



Computer-Graphik II

Ray-Tracing



G. Zachmann
Clausthal University, Germany
cg.in.tu-clausthal.de



Effekte für eine realistische Darstellung

- Das lokale Beleuchtungsmodell (CG1) versagt bei folgenden Effekten:
 - (Soft) Shadows (Halbschatten)
 - Reflexionen (*reflection*, z.B. Spiegel)
 - Brechung (*refraction*, z.B. Wasser, Glas)
 - Indirekte Beleuchtung ("*color bleeding*")
 - Beugung (*dispersion*)
 - ...

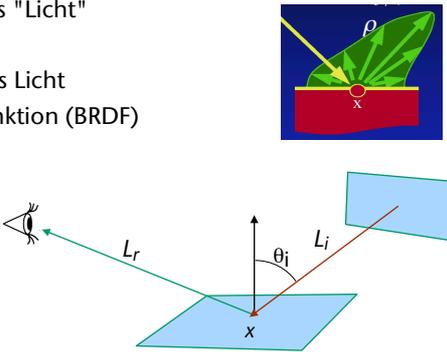
G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 2

Globale Beleuchtungsrechnung

- Ziel: **Photorealistisches Rendering**
- Die "Lösung": die **Rendering-Gleichung** [Kajiya, Siggraph 1986]

$$L_r(x, \omega_r) = L_e(x, \omega_r) + \int_{\Omega} \rho(x, \omega_r, \omega_i) L_i(x, \omega_i) \cos(\theta_i) d\omega_i$$

L_i = aus Richtung ω_i inzidentes "Licht"
 L_e = emittiertes Licht
 L_r = in Richtung ω_r reflektiertes Licht
 ρ = Reflexionskoeffizientenfunktion (BRDF)
 Ω = Halbkugel um Normale




G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 3

Die Erfindung des Ray-Tracings

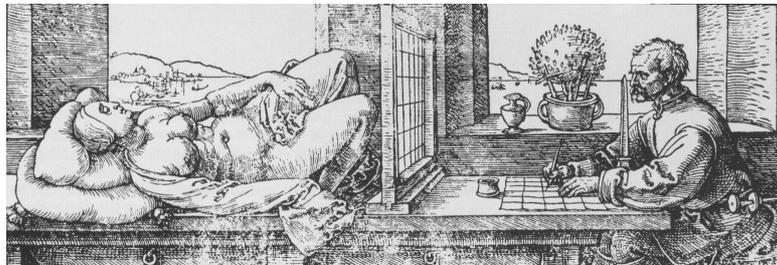
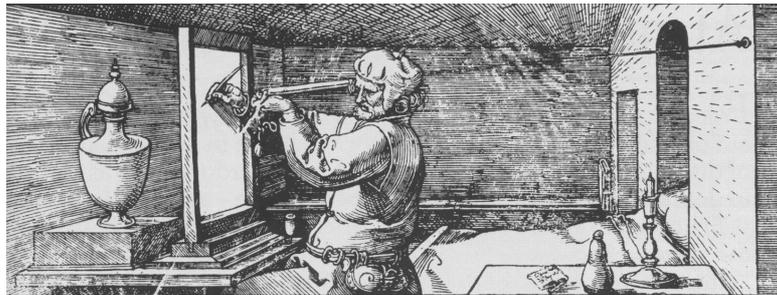
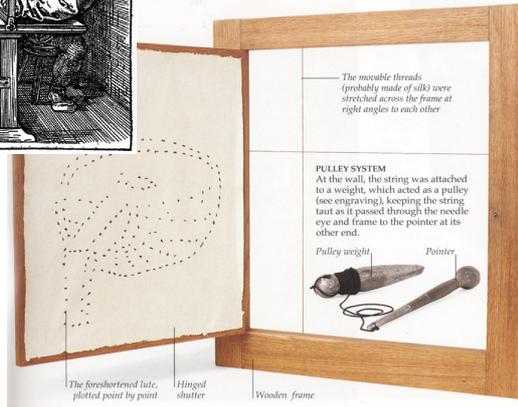
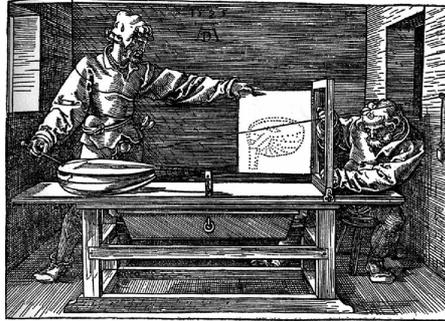
- Analytische Lösung ist unmöglich!
- Die Rendering Gleichung kann als rekursive Funktion aufgefaßt werden
- Daraus folgen praktische **Approximations-Verfahren**, die auf der Verfolgung des Lichts entlang Strahlen beruhen
 - **Ray tracing** [Whitted, Siggraph 1980, "An Improved Illumination Model for Shaded Display"]
 - **Radiosity** [Goral et. al, Siggraph 1984, "Modeling the Interaction of Light between diffuse Surface"]
 - **Monte Carlo Verfahren**



Turner Whitted,
Microsoft Research

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 4

Albrecht Dürers "Ray-Casting-Maschinen" [16. Jhrdt.]



Rekursives Ray-Tracing

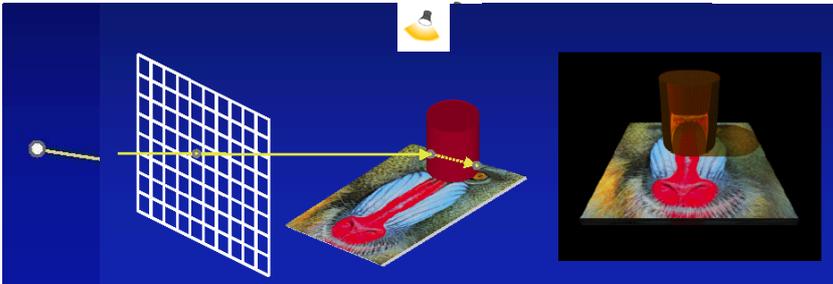
- Algorithmus zur Approximation der Rendering-Gleichung
- Modelliert werden nur:
 - Reflexion
 - Brechung
 - Verdeckungsrechnung
 - Schatten
- Strahlen werden nur in Richtung des **reflektierten** bzw. **gebrochenen** Strahls verfolgt
- Annahmen:
 - Punktlichtquellen
 - Phong-Modell
 - keine Halbschatten



G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 7

Funktionsweise

1. Synthetische Kamera = Augpunkt + Bildebene in Weltkoordinaten
2. Schieße Strahlen vom Augpunkt aus durch die Pixel in die Szene
3. Falls der Strahl mehr als ein Objekt schneidet, betrachte nur den ersten Schnittpunkt
4. Schieße weitere Strahlen vom dort zu allen Lichtquellen (Schattenstrahlen; "shadow feelers")
5. Treffen diese Schattenstrahlen auf ein Objekt, so liegt der betrachtete Flächenpunkt im Schatten. Ansonsten wird das Phong-Beleuchtungsmodell ausgewertet
6. Ist das sichtbare Objekt spiegelnd, dann schieße weiteren reflektierten Strahl in die Szene
7. Ist das Objekt transparent, so wird zusätzlich ein gebrochener Strahl weiterverfolgt



G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 8

Der Strahlbaum

- Grundidee des Raytracing: Strahlengänge von den Lichtquellen bis zum Auge konstruieren, aber dabei beim Auge starten und diese Strahlengänge rückwärts "suchen"
- Ergibt (konzeptionell!) einen Strahlenbaum:

E1 = Primärstrahl
 Ri = reflektierter Strahl
 Ti = transmittierter Strahl

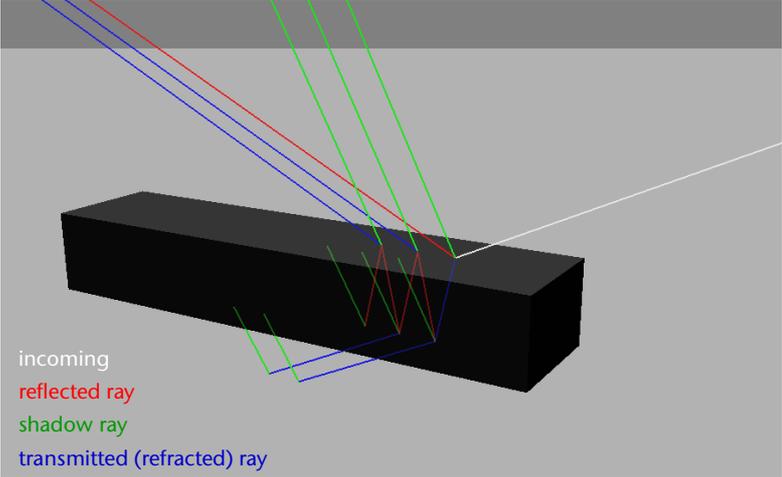
Si = Schattenstrahl

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 9

Die antike Erklärung des Sehens: Sehstrahlen

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 10

Visualisierung eines Strahlbaumes (eignet sich hervorragend zum Debugging)



incoming
reflected ray
shadow ray
transmitted (refracted) ray

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 11

Das Beleuchtungsmodell

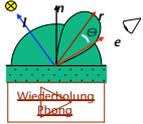
Beleuchtung auf der Fläche

$$L_{\text{ges}} = L_{\text{Phong}} + r_s L_s + r_t L_t$$

r_s = Reflexionskoeffizient für das reflektierte Licht L_s
 r_t = Transmissionskoeffizient für das transmittierte Licht L_t

Abbruch der Rekursion:

- Falls maximale Rekursionstiefe erreicht; oder/und,
- Falls Beitrag zur Beleuchtung zu klein (schrumpft wie r_t^n)

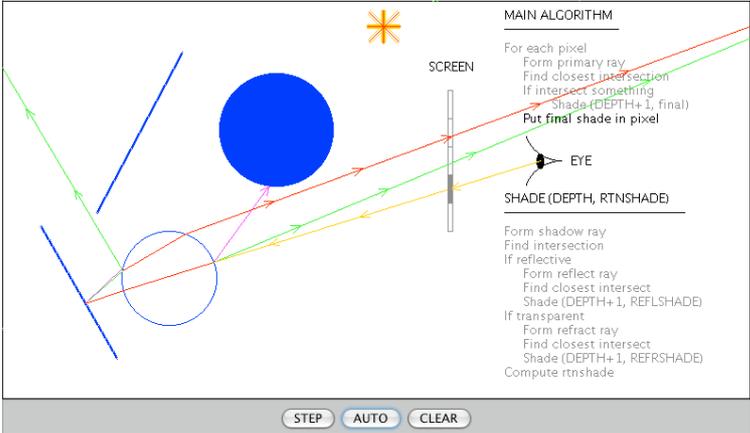




Rek. Tiefe: 3 Rek. Tiefe: 5 Rek. Tiefe: 100

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 12

Demo



MAIN ALGORITHM

```

For each pixel
  Form primary ray
  Find closest intersection
  If intersect something
    Shade (DEPTH+1, final)
  Put final shade in pixel

```

SHADE (DEPTH, RTNSHADE)

```

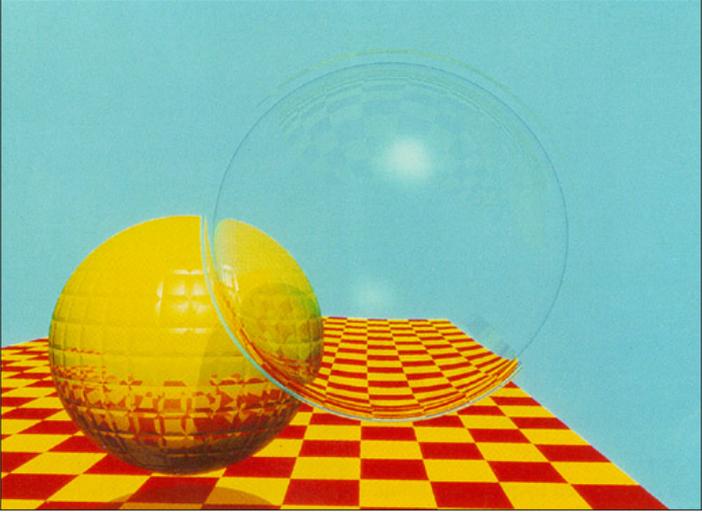
Form shadow ray
Find intersection
If reflective
  Form reflect ray
  Find closest intersect
  Shade (DEPTH+1, REFLSHADE)
If transparent
  Form refract ray
  Find closest intersect
  Shade (DEPTH+1, REFRSHADE)
Compute rtshade

```

http://www.siggraph.org/education/materials/HyperGraph/raytrace/rt_java/raytrace.html

G. Zachmann Computer-Graphik 2 — SS 10 Ray-Tracing 13

Eines der ersten Ray-Tracing-Bilder



Turner Whitted 1980

G. Zachmann Computer-Graphik 2 — SS 10 Ray-Tracing 14

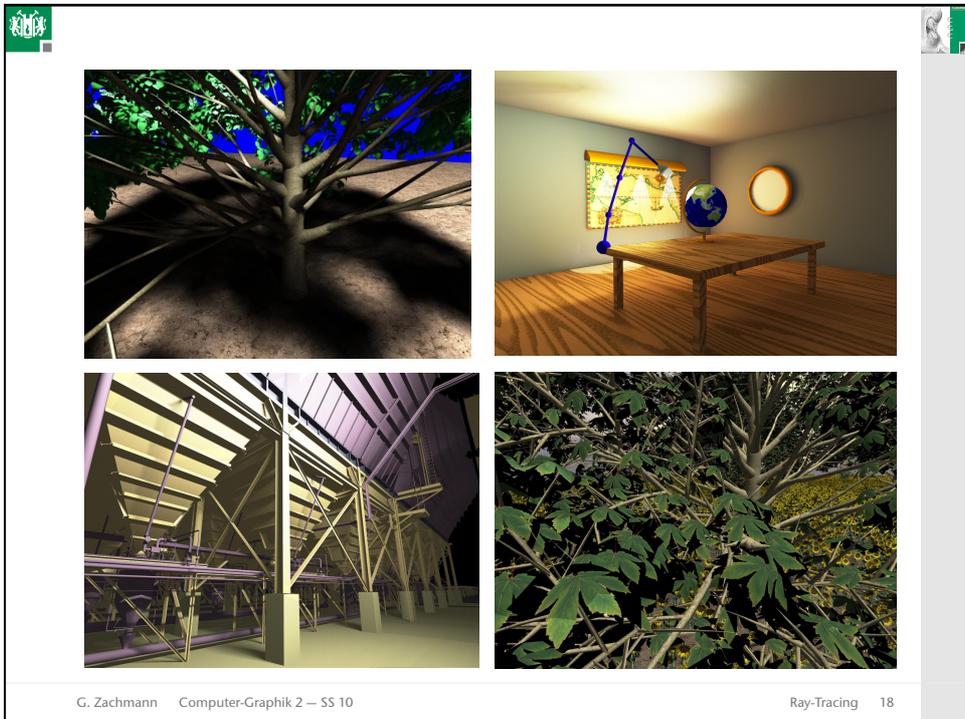
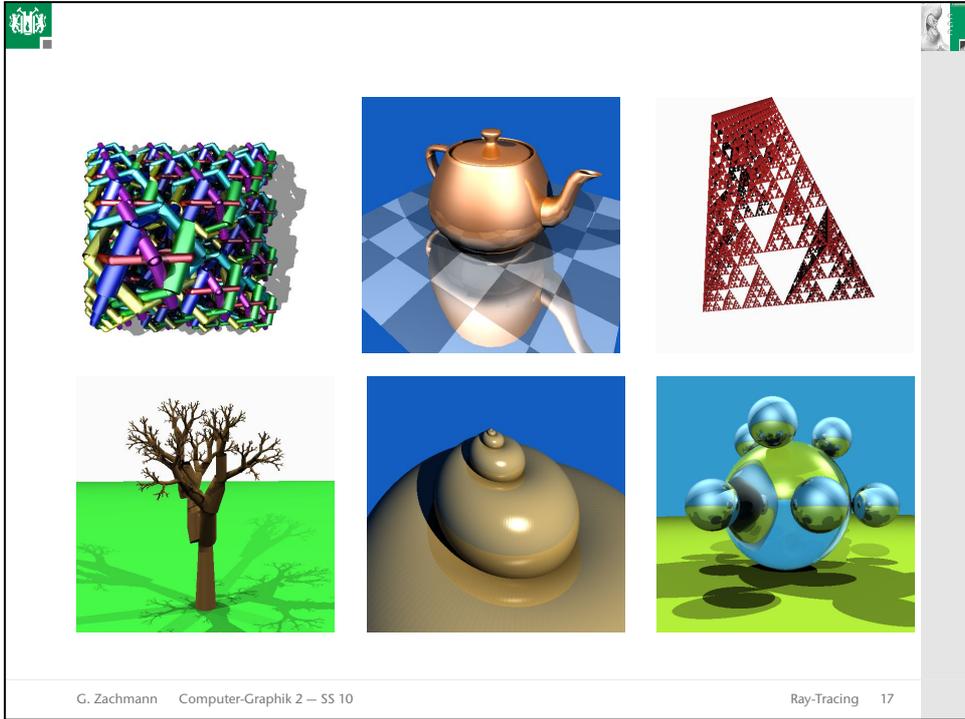
Beispiele

Jensen, Lightscape

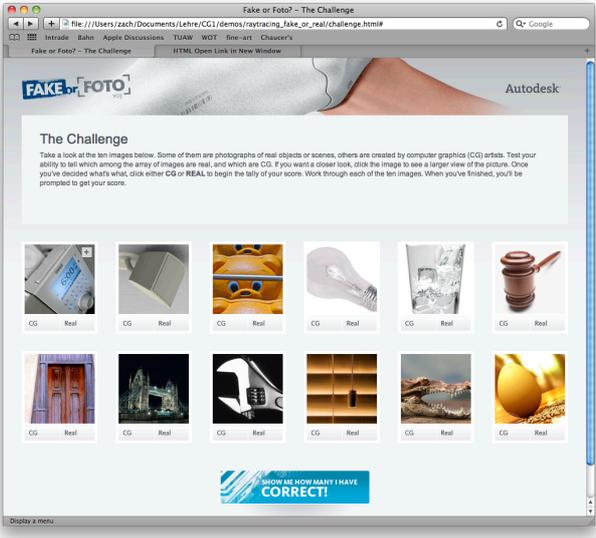
G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 15

Objekt "sphere flake" aus der "Standard Procedural Databases" (SPD) von Eric Haines
[<http://www.acm.org/tog/resources/SPD/>].

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 16

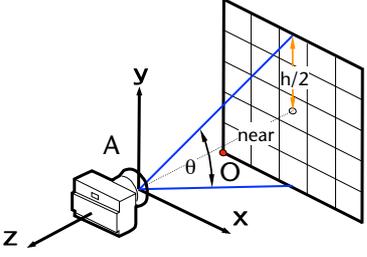


Fake or Real?



G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 19

Die Kamera (ideale Lochkamera)



$$h = 2 \cdot \text{near} \cdot \tan \frac{\theta}{2}$$

$$O = A - \text{near} \cdot z - \frac{b}{2}x - \frac{h}{2}y$$

Die Main-Loop eines Ray-Tracers

```
for ( t = 0; t < h; t ++ )
  for ( s = 0; s < b; s ++ )
    ray.from = A
    ray.at = O + s·x + t·y
    trace ( 0, ray, &color );
    putPixel ( x, y, color );
```

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 20

Älteste Abbildung einer Lochkamera

*Solis deliquium Anno Christi
1544. Die 24. Januarij
L. onanij*

Von R. Gemma Frisius, 1545

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 21

Camera Obscura

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 22

Andere sonderbare Kameras

- Mit Ray-Tracing sind andere Projektionen sehr einfach
- Z.B. Fischaue, Omnimax, Panorama



G. Zachmann Computer-Graphik 2 – SS 10

Ray-Tracing 23

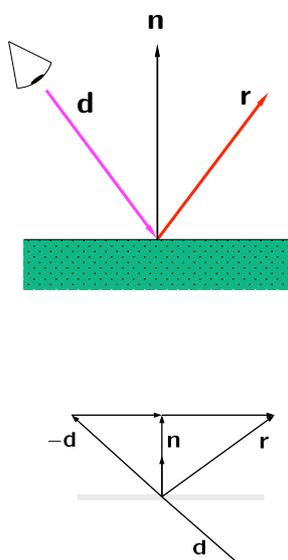


G. Zachmann Computer-Graphik 2 – SS 10

Ray-Tracing 24

Sekundärstrahlen

- Annahme: Hit zwischen Primärstrahl und Szene gefunden
- Reflektierter Strahl:



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

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

wobei $\|\mathbf{n}\| = 1$

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 25

Gebrochener Strahl

- Brechungsgesetz [Snell ~1600]:

$$n_1 \sin \theta_1 = n_2 \sin \theta_2$$

- Der transmittierte Strahl:

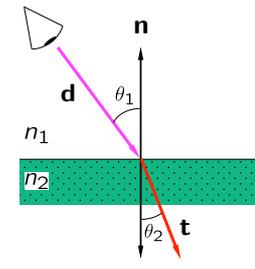
$$\mathbf{t} = \frac{n_1}{n_2} (\mathbf{d} + \mathbf{n} \cos \theta_1) - \mathbf{n} \cos \theta_2$$

$$\cos \theta_1 = -\mathbf{d} \cdot \mathbf{n}$$

$$\cos^2 \theta_2 = 1 - \frac{n_1^2}{n_2^2} (1 - (\mathbf{d} \cdot \mathbf{n})^2)$$

- Brechungsindizes:

	Luft	Wasser	Glas	Diamant
	1.0	1.33	1.5 - 1.7	2.4



G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 26

$|\mathbf{n}| = |\mathbf{b}| = 1$

$\mathbf{t} = \cos \theta_2 \cdot (-\mathbf{n}) + \sin \theta_2 \cdot \mathbf{b}$

$\mathbf{d} = \cos \theta_1 \cdot (-\mathbf{n}) + \sin \theta_1 \cdot \mathbf{b}$

$\mathbf{b} = \frac{\mathbf{d} + \mathbf{n} \cdot \cos \theta_1}{\sin \theta_1}$

$\mathbf{t} = -\mathbf{n} \cdot \cos \theta_2 + \frac{\sin \theta_2}{\sin \theta_1} (\mathbf{d} + \mathbf{n} \cdot \cos \theta_1)$

cos θ_2 ausrechnen:

$$\sin \theta_2 = \frac{n_1}{n_2} \sin \theta_1$$

$$\sin^2 + \cos^2 = 1$$

$$\cos^2 \theta_2 = 1 - \left(\frac{n_1}{n_2} \sin \theta_1\right)^2$$

$\frac{\sin \theta_2}{\sin \theta_1} = \frac{n_1}{n_2}$
 $\cos \theta_1 = \mathbf{n} \cdot (-\mathbf{d})$

G. Zachmann Computer-Graphik 2 – SS 10
Ray-Tracing 27

▪ Totalreflexion:

wenn Radikand $< 0 \Leftrightarrow \sin \theta_1 \geq \frac{n_2}{n_1}$




G. Zachmann Computer-Graphik 2 – SS 10
Ray-Tracing 28

Refraction and the Lifeguard Problem

- Running is faster than swimming

The diagram illustrates the lifeguard problem. On the left, a person is in trouble in the water. A red line shows the path: swim from the person to the beach, then run along the beach to the lifeguard. A dashed white line shows a direct path from the person to the lifeguard. Labels include 'Water', 'Beach', 'Person in trouble', 'Swim', 'Run', and 'Lifeguard'. A photo of the Baywatch lifeguards is also shown.

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 29

Wirkung des Brechungsindex

The image shows a grid of 8 spheres on a green surface, illustrating the effect of increasing refractive index. The spheres are labeled with refractive index values: n=1.0, n=1.1, n=1.2, n=1.3, n=1.4, n=1.5, n=1.6, and n=1.7. As the refractive index increases, the sphere's appearance changes from transparent to more opaque and distorted.

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 30

Welches ist die "richtige" Normale?

- Klappt die korrekte Berechnung des reflektierten und des gebrochenen Strahls auch, wenn die Normale in die "falsche" Richtung zeigt?

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 31

Was können wir hier noch nicht simulieren?



G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 32

Fresnel-Terme

- Beim Wechsel von einer Materie in eine andere wird immer ein Anteil Licht reflektiert, der restliche Anteil gebrochen
- Der **Reflexionskoeffizient** ρ hängt ab vom Brechungsindex der beiden Materialien und vom Einfallswinkel:

$$\rho_{\parallel} = \frac{n_2 \cos \theta_1 - n_1 \cos \theta_2}{n_2 \cos \theta_1 + n_1 \cos \theta_2}$$

$$\rho_{\perp} = \frac{n_1 \cos \theta_1 - n_2 \cos \theta_2}{n_2 \cos \theta_1 + n_1 \cos \theta_2}$$

$$\rho = \frac{1}{2} \cdot (\rho_{\parallel}^2 + \rho_{\perp}^2)$$

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 33

- Beispiel:**
 - Luft ($n = 1.0$) nach Glas ($n = 1.5$), senkrechter Lichteinfall:

$$\rho_{\parallel} = \frac{1.5 - 1}{1.5 + 1} = \frac{1}{5} \quad \rho_{\perp} = \frac{1 - 1.5}{1.5 + 1} = \frac{1}{5} \quad \rho = \frac{1}{2} \cdot \frac{2}{25} = 4\%$$
 - D.h., beim Übergang von Luft nach Glas wird 4% des Lichtes reflektiert, der Rest gebrochen
- Approximation der Fresnel-Terme [Schlick 1994]:

$$\rho(\theta) \approx \rho_0 + (1 - \rho_0)(1 - \cos \theta)^5$$

$$\rho_0 = \left(\frac{n_2 - 1}{n_2 + 1} \right)^2$$

wobei ρ_0 der Fresnel-Term des senkrechten Lichteinfalls ist und θ der Winkel im dünneren Medium (also der größere).

 - $1 - \rho$ ergibt dann den transmittierten Anteil

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 34

Beispiel für Brechung unter Berücksichtigung der Fresnel-Terme

n=1.0 n=1.1 n=1.2 n=1.3

n=1.4 n=1.5 n=1.6 n=1.7

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 35

Dämpfung im Medium

- Die durch ein Medium transportierte Lichtintensität schwächt sich mit zunehmender Entfernung gemäß dem **Lambert-Beer'schen Gesetz** ab:

$$I(s) = I_0 e^{-\alpha s}$$

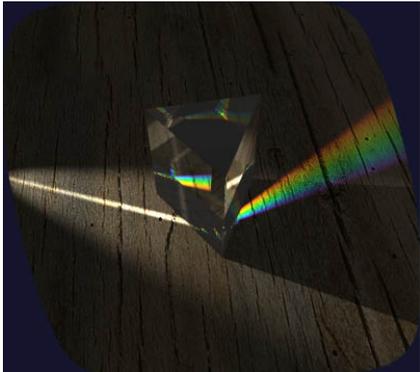
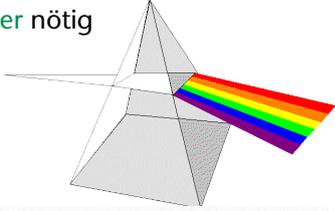
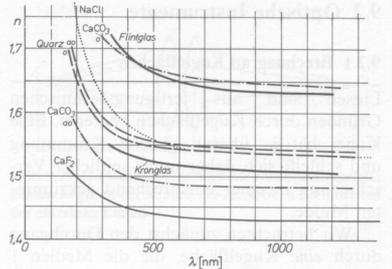
wobei α eine Materialkonstante ist und s der im Medium zurückgelegte Weg.

- α kann auch von der Wellenlänge abhängen

G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 36

Dispersion

- Brechungsindex ist abhängig von der Wellenlänge
- Diese Effekte lassen sich allerdings in RGB nicht mehr abbilden; hierzu wäre ein „spektraler“ Ray-Tracer nötig

G. Zachmann Computer-Graphik 2 – SS 10
Ray-Tracing 37

Giovanni Battista Pittoni, 1725, "An Allegorical Monument to Sir Isaac Newton"





Pink Floyd, *The Dark Side of the Moon*

G. Zachmann Computer-Graphik 2 – SS 10
Ray-Tracing 38

Beispiel mit Fresnel-Term und Dispersion



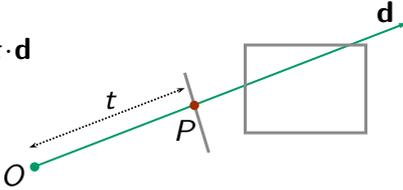
G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 39

Schnittberechnungen

- Der wesentliche Bestandteil der Rechenzeit
- Gegeben: Menge von Objekten (Polygone, Kugeln, ...) und ein Strahl

$$P(t) = O + t \cdot \mathbf{d}$$

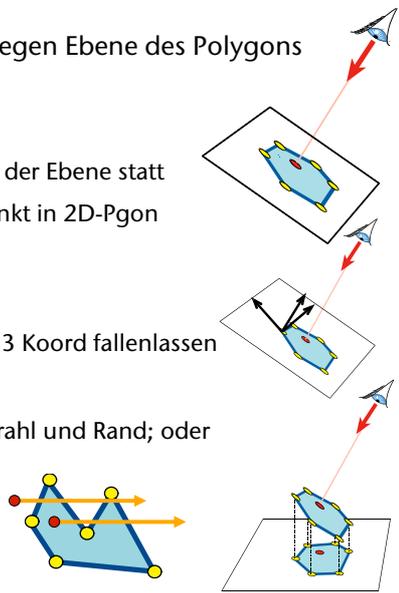
- Gesucht: Linienparameter t des ersten Schnittpunktes $P = P(t)$ mit der Szene



G. Zachmann Computer-Graphik 2 – SS 10 Ray-Tracing 40

Schnitt Strahl—Polygon

- Schneide Strahl (parametrisch) gegen Ebene des Polygons (implizit) → Punkt
- Teste "Punkt in Polygon"
 - Dieser Test findet ausschließlich in der Ebene statt
 - 3D-Punkt in 3D-Polygon \Leftrightarrow 2D-Punkt in 2D-Polygon
- Projiziere Punkt & Polygon
 - Entlang der Normale: zu teuer
 - Auf Koord.ebene: einfach eine der 3 Koord fallenlassen
- Test "Punkt in Polygon":
 - Zähle Anzahl Schnitte zwischen Strahl und Rand; oder
 - Bestimme "Winding Number"



G. Zachmann Computer-Graphik 2 — SS 10 Ray-Tracing 41

Interludium: die vollständige Ray-Tracing-Routine

```

traceRay( ray ):
  hit = intersect( ray )
  if no hit:
    return no color
  reflected_ray = reflect( ray, hit )
  reflected_color = traceRay( reflected_ray )
  refracted_ray = refract( ray, hit )
  refracted_color = traceRay( refracted_ray )
  for each lightsource[i]:
    shadow_ray = calcLightFeeler( hit, lightsource[i] )
    if intersect( shadow_ray ):
      light_color[i] = 0
  overall_color = shade( hit,
                        reflected_color,
                        refracted_color,
                        light_color )
  return overall_color

```

hit ist eine Datenstruktur, die alle Infos über einen Schnitt zwischen Strahl und Szene enthält, u.a. Schnittpunkt, Objekt, Normale, ...

Diese **intersect**-Funktion kann deutlich optimiert werden gegenüber der obigen; außerdem interessiert nur ein Schnitt vor der Lichtquelle.

Wertet die Beleuchtungsgleichung für das getroffene Obj aus.

G. Zachmann Computer-Graphik 2 — SS 10 Ray-Tracing 42