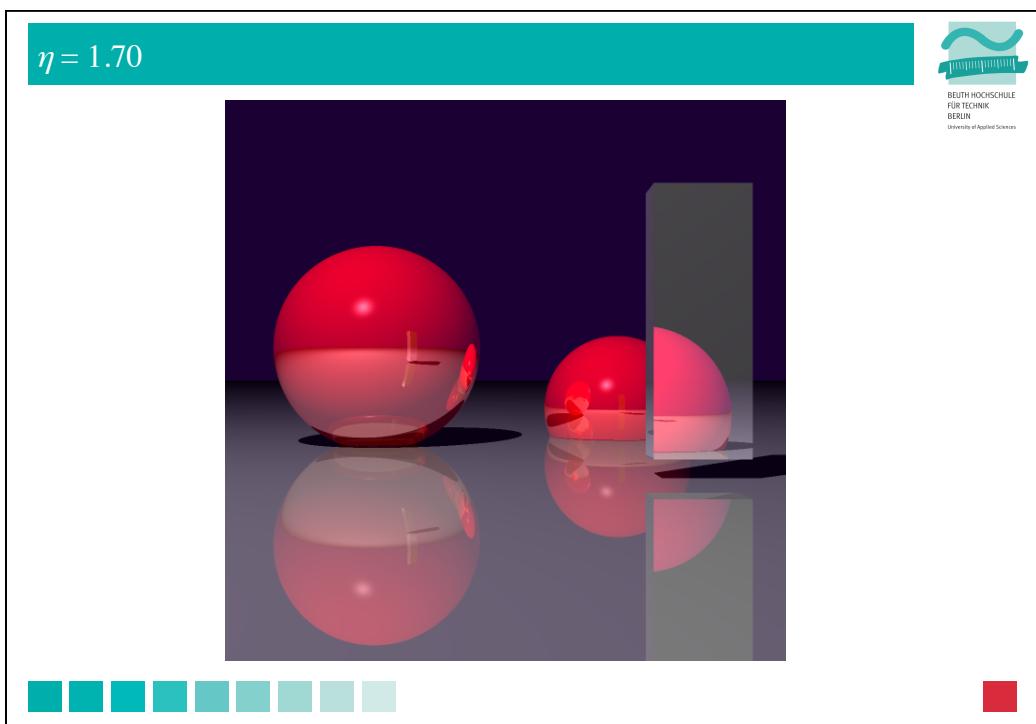
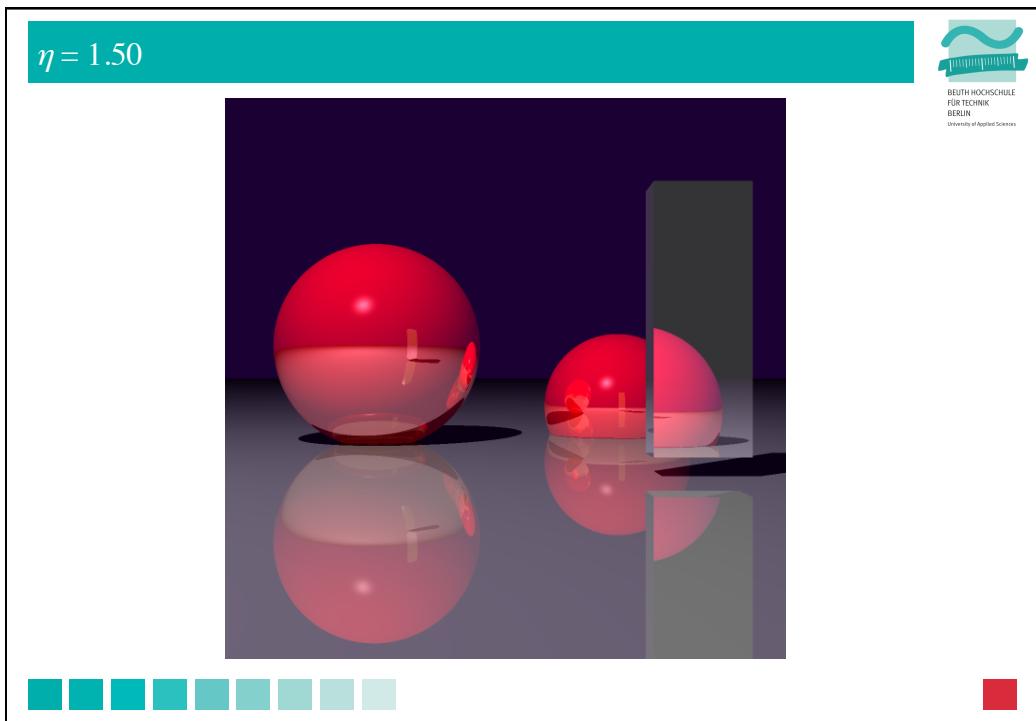
$$L(\mathbf{p}, \mathbf{v}) = k_a L^A + k_d \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j (\mathbf{n} \cdot \mathbf{s}_j) + k_s \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j (\mathbf{v} \cdot \mathbf{r}_j)^a + R k_r L(\mathbf{p}, -\mathbf{r}_v) + T k_r L(\mathbf{p}, -\mathbf{t}_v)$$

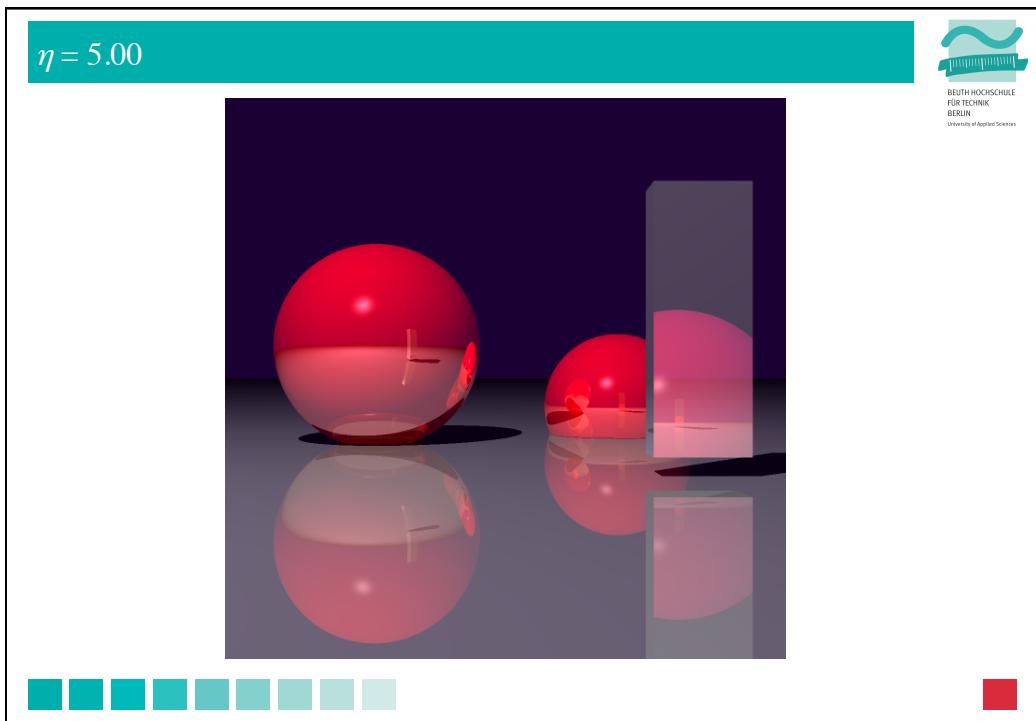
- \mathbf{v} : Betrachtungsrichtung (entspricht $-\mathbf{i}$)
- \mathbf{p} : Oberflächenpunkt
- \mathbf{n} : Normale in \mathbf{p}
- \mathbf{s}_j : Richtung zur Lichtquelle
- \mathbf{r}_j : Reflexionsrichtung der Lichtquelle
- \mathbf{r}_v : Reflexionsrichtung von \mathbf{v}
- \mathbf{t}_v : Transmissionsrichtung von \mathbf{v}
- $R = R(\mathbf{v}, \mathbf{n}, \eta_1, \eta_2)$: Reflektivität (Fresnel)
- $T = 1 - R$: Transmissivität (Fresnel)
- k_r : Reflexionskoeffizient (Materialkonstante)











Pseudocode des Raytracing Main Loops



```
// calculate color by tracing rays from the camera into the scene
raytrace(scene, camera, image)
{
    // loop over all pixels in image
    for all (i,j) from (0,0) to (image.width-1, image.height-1) do
    {
        // generate ray from eye point through pixel center
        ray ← generate_ray(i, j, image.width, image.height, camera);

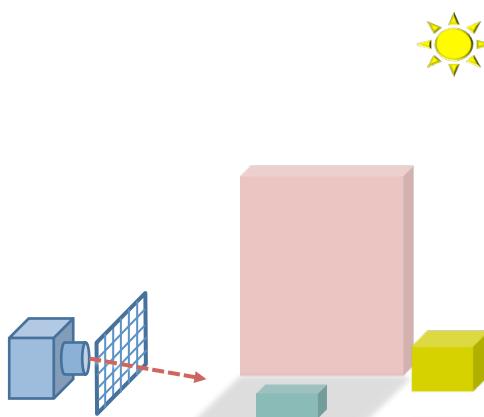
        // find first intersection point with scene objects
        hit ← intersect(ray, scene);

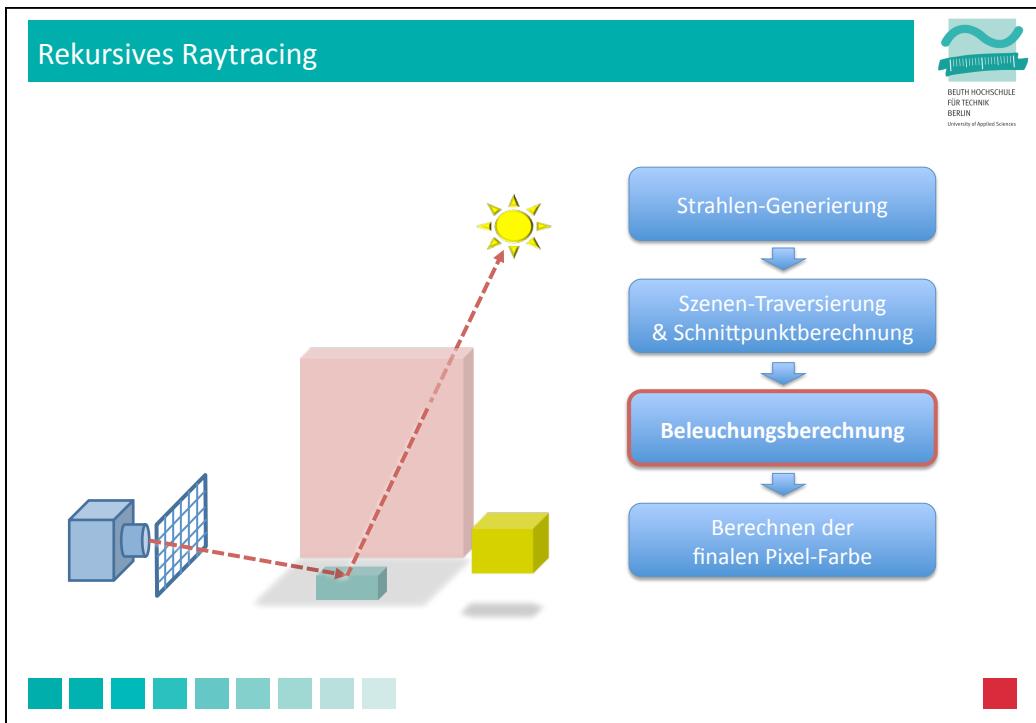
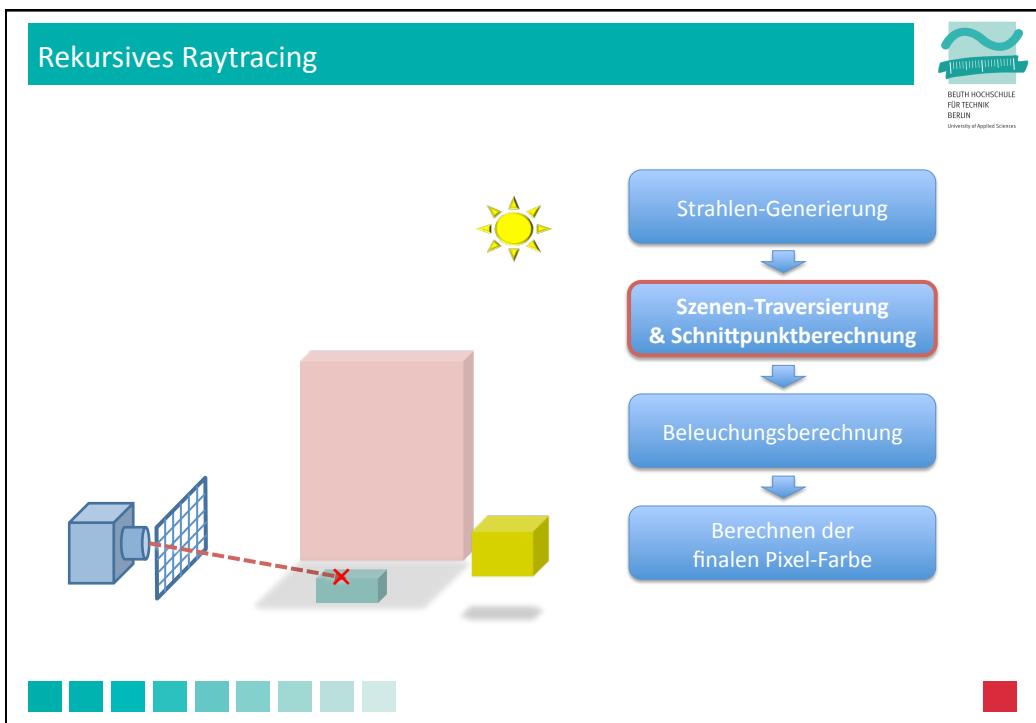
        // calculate light intensity/color
        color ← shade(hit, ray, scene); Potentiell rekursiv!!!

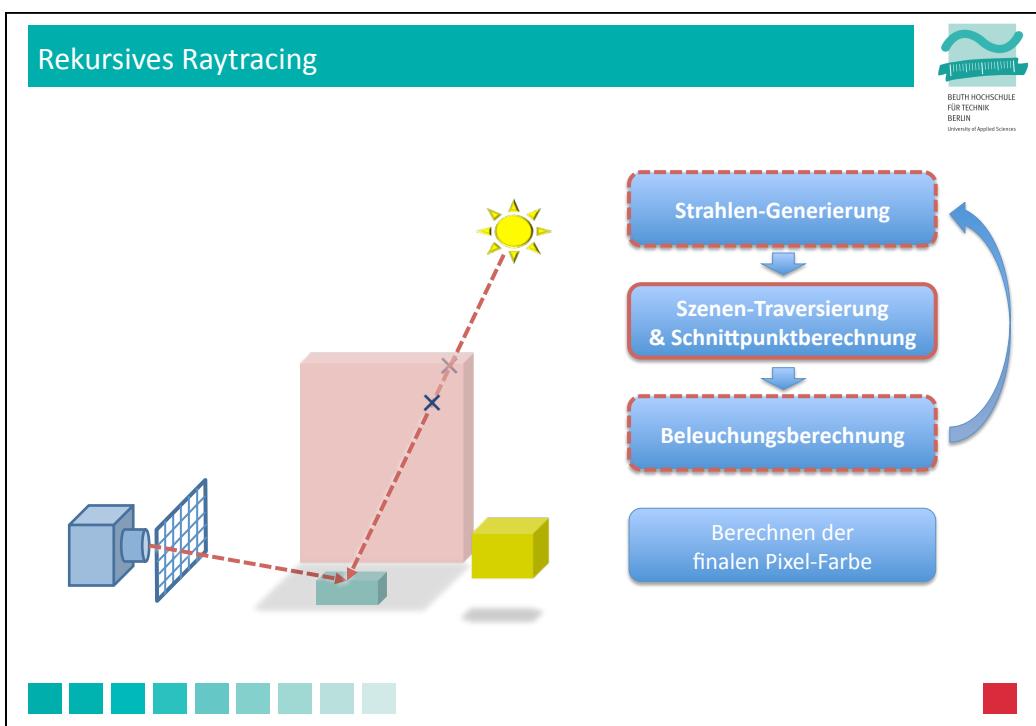
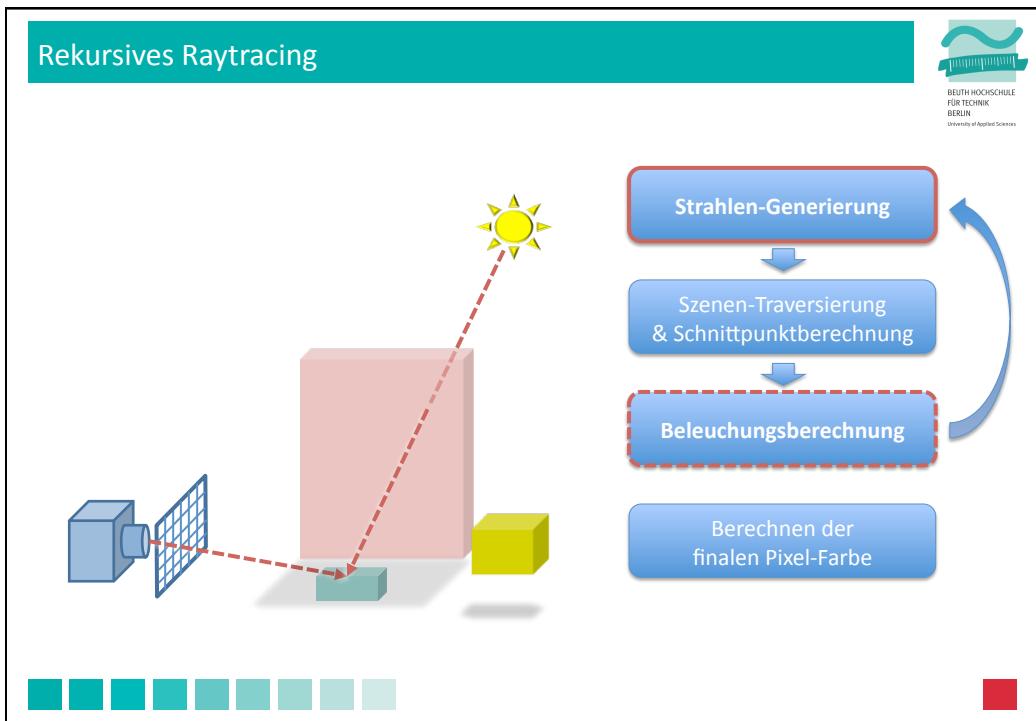
        // set corresponding pixel color in image
        image.pixel(i,j) ← color;
    }
}
```

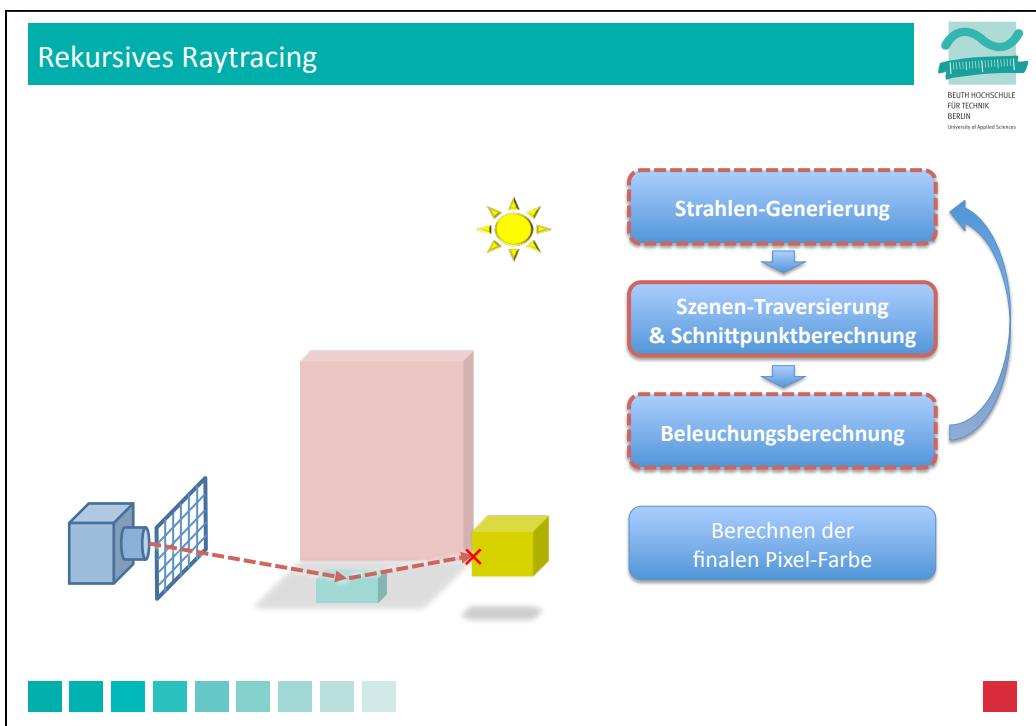
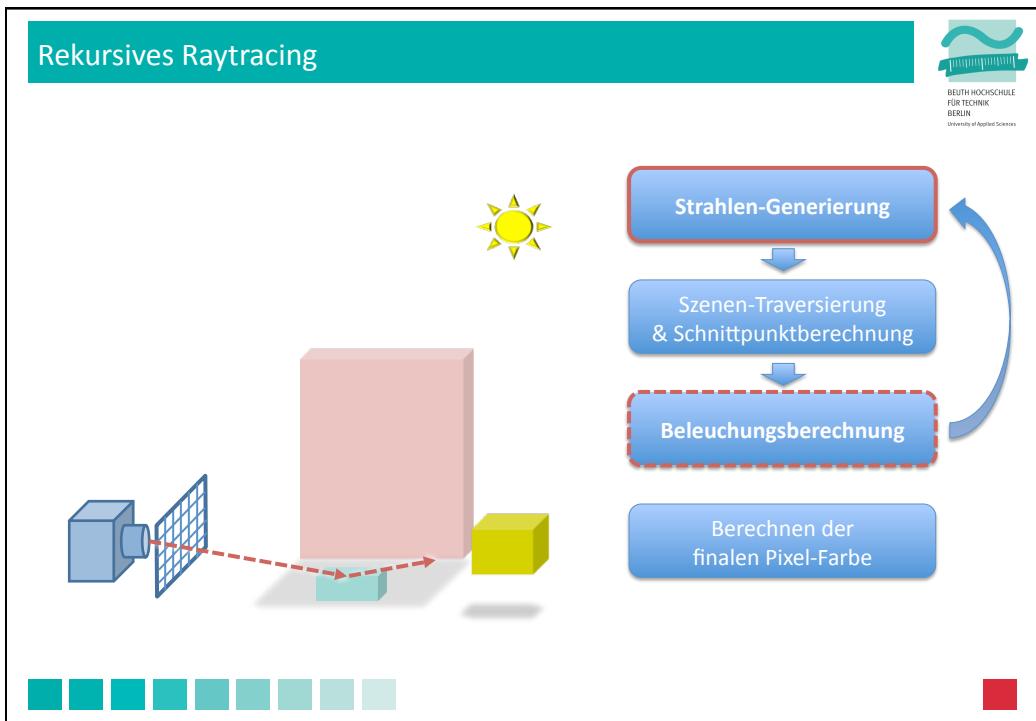


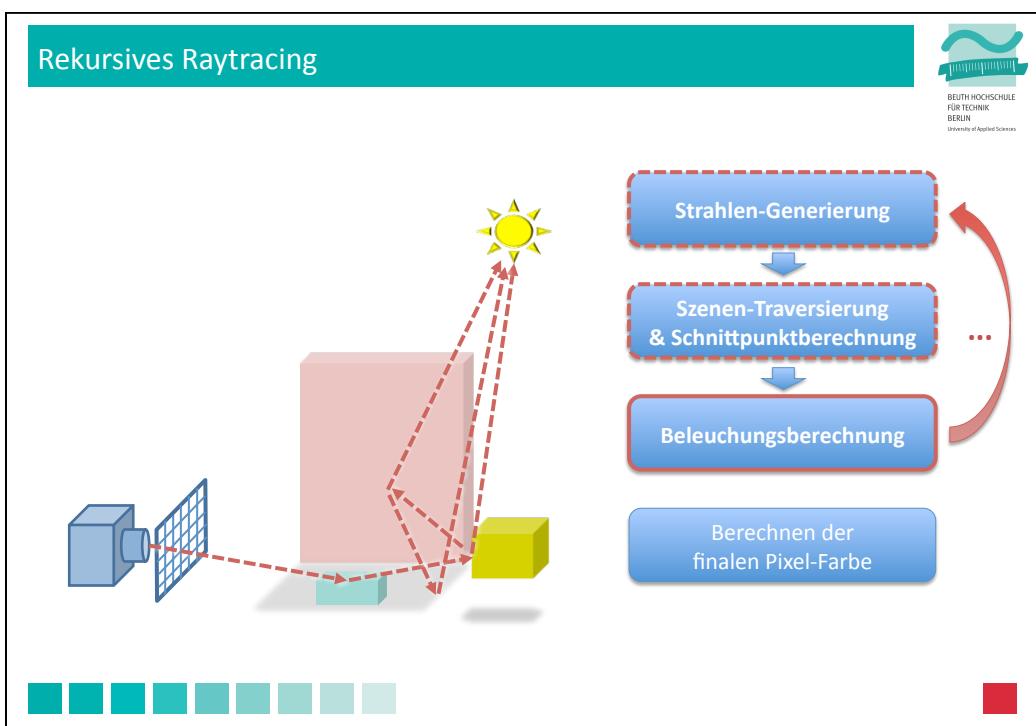
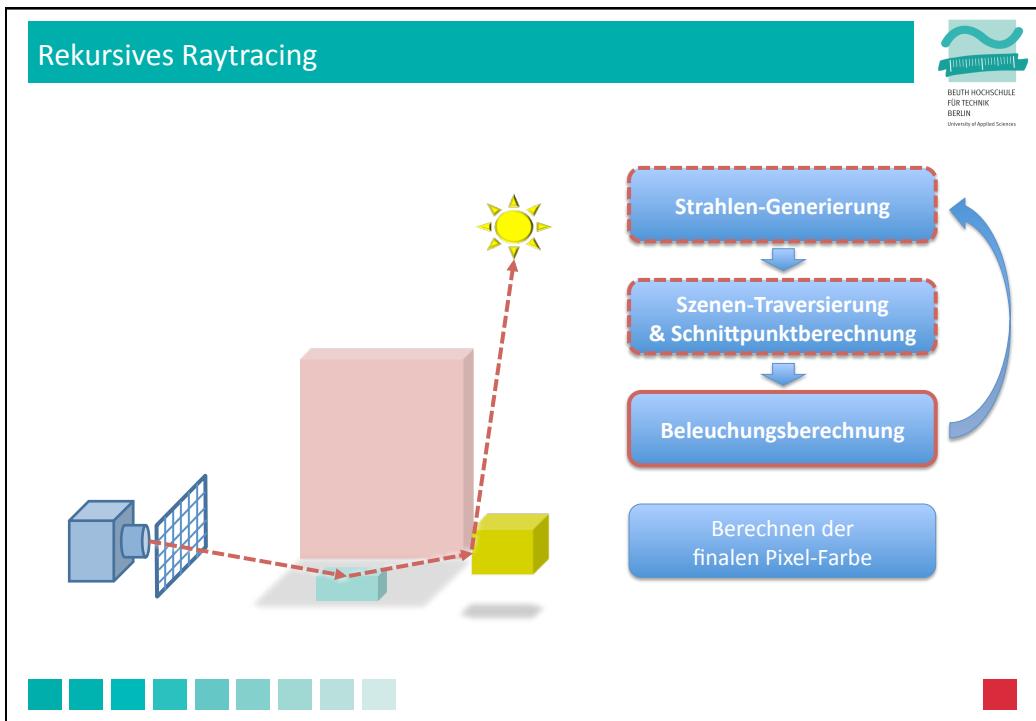
Rekursives Raytracing

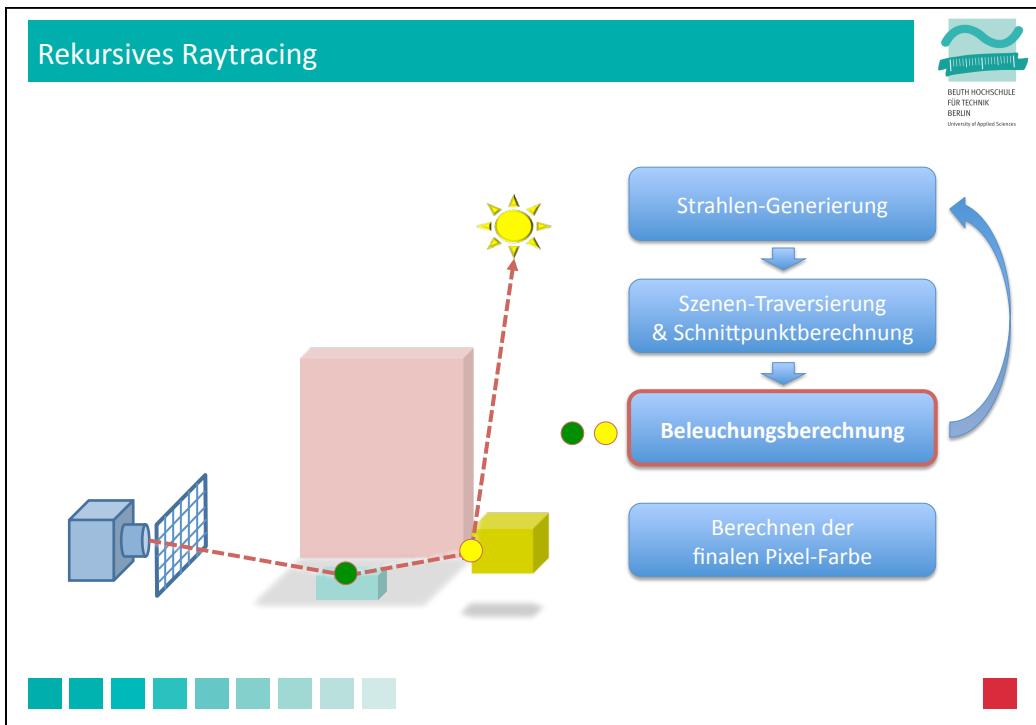
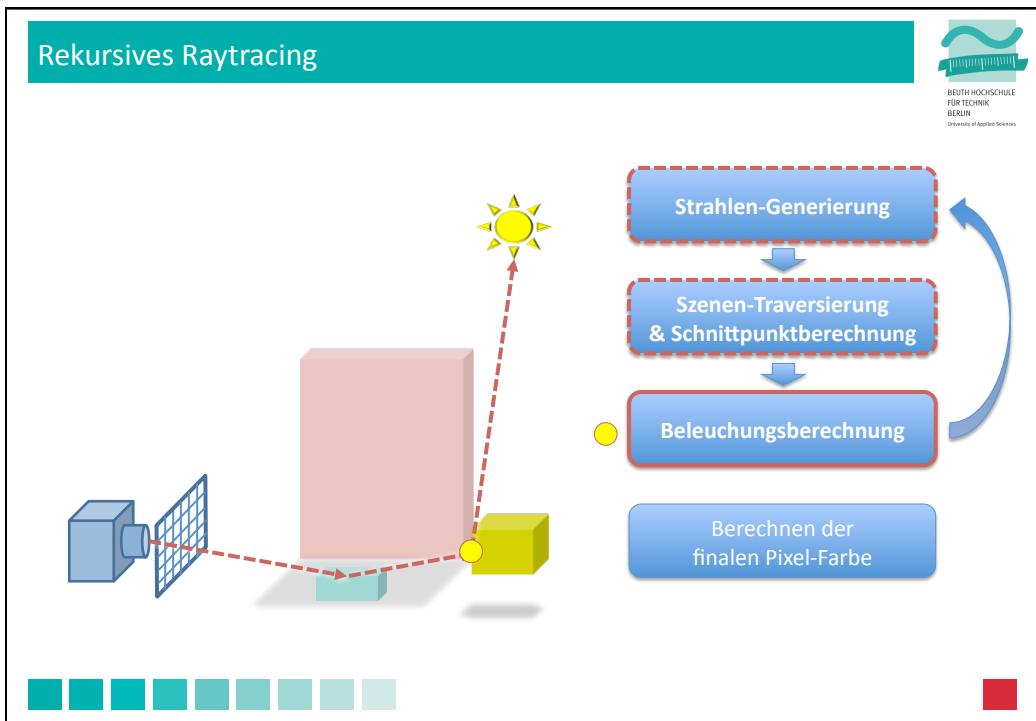


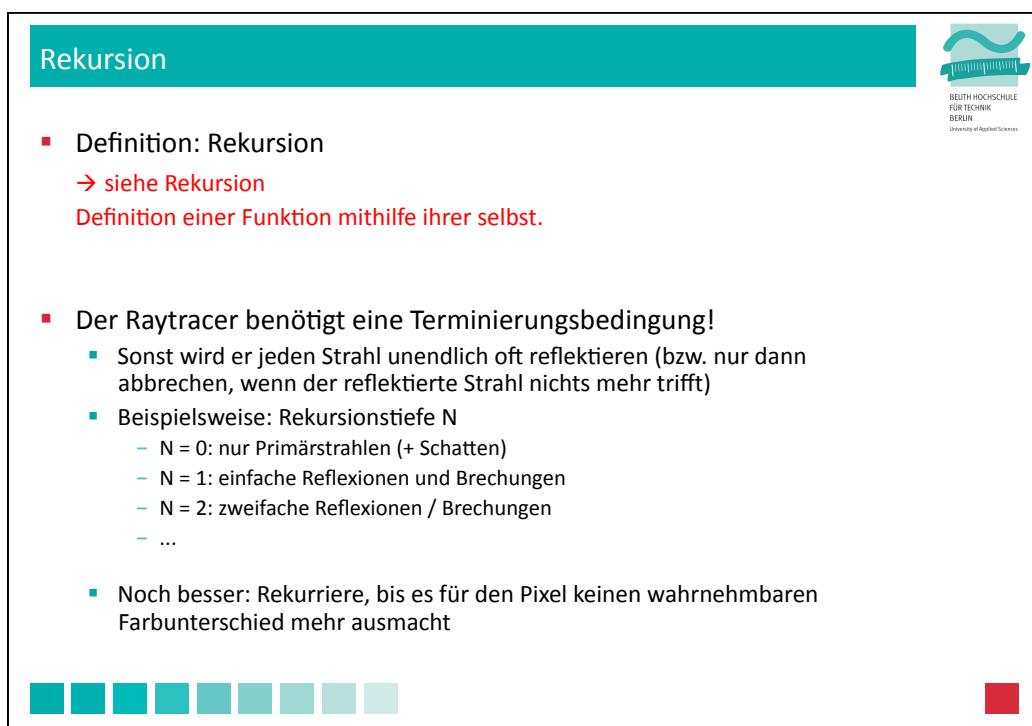
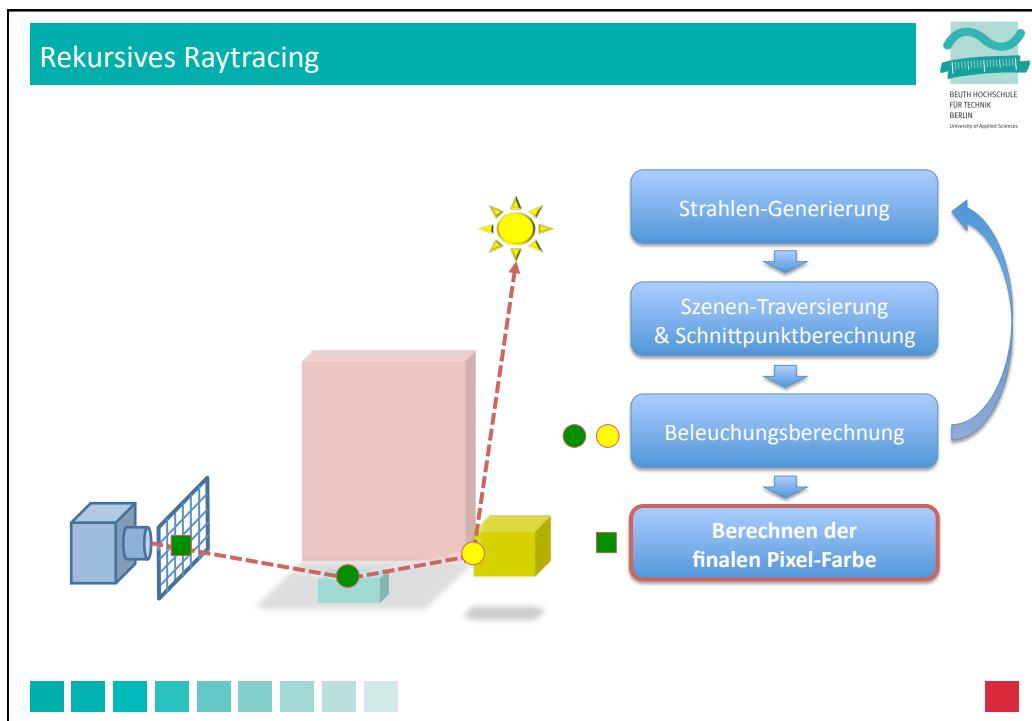












Rekursion in der Phong-Formel



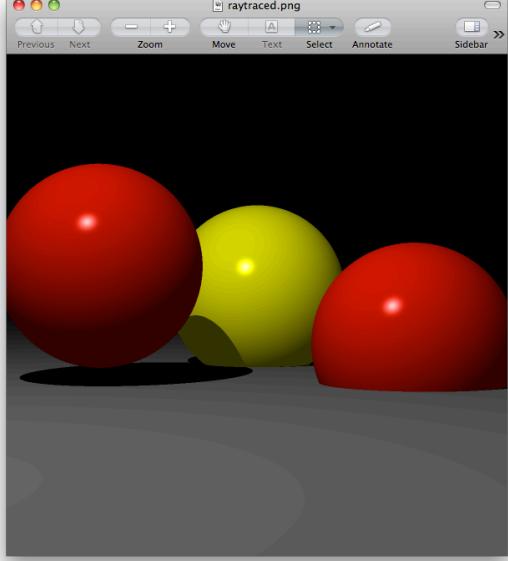
BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

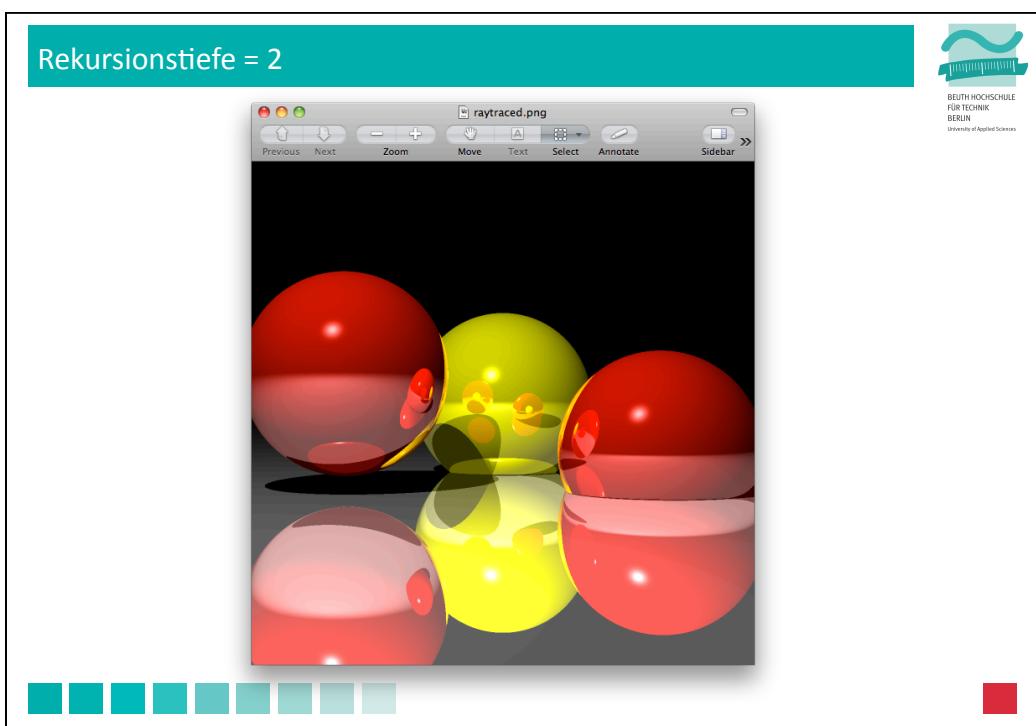
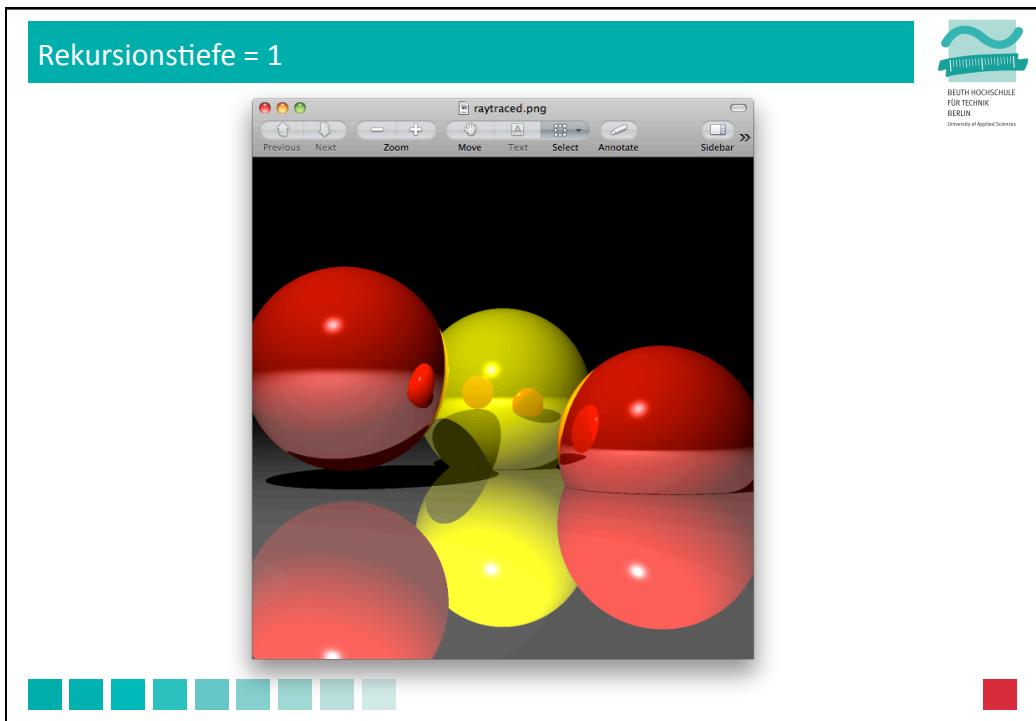
$$L(\mathbf{p}, \mathbf{v}) = k_a L^A + k_d \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j(\mathbf{n} \cdot \mathbf{s}_j) + k_s \sum_j S(\mathbf{p}, \mathbf{s}_j) L_j(\mathbf{v} \cdot \mathbf{r}_j)^a + R k_r L(\mathbf{p}, -\mathbf{r}_v) + T k_r L(\mathbf{p}, -\mathbf{t}_v)$$

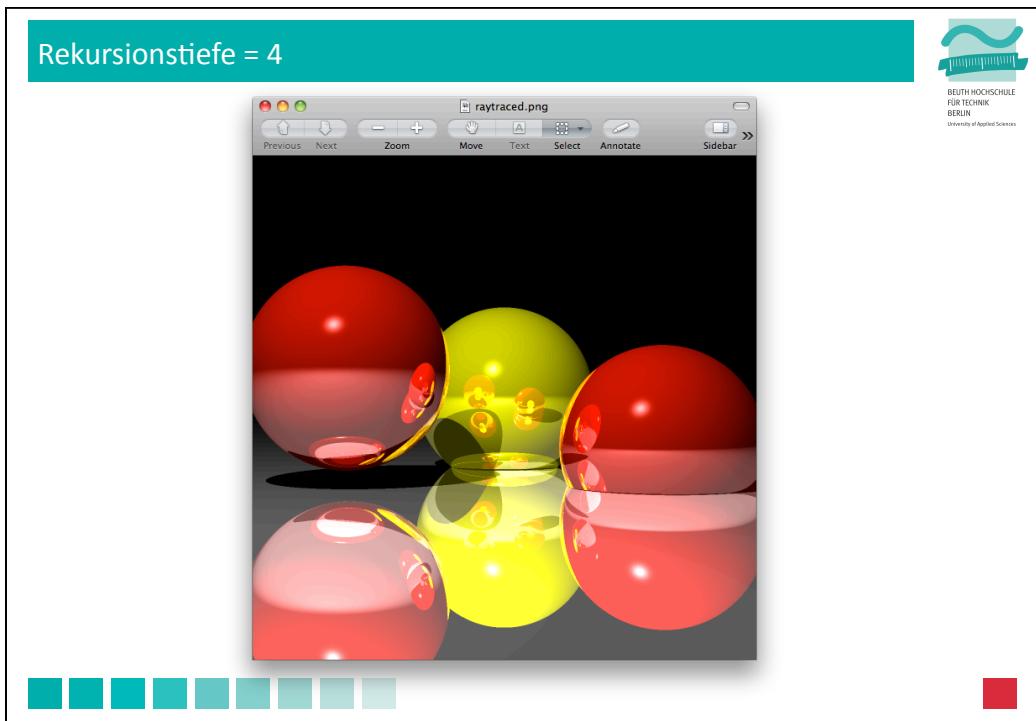
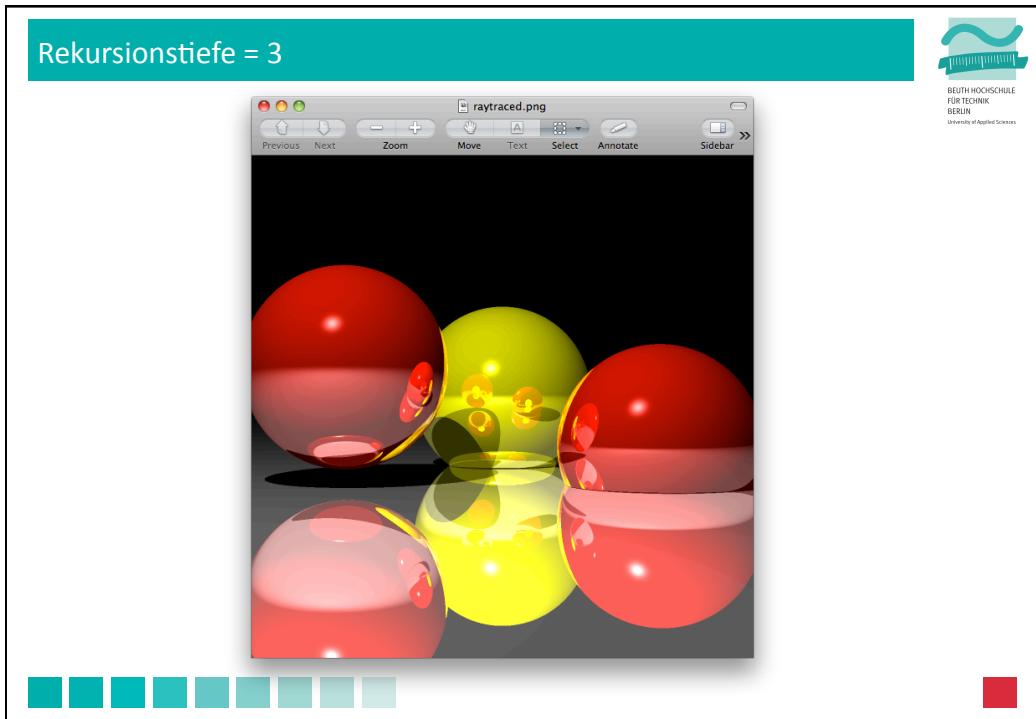

Rekursionstiefe = 0

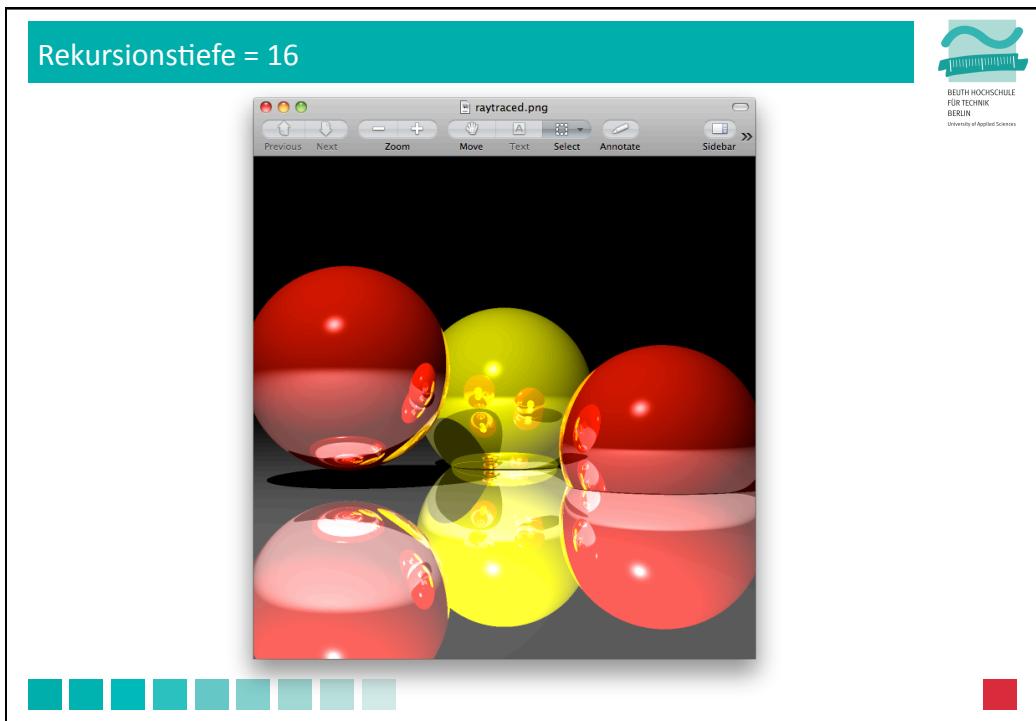


BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences









Möglicher Pseudocode der Shading-Funktion

```
// Phong shading function with shadows, reflections, and refractions
Material.shade(ray: Ray, hit: Hit, scene: Scene, [REDACTED]): Color
{
    Color result = [REDACTED];
    if([REDACTED])
        for each light in scene.getLightSources() do
            if([REDACTED])
                Ray shadowRay = <generate shadow ray>;
                if([REDACTED])
                    result += [REDACTED];
                    if([REDACTED])
                        result += [REDACTED];
                if([REDACTED] &&
                   float R = <calculate reflectance>;
                   if([REDACTED])
                       Hit reflHit = <intersect scene with reflection ray>;
                       Color reflectedColor = reflHit.material().shade(..., [REDACTED]);
                       result += [REDACTED];
                   if([REDACTED])
                       Hit refrHit = <intersect scene with refraction ray>;
                       Color refractedColor = refrHit.material().shade(..., [REDACTED]);
                       result += [REDACTED];
                   return result;
    }
```

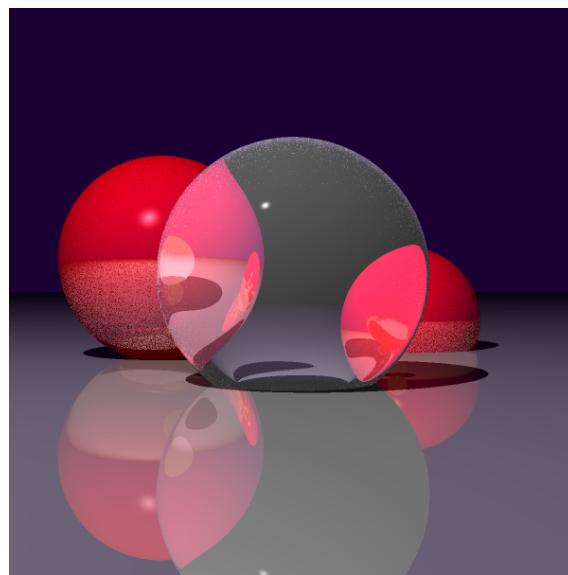
The screenshot shows a Mac OS X interface for a raytracer. The main window displays a 3D scene with several spheres (red, yellow, and green) and a small light source. The title bar says "raytraced.png". The status bar at the bottom indicates a recursion depth of 16. A color calibration strip is visible at the bottom of the screen.

Wie gehe ich beim Schreiben des Shaders am besten vor?

- Schrittweise die verschiedenen Terme der direkten Beleuchtung hinzufügen / testen
 - Ambienter Term
 - Schleife über Lichtquellen: diffuser Term, spekularer Term
- Schatten hinzufügen
 - Schattenstrahlen erzeugen, mit Szene schneiden
 - Treffer auf richtigen Wertebereich testen
- Reflexion hinzufügen
 - Reflexionsstrahl ausrechnen und mit Szene schneiden
 - Shader rekursiv aufrufen
- Refraktion hinzufügen
 - Reflektivität R berechnen und für Reflexion + Refraktion verwenden
 - Refraktionsstrahl ausrechnen und mit Szene schneiden
 - Shader rekursiv aufrufen
- Grad der Modularisierung ist Geschmackssache
 - Persönlicher Tip: nicht zu früh zu sehr in Unterfunktionen zerlegen
 - Zunächst Variablen zur Wiederverwendung von Zwischenergebnissen einführen
 - Dann analysieren, welchen Unterfunktionen man welche Variablen weiterreichen müßte



Problem: „Pixelrauschen“ bei Schatten, Reflexionen, ...

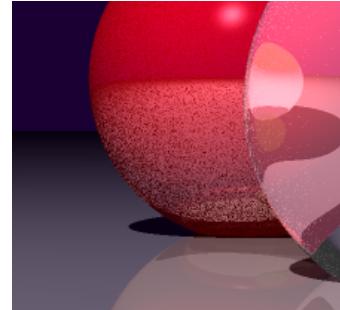
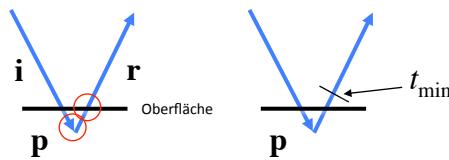


Problem: Objekt wird direkt am Schnittpunkt nochmals getroffen



- Problem

- Primärstrahl i trifft auf rote Kugel K
- Schnittpunkt p wird berechnet
- Ungenauigkeit $\rightarrow p$ hinter der Oberfläche
- Reflexionsstrahl r schneidet Kugel K nochmals!



- Lösung

- Kleiner Offset vom Ursprung des Strahls
- Erlaube Schnittpunkte nur für $t > t_{\min}$



Scene.intersect()



- Betrachtung der Methode `Scene.intersect()`

- Schneide Strahl mit Szene; liefere Treffer zurück
- Verschiedene Anwendungen:
 - Primärstrahlen (aus der Kamera)
 - Strahlen zu den Lichtquellen (Schattenstrahlen)
 - Reflexions- und Refraktionsstrahlen (Sekundärstrahlen)
- Gemeinsamkeiten:
 - Es wird immer der erste Treffer entlang des Strahls benötigt
 - Treffer soll immer vor dem Startpunkt des Strahls liegen
- Unterschiede
 - Startpunkt des Strahls kann entweder das Auge oder ein vorheriger Trefferpunkt sein
 - Bei Schattenstrahlen zählen nur Treffer, die vor der Lichtquelle liegen



Scene.intersect()



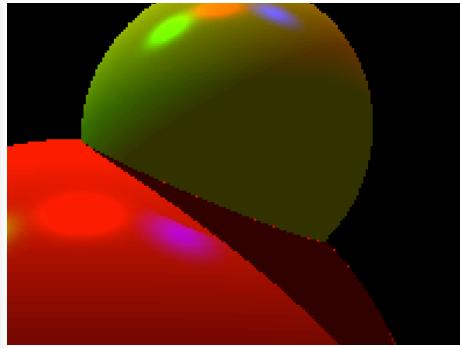
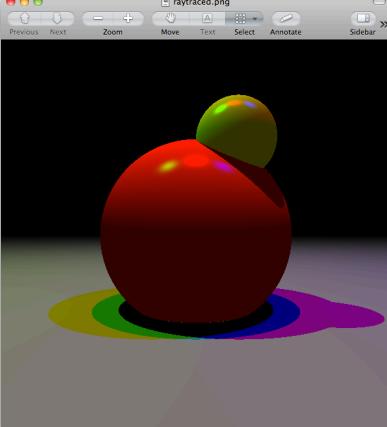
- Empfohlene Signatur der Methode `intersect()`:
 - `Scene.intersect(ray: Ray, tMin: float, tMax: float) : Hit`
- Anwendung / Parameter
 - Primärstrahl: (`<camera ray>`, `0.0f`, `Float.POSITIVE_INFINITY`)
 - Schattenstrahl: (`<shadow ray>`, `<epsilon>`, `<distance to source>`)
 - Sekundärstrahl: (`<secondary ray>`, `<epsilon>`, `Float.POSITIVE_INFINITY`)
- Außerdem später nützlich für Beschleunigungsstrukturen
 - Nur Treffer berücksichtigen, die im Inneren eines Bounding Volumes liegen
(also zwischen Eintritts- und Austrittspunkt)
- Hinweise
 - Wahl des `<epsilon>` abhängig von der Skala / Beschaffenheit der Geometrie



Weitere Hinweise zur
Implementierung



Problempixel

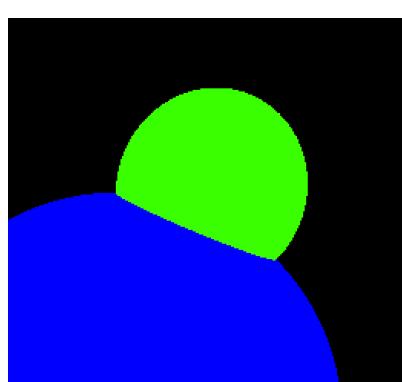
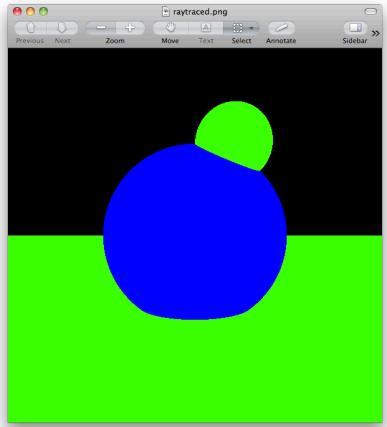


BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

- Wie finde ich heraus, an welchem Teil der Berechnung das liegt?
 - Schnittberechnung?
 - Beleuchtungsberechnung?
 - Schattenberechnung?



Debugging: ist es die Schnittberechnung?

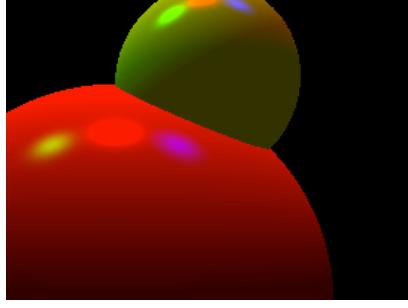
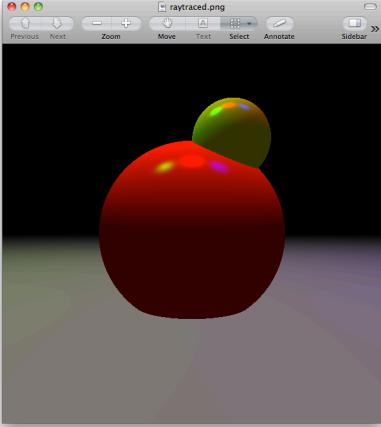


BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

- Schattenberechnung auskommentieren
- Jedem Objekt nur eine ambiente Farbe zuweisen



Debugging: ist es die Beleuchtungs- oder die Schattenberechnung?

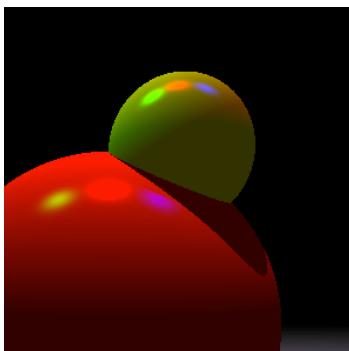
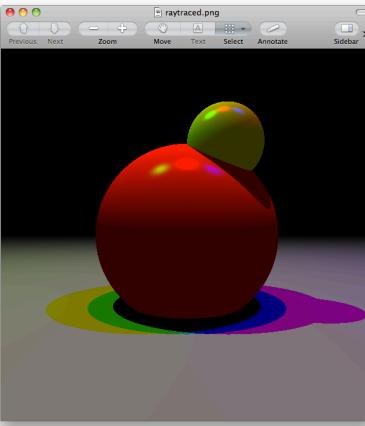


BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

- Schattenberechnung auskommentiert
- Objekten werden „richtige“ Materialien mit Beleuchtungsberechnung zugewiesen



Debugging: Fehler gefunden!



BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

- Fehler in der Schattenberechnung behoben





BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Sonstiges

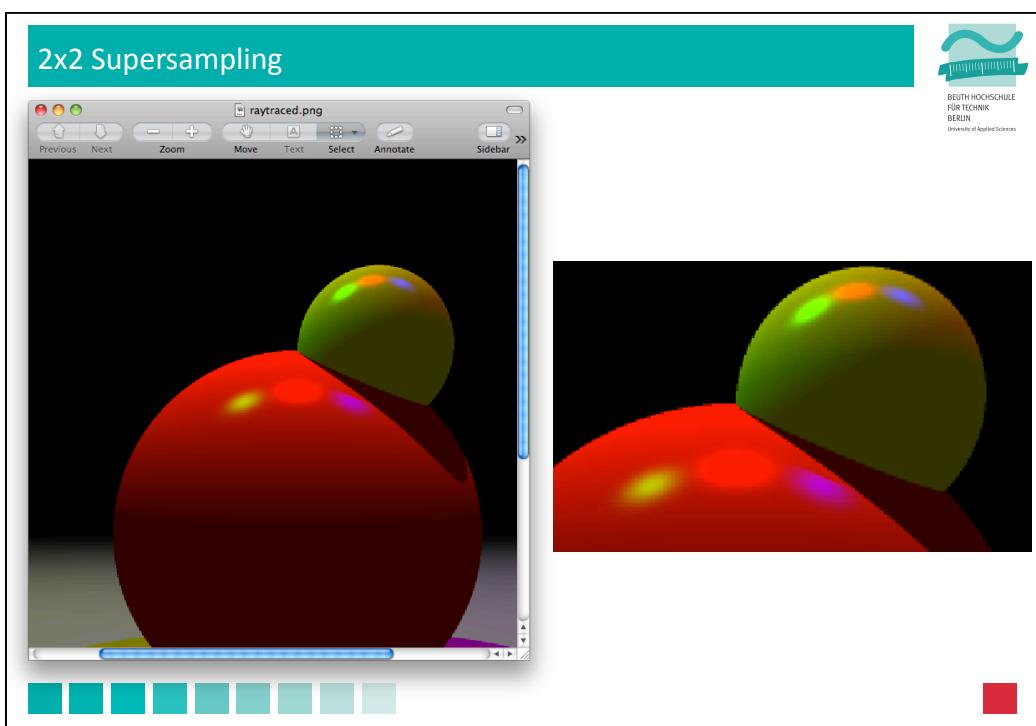
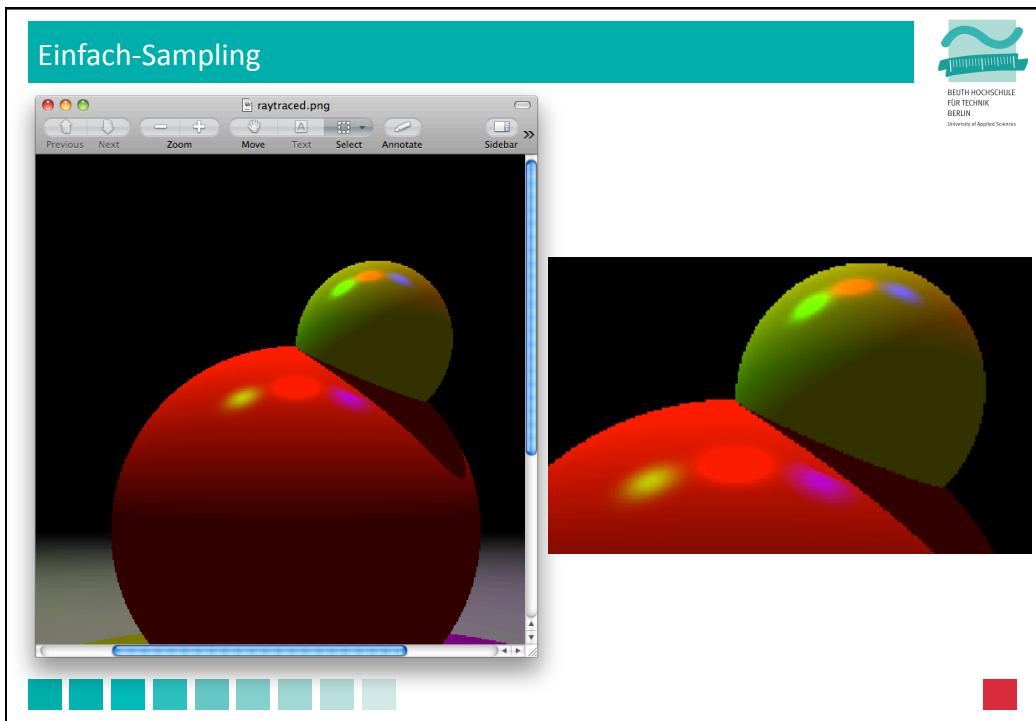


BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Was gibt es beim Raytracing noch?

- Häufige Erweiterung des Raytracers
 - Direktionale Lichtquellen, Lichtquellen mit linearem/quadratischem Abfall
 - Andere geometrische Primitive
 - Beschleunigungsstrukturen
 - Texturen
 - Bump Maps / Verfeinerung der Geometrie
 - Volumen / Participating Media
- Distribution Ray Tracing
 - Tiefenschärfe
 - Flächenlichtquellen
 - Glossy reflection
 - Kaustiken / Schatten semitransparenter Objekte
 - Wie tastet man richtig ab?
- Interactive Ray Tracing
- U.v.a.m. ...





Punktlicht vs. direktionales Licht

The figure consists of three side-by-side screenshots of a raytracer application window. Each window shows a red sphere resting on a gray rectangular plane against a black background. In the first window, labeled "Punktlich von oben" (Point light from above), the sphere has a large, dark shadow cast onto the plane directly beneath it. In the second window, labeled "Direktionales Licht von oben" (Directional light from above), the sphere has a much smaller and more diffused shadow. In the third window, labeled "Beide Lichtquellen" (Both light sources), the sphere has a medium-sized shadow. The raytracer interface includes standard Mac OS X window controls at the top.

Punktlich von oben Direktionales Licht von oben Beide Lichtquellen

- Beobachtungen
 - Größe des Schattens
 - Helligkeitsverteilung in der Ebene

Punktlicht mit quadratischer Abschwächung (Falloff)

The figure consists of three side-by-side screenshots of a raytracer application window. Each window shows two red spheres resting on a gray rectangular plane against a black background. In the first window, labeled "Licht mittig, quadratische Abschwächung" (Light in the center, quadratic falloff), the left sphere is brightly lit while the right sphere is in deep shadow. In the second window, labeled "Licht links, quadratische Abschwächung" (Light on the left, quadratic falloff), the left sphere is brightly lit and the right sphere is partially illuminated. In the third window, labeled "Licht links, keine Abschwächung" (Light on the left, no falloff), both spheres are similarly bright. The raytracer interface includes standard Mac OS X window controls at the top.

Licht mittig,
quadratische Abschwächung Licht links,
quadratische Abschwächung Licht links,
keine Abschwächung



Komplexität und Grenzen des Raytracing



Lichtpfade für direkte und indirekte Beleuchtung



- **Licht-Pfade** (Paul Heckbert, SIGGRAPH 1990)
 - *Reguläre Ausdrücke* repräsentieren den Pfad des Lichts von der Lichtquelle L zum Auge E :
 - LD^*E : mehrfache diffuse Reflexion Radiosity
 - $LD?S^*E$: erst (potentiell) diffuse, dann (mehrfach) spekular Raytracing
 - $LS+DE$: erst 1+ mal spekular, dann diffus („Kaustiken“) Photon Mapping
 - $L(D|S)^*E$: alle möglichen Lichtpfade Path Tracing / Heiliger Gral!
 - Die meisten Rendering-Verfahren können nur bestimmte Typen von Lichtpfaden effizient berechnen!



Grenzen und Komplexität des Raytracing-Ansatzes



- Wie viele Sekundärstrahlen sind nötig, um das einfallende Licht adäquat zu repräsentieren?
 - Hängt stark von den Reflexionseigenschaften ab
 - Flächenlichtquellen → Mehrere Strahlen pro Lichtquelle
 - Diffuse indirekte Beleuchtung → Strahlen in alle Richtungen senden!
 - Importance Sampling: wo keine großen Änderungen der Farbe auftreten können, verwende weniger Samples / Strahlen
- Komplexität des Algorithmus
 - Linear in der Anzahl der Pixel
 - ... in der Anzahl der Szenen-Objekte (später!)
 - Linear in der Anzahl der Lichtquellen
 - Linear in der Anzahl Sekundärstrahlen pro Treffer
 - Exponentiell in der Rekursionstiefe des Algorithmus!!!



Zusammenfassung
und Ausblick



Ray Tracing Zusammenfassung



- Sehr anschaulicher Ansatz
 - Unabhängig von dedizierter Grafik-Pipeline implementierbar
 - Grundbausteine des Raytracing sind vielseitig einsetzbar
- Ein einziges System kombiniert auf einfache Weise:
 - Verdeckungsberechnung
 - *Front-to-back traversal* und *early termination*
 - Schattenberechnung
 - Durch Schattenstrahlen
 - Exakte Simulation einiger Lichtpfade
 - Reflexion (an spiegelnden Flächen)
 - Refraktion (an durchlässigen Grenzflächen)
- Grenzen / Limitationen
 - Wird ineffizient für volle globale Beleuchtungsrechnung
 - Typische indirekte Beleuchtung (diffus) erfordert sehr viele Strahlen



Wofür kann man Raytracing noch verwenden?



- Nützliche Komponenten des Raytracings
 - Picking: welches 3D-Objekt liegt unter Pixel (i,j)?
 - Schneidet der Strahl ein Objekt?
 - Wie weit ist das nächste Objekt entfernt?
- Raytracing in anderen Anwendungen
 - Simulation von Schallwellen (Sound Engineering) und Funkwellen (Cell Phone Coverage)
 - Kollisionserkennung (nächstes Objekt / Abstand zu Objekt; Bounding Volumes zur Beschleunigung)
 - Volumen-Visualisierung
 - ...

