



## Computer-Graphik II Ray-Tracing




G. Zachmann  
Clausthal University, Germany  
[cg.in.tu-clausthal.de](http://cg.in.tu-clausthal.de)



## Die antike Erklärung des Sehens: Sehstrahlen




G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 2



## Effekte für eine realistischen Darstellung

- Das lokale Beleuchtungsmodell versagt bei folgenden Effekten
- (Soft) Shadows (Halbschatten)
- Reflexion (Spiegel und Glanz)
- Transparenz (Wasser, Glas)
- Interreflexion ("color bleeding")
- ...

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 3

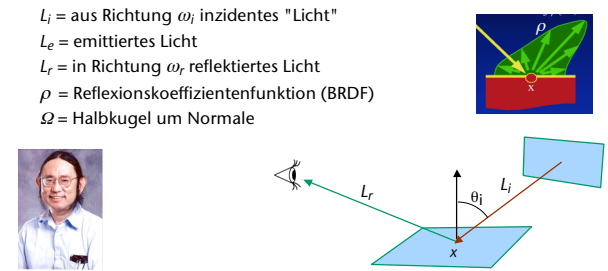



## Globale Beleuchtungsrechnung

- 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  $\omega_i$  inzidentes "Licht"  
 $L_e$  = emittiertes Licht  
 $L_r$  = in Richtung  $\omega_r$  reflektiertes Licht  
 $\rho$  = Reflexionskoeffizientenfunktion (BRDF)  
 $\Omega$  = Halbkugel um Normale

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 4

- 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 07 Ray-Tracing 5

### Rekursives Ray-Tracing

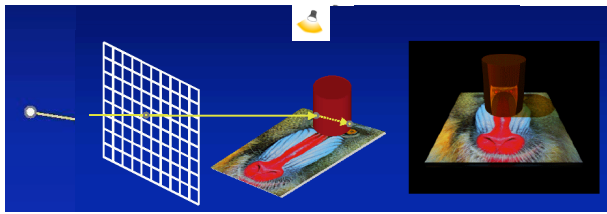
- Algorithmus zur Approximation der Rendering-Gleichung
- Modelliert werde nur:
  - Reflektion
  - Beugung
  - Verdeckungsrechnung
  - Schatten
- Strahlen werden nur in Richtung des **reflektierten** bzw. **gebrochenen** Strahls verfolgt
- Annahmen:
  - Punktlichtquellen
  - Phong-Modell
  - keine Halbschatten



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 6

### Funktionsweise

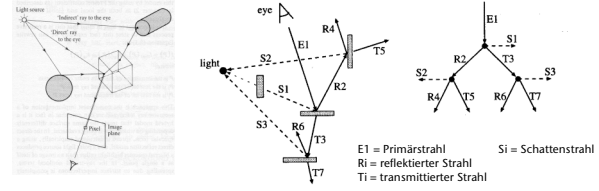
- Synthetische Kamera = Augpunkt + Bildebene in Weltkoordinaten
- Schieße Strahlen vom Augpunkt aus durch die Pixel in die Szene
- Falls der Strahl mehr als ein Objekt schneidet, betrachte nur den ersten Schnittpunkt
- Schieße weitere Strahlen von dort zu allen Lichtquellen (Schattenstrahlen; "shadow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



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 7

### 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 07 Ray-Tracing 8

Visualisierung eines Strahlbaumes (eignet sich hervorragend zum Debugging)

incomig  
reflected ray  
shadow ray  
transmitted (refracted) ray

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 9

Das Beleuchtungsmodell

Beleuchtung auf der Fläche

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

$r_s$  = 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 07 Ray-Tracing 10

Demo

MAIN ALGORITHM

For each pixel

- Form primary ray
- Find closest intersection
- If intersect computing
  - 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 rtnshade

[http://www.siggraph.org/education/materials/HyperGraph/raytrace/rt\\_java/raytrace.html](http://www.siggraph.org/education/materials/HyperGraph/raytrace/rt_java/raytrace.html)

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 11

Eines der ersten Ray-Tracing-Bilder

Turner Whitted 1980

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 12

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

The movable threads, probably made of silk, were stretched across the frame at right angles to each other.

**PULLEY SYSTEM**  
At the wall, the string was attached to a weight, which acted as a pulley (see diagram), keeping the string taut as it passed through the needle eye and frame to the pointer at its other end.

The horizontal line, plotted point by point

Hinged shutter

Wooden frame

Pulley weight

Pointer

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 13

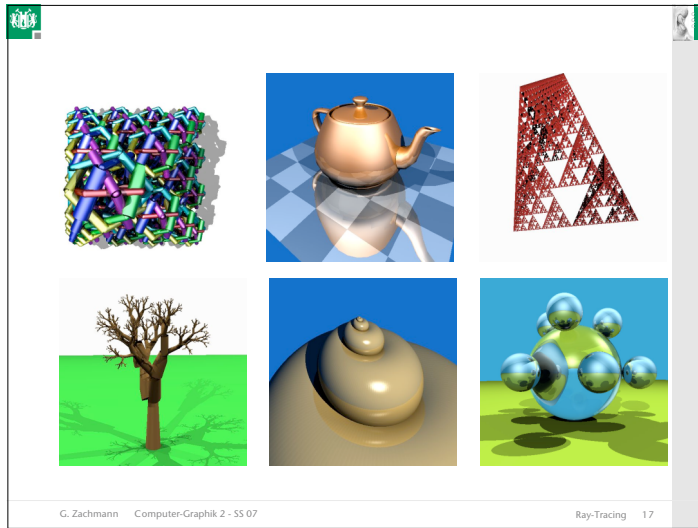
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 14

### Beispiele

jensen, Lightscape

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 15

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 16



### 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 );
    
```

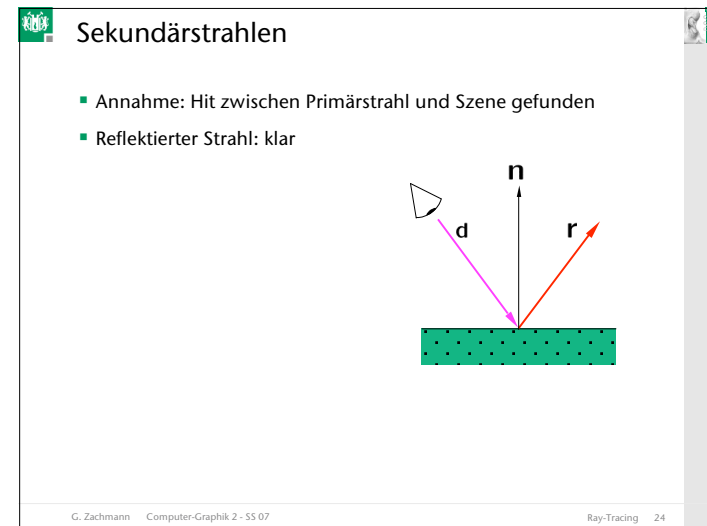
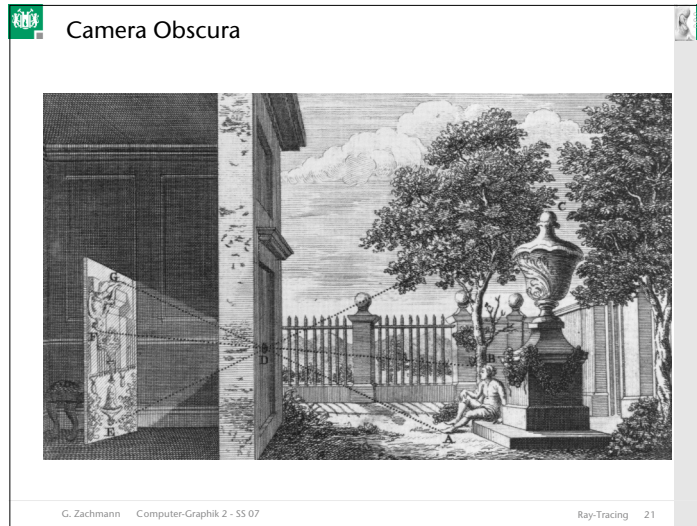
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 19

### Älteste Abbildung einer Lochkamera

Solis deliquium Anno Christi  
1544. Die 24. Januarij  
Lovanij

Von R. Gemma Frisius, 1545

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 20

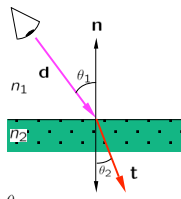


### Gebrochener Strahl

- Brechungsgesetz [Snell ~1600]:
 
$$n_1 \sin \theta_1 = n_2 \sin \theta_2$$
- Der transmittierte Strahl:
 
$$\mathbf{t} = \frac{n_1}{n_2}(\mathbf{d} + \mathbf{n} \cos \theta_1) - \mathbf{n} \cos \theta_2$$

$$\cos \theta_1 = -\mathbf{dn}$$

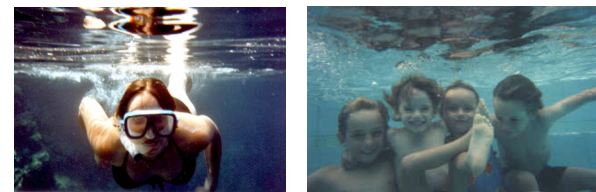
$$\cos^2 \theta_2 = 1 - \frac{n_1^2}{n_2^2} (1 - (\mathbf{dn})^2)$$



Brechungsindizes:	Luft	Wasser	Glas	Diamant
	1.0	1.33	1.5 - 1.7	2.4

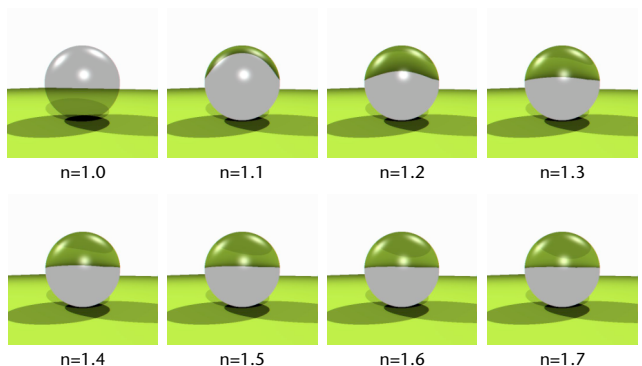
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 25

- Totalreflexion:
 
$$\text{wenn Radikand} < 0 \Leftrightarrow \sin \theta_2 \leq \frac{n_1}{n_2}$$



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 26

### Wirkung des Brechungsindex



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 27

### Was können wir hier noch nicht simulieren?



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 28

### Fresnel-Terme

- Beim Wechsel von einer Materie in eine andere wird immer ein Anteil Licht reflektiert, der restliche Anteil gebrochen
- Der Reflexionskoeffizient  $\rho$  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 07 Ray-Tracing 29

- 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  $\rho_0$  der Fresnel-Term des senkrechten Lichteinfalls ist und  $\theta$  der Winkel im dünneren Medium (also der größere).
  - 1- $\rho$  ergibt dann den transmittierten Anteil

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 30

### Beispiel für Brechung unter Berücksichtigung der Fresnel-Terme

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 31

### Dämpfung im Medium

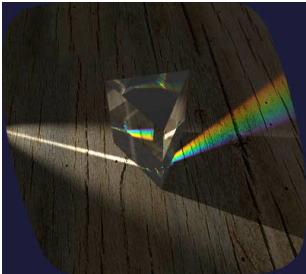
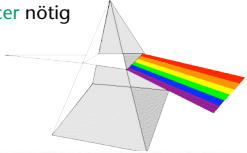
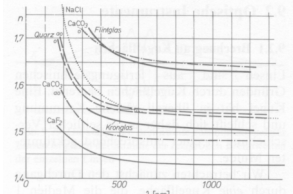
- Die durch ein Medium transportierte Lichtintensität schwächt sich mit zunehmender Entfernung gemäß dem Lambert-Beer'schen Gesetz ab:
 
$$I(s) = I_0 e^{-\alpha s}$$
 wobei  $\alpha$  eine Materialkonstante ist und  $s$  der im Medium zurückgelegte Weg.
- $\alpha$  kann auch von Wellenlänge abhängen

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 32




### Dispersion

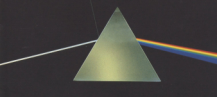
- Brechungsindex ist abhängig von der Wellenlänge
- Diese Effekte lassen sich allerdings in RGB nicht mehr abbilden; hierzu wäre ein „spektraler“ Ray-Tracer nötig

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 33




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



Pink Floyd, The Dark Side of the Moon

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 34

### Beispiel mit Fresnel-Term und Dispersion



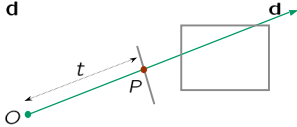
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 35

### Schnittberechnungen

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

$$P(t) = O + t \cdot d$$

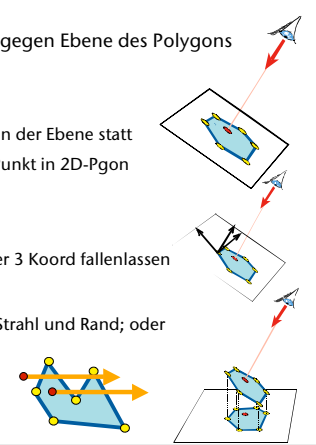
- Gesucht: Linienparameter  $t$  des ersten Schnittpunktes  $P = P(t)$  mit der Szene



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 36

### 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 ↔ 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 07 Ray-Tracing 37

### 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(ray):
light_color[i] = 1
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, ...

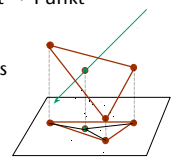
Diese intersect-Funktion kann deutlich optimiert werden gegenüber der obigen; außerdem interessiert nur ein Schnitt vor der Lichtquelle.

Wertet die Beleuchtungsgleichung für das getroffene Obj aus.

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 38

### Schnitt Strahl—Dreieck [Badouel 1990]

- Verwende Methode Strahl—Polygon; oder
- Cleverer sein: baryzentrische Koordinaten + Projektion
- Schneide Strahl mit Ebene (Normalenform) → t → Punkt
- Projiziere Punkt & Dreieck in Koord.ebene
- Berechne baryzentrische Koord. des 2D-Punktes
- Baryzentrische Koord. des 2D-Punktes = baryzentrische Koord. des 3D-Punktes!
- 3D-Punkt in Dreieck ↔  $\alpha, \beta, \gamma > 0, \alpha + \beta + \gamma < 1$
- Alternative Methode: siehe Möller & Haines "Real-time Rendering"
- Code: <http://jgt.akpeters.com/papers/MollerTrumbore97/>
- Geht noch schneller, falls Schnittpunkt nicht nötig [Segura & Feito]

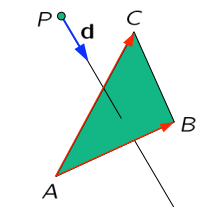


G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 39

### Alternative Schnittberechnung Strahl—Dreieck [Möller]

- Geradengleichung  $X = P + t \cdot d$
- Ebenengleichung  $X = A + r \cdot (B - A) + s \cdot (C - A)$
- Gleichsetzen  $-t \cdot d + r \cdot (B - A) + s \cdot (C - A) = P - A$
- In Matrixschreibweise  $(-d \ B - A \ C - A) \cdot \begin{pmatrix} t \\ r \\ s \end{pmatrix} = P - A$

$u = B - A$   
 $v = C - A$   
 $w = P - A$



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 40

$$\begin{pmatrix} t \\ r \\ s \end{pmatrix} = \frac{1}{\det(-\mathbf{d}, \mathbf{u}, \mathbf{v})} \cdot \begin{pmatrix} \det(\mathbf{w}, \mathbf{u}, \mathbf{v}) \\ \det(-\mathbf{d}, \mathbf{w}, \mathbf{v}) \\ \det(-\mathbf{d}, \mathbf{u}, \mathbf{w}) \end{pmatrix}$$

$$\det(\mathbf{a}, \mathbf{b}, \mathbf{c}) = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})$$

$$\begin{pmatrix} r \\ s \end{pmatrix} = \frac{1}{(\mathbf{d} \times \mathbf{v}) \cdot \mathbf{u}} \cdot \begin{pmatrix} (\mathbf{w} \times \mathbf{u}) \cdot \mathbf{v} \\ (\mathbf{d} \times \mathbf{v}) \cdot \mathbf{w} \\ (\mathbf{w} \times \mathbf{u}) \cdot \mathbf{d} \end{pmatrix}$$

- Kosten: 2 Kreuzprodukte + 4 Skalarprodukte
- Liefert: Geradenparameter + baryzentrische Koordinaten bzgl. Dreieck
- Test ob s,t im Bereich (0,1) und s+t <= 1

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 41

### Schnitt Strahl – Box

- Box (Quader) wird später noch wichtig als Bounding Box
- Hier: nur achsenparallele Boxes (AABB = *axis-aligned bounding box*)
- Definition einer AABB: durch die zwei extremen Eckpunkte  $(x_{\min}, y_{\min}, z_{\min})$  und  $(x_{\max}, y_{\max}, z_{\max})$
- Idee des Algo:
  - Eine Box ist der Schnitt von 3 *Slabs* (ein *Slab* = Schicht des Raumes, wird von 2 parallelen Ebenen begrenzt)
  - Jeder Slab schneidet vom Strahl ein Intervall heraus
  - Betrachte also sukzessive jeweils Paare von Box-Seiten

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 42

- Der Algorithmus:

```

setze t_min = -∞, t_max = +∞
loop über alle Paare von Ebenen:
  schneide Strahl mit den
    beiden Ebenen → t1, t2
  if t2 < t1:
    vertausche t1, t2
  // jetzt gilt: t1 < t2
  t_min ← max(t_min, t1)
  t_max ← min(t_max, t2)
// now: [t_min, t_max] = interval inside box
if t_min > t_max → kein Schnitt
if t_max < 0 → kein Schnitt
  
```

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 43

### Bemerkungen

- Optimierung: beide Ebenen eines Slabs haben dieselbe Normale
  - Spart ein Skalarprodukt
- Bemerkung: der Algo funktioniert genauso für "schiefe" Boxes (sog. *OBBs* = *oriented bounding boxes*)
- Weitere Optimierung: falls AABB, nutze aus, daß die Normalen nur 1 Komponente  $\neq 0$  haben!
- Achtung: teste auf Parallelität!
  - "shit happens"
  - Im Fall der AABB:
 

```

if |dx| < ε:
  if Px < x_min || Px > x_max:
    Strahl geht an Box vorbei
  else:
    t1, t2 = y_min, y_max // evtl noch swappen!
          
```

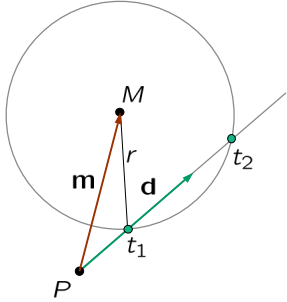
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 44

### Schnitt Strahl—Kugel

- Annahme:  $d$  normiert
- Ansatz:
 
$$|t \cdot d - m| = r$$

$$(t \cdot d - m)^2 = r^2$$

$$t^2 - 2t \cdot md + m^2 - r^2 = 0$$
- Es gibt noch andere Ansätze ...



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 45

- Der Algorithmus, mit kleinen Optimierungen:

```

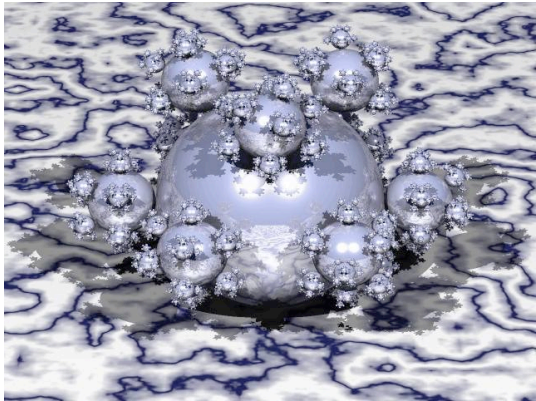
berechne  $m^2 - r^2$ 
berechne  $b = m \cdot d$ 
if  $m^2 - r^2 >= 0$  // Blickpunkt ausserhalb Kugel
  and  $b <= 0$  : // sieht von Kugel weg
then
  return "kein Schnittpunkt"
setze  $d = b^2 - m^2 + r^2$ 
if  $d < 0$ :
  return "kein Schnittpunkt"
if  $m^2 - r^2 > \epsilon$  :
  return  $t_1 = b - \sqrt{d}$  // enter;  $l_1 > 0$ 
else:
  return  $t_2 = b + \sqrt{d}$  // leave;  $l_2 > 0$ 
  
```

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 46

- Es ist so einfach, daß alle Ray-Tracer Kugeln haben!



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 47



Die sog. "sphere flake"

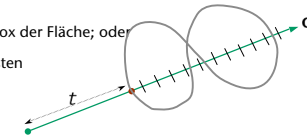
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 48

## Geometrisch vs. Algebraisch

- Die algebraische Methode ist einfach und generisch
- Die geometrische Methode ist schneller
  - Durch geometrische Einsicht
  - Frühe Tests
  - Insbesondere für die Strahlen, die weg zeigen

## Schnitt Strahl — implizite Fläche

- Implizite Fläche vom Grad  $n$ :  $F(x, y, z) = 0$
- Strahl:  $P(t) = O + t \cdot \mathbf{d}$
- Einsetzen:  $F(P(t)) = 0$   
liefert Polynom in  $t$  vom Grad  $n$
- Nullstellensuche:
  - Falls Grad  $< 5$ : nach  $t$  auflösen
  - Intervallschachtelung, Newton-Verfahren, ...
  - Startwerte:
    - Schnitt zwischen Strahl und BBox der Fläche; oder
    - Strahl innerhalb der BBox abtasten
- Z.B.: Kugel ...



## Nullstellensuche mit Laguerre's Methode

- Eine von wenigen "sure-fire"-Methoden
- Algorithmus erfordert Arithmetik mit komplexen Zahlen, auch wenn alle Wurzeln reell sind (und damit auch alle Koeffizienten)
- Sehr wenig theoretische Erkenntnisse zum Konvergenzverhalten
- Sehr viel empirische Hinweise, daß Algo (fast) **immer** zu einer Wurzel konvergiert, und zwar von (fast) **jedem** Startwert aus!
- Konvergenz-Ordnung 3, falls die Wurzel einfach ist

## Motivation für den Algorithmus

- Gegeben:

$$P(x) = (x - x_1)(x - x_2) \dots (x - x_n) \quad (0)$$

- Beziehungen:

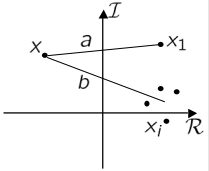
$$\ln |P(x)| = \ln |x - x_1| + \ln |x - x_2| + \dots + \ln |x - x_n|$$

$$\frac{d}{dx} \ln |P(x)| = \frac{1}{x - x_1} + \dots + \frac{1}{x - x_n} = \frac{P'(x)}{P(x)} =: G \quad (1)$$

$$\begin{aligned} \frac{d^2}{dx^2} \ln |P(x)| &= -\frac{1}{(x - x_1)^2} - \dots - \frac{1}{(x - x_n)^2} \\ &= \frac{P''(x)}{P(x)} - \left( \frac{P'(x)}{P(x)} \right)^2 =: -H \quad (2) \end{aligned}$$

- Sei  $x$  unsere aktuelle Näherung an Wurzel  $x_1$
- "Drastische" Annahme:
  - Abstand  $|x - x_1| = a$
  - Abstand zu allen anderen Wurzeln ist
 
$$|x - x_i| \approx b, \quad i = 2, 3, \dots, n$$
- Dann kann man (1) & (2) so darstellen
 
$$G \approx \frac{1}{a} + \frac{n-1}{b} \quad (3)$$

$$H \approx \frac{1}{a^2} + \frac{n-1}{b^2} \quad (4)$$



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 53

- Einsetzen ergibt Lösung für  $a$ :
 
$$a \approx \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}} \quad (5)$$
- VZ so wählen, daß  $|a|$  minimal wird
- Wurzel kann negativ werden
  - $\rightarrow a$  kann komplex werden
- Neue Näherung für  $x_1$  ist
 
$$x_1 = x - a$$

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 54

### Algorithmus

```

wähle  $x_0$ 
loop:
  berechne  $G = \frac{P'(x_i)}{P(x_i)}$ 
            $H = G^2 - \frac{P''(x_i)}{P(x_i)}$ 
  berechne  $a = \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}}$ 
  setze  $x_{i+1} = x_i - a$ 
until  $a$  "klein genug" oder  $k \geq \max$ 
    
```


- Achtung: möglichst Code aus *Numerical Recipes* verwenden
  - Selbst implementieren ist fehlerträchtig
  - NR-Code hat elegantes Abbruchkriterium
- Für Ray-Tracing: alle Nullstellen berechnen
  - faktoriere gefundene Nullstelle aus, wiederhole Laguerre  $n$  Mal

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 55

### Instancing / Strahltransformation

- Kompliziertere (transformierte) Formen lassen sich oft reduzieren auf einfache (kanonische) Formen
- Idee: transformiere Strahl; führe Schnitt mit der einfachen Form durch; transformiere Schnittpunkt und -normale zurück
- Beispiel Ellipsoid:
 

berechne  $P'(t) = M^{-1}O + tM^{-1}d$   
 schneide  $P'(t)$  mit Einheitskugel  $\rightarrow P', n', t'$   
 $P := M \cdot P'; n := (M^{-1})^T \cdot n'; t := ?$



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 56

- Falls M keine Skalierung enthält, dann ist
 
$$\|d'\| = \|M^{-1}d\| = \|d\| \Rightarrow t = t'$$
- Falls M eine Skalierung enthält:
  - $\|d'\| \neq \|d\|$
  - Normiere  $d$  nicht!
  - Damit ist  $t = t'$
  - Einzigste Bedingung: in den Schnittberechnungen darf man nirgendwo die Annahme  $\|d'\| = 1$  machen.

World Space

Object Space

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 57

### Weiterer Grund für Instancing

- Speichereinsparung: nur mittels Instancing passen solch riesige Szenen komplett in den Speicher

61 unique plant models, 1.1M unique triangles, 300MB – 4000 plants in the scene, 19.5M triangles

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 58

### Weitere Quadriken

- Mit diesen Techniken kann man viele weitere Objekte testen
  - Parametrische Liniengleichung in implizite Form des Objektes einsetzen
  - Ergibt Polynom in  $t$  vom Grad 2, 4, ...
  - Nach  $t$  lösen (analytisch, Newton, ...)
- Unendlicher Zylinder:
 
$$x^2 + y^2 = 1$$
- Paraboloid:
 
$$x^2 + y^2 - z = 0$$
- Hyperboloid (one sheet):
 
$$x^2 + y^2 - z^2 = 1$$

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 59

### Torus:

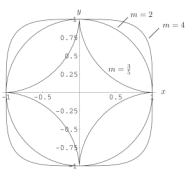
$$(c - \sqrt{x^2 + y^2})^2 + z^2 = a^2$$

	full view	cutaway	cross-section
ring torus			
horn torus			
spindle torus			

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 60

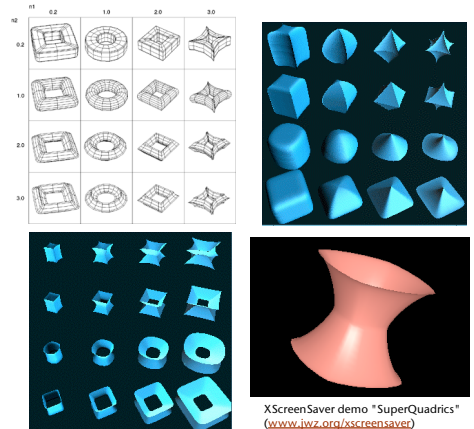
### Superquadrics

- Verallgemeinerungen der Quadriken
- Super-Ellipsoid:
 
$$\left(\frac{x}{a}\right)^p + \left(\frac{y}{b}\right)^q + \left(\frac{z}{c}\right)^r = 1$$
- Super-Hyperboloid:
 
$$\left(\frac{x}{a}\right)^p + \left(\frac{y}{b}\right)^q - \left(\frac{z}{c}\right)^r = 1$$
- Super-Toroid:
 
$$\left(d - \left(\left(\frac{x}{a}\right)^m + \left(\frac{y}{b}\right)^n\right)^q\right)^r + \left(\frac{z}{c}\right)^p = e^2$$
- Achtung: hier ist immer  $|x|^p$  gemeint!



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 61

### Beispiele von Super-Quadrics



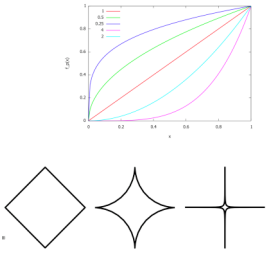
XScreenSaver demo "SuperQuadrics" ([www.lwz.org/xscreensaver](http://www.lwz.org/xscreensaver))

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 62

### Ratioquadrics [Blanc & Schlick, 1996]


- Variante der Superquadrics mit u.U. besseren Eigenschaften
- Idee der Superquadrics kann man auch so schreiben:
 
$$F(x, y, z) = f_p\left(\frac{x}{a}\right) + f_q\left(\frac{y}{b}\right) + f_r\left(\frac{z}{c}\right) - 1$$

$$f_p(x) = |x|^p$$
- Problem:
  - $f_p(x)$  ist an der Stelle  $x=0$  nicht differenzierbar für  $p \leq 1$
  - Dadurch entstehen für  $p < 1$  "Spitzen", die möglicherweise unerwünscht sind
  - Außerdem ist  $f_p(x)$  rel. teuer auszuwerten



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 63

- Simple Idee: verwende einfach eine andere "Potenzfunktion"
- Die Pseudo-Potenzfunktion von Blanc & Schlick:
 
$$g_p(x) = \frac{x}{p + (1-p)x}$$
- Die Ratioquadric für "Ratio-Ellipsoide" ist damit
 
$$F(x, y, z) = g_p\left(\frac{x}{a}\right) + g_q\left(\frac{y}{b}\right) + g_r\left(\frac{z}{c}\right) - 1$$
- Resultat:



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing 64