



CAUSTICS & GODRAYS



Jacob Skuratovski

Kaustiken

- Kaustik, caustic (engl.), Brennlinie, Brennfläche
- Bereich, in dem Lichtstrahlen gebündelt werden
- 3 verschiedene Arten
 - Katakaustik
 - Diakaustik
 - Regenbogen

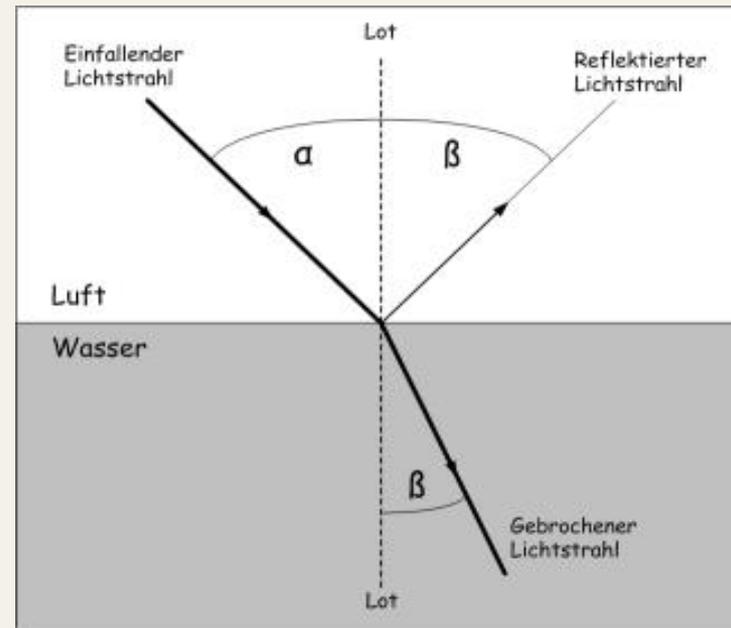


Eine Katakaustik

http://de.wikipedia.org/wiki/Kaustik_%28Optik%29

Diakaustik

- Ablenkung von Lichtstrahlen durch ein brechendes Medium
 - ▣ Linse
 - ▣ Wasser

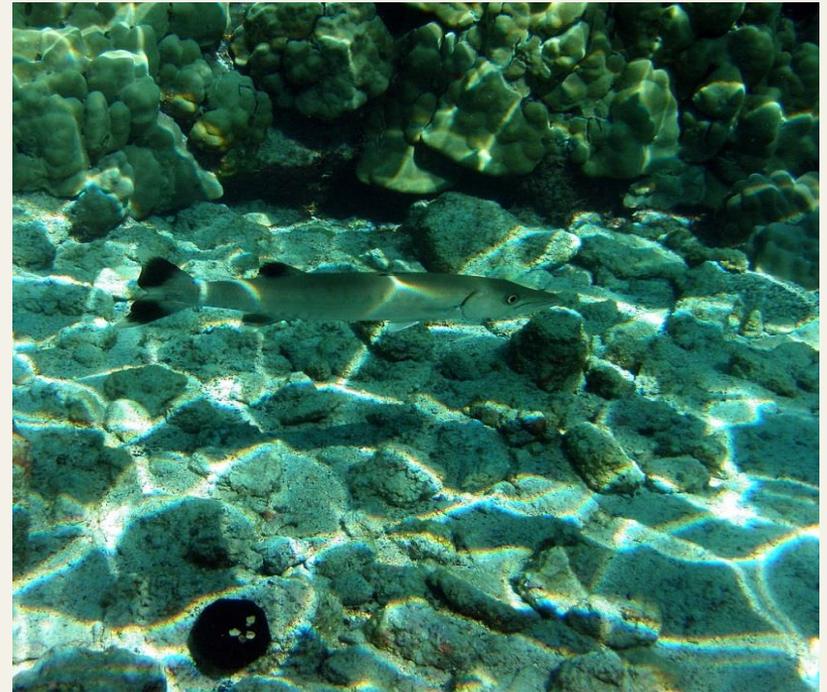


Lichtbrechung

<https://macsclassroom53.files.wordpress.com/2008/06/lichtbrechung.jpg>

Diakaustik im Wasser

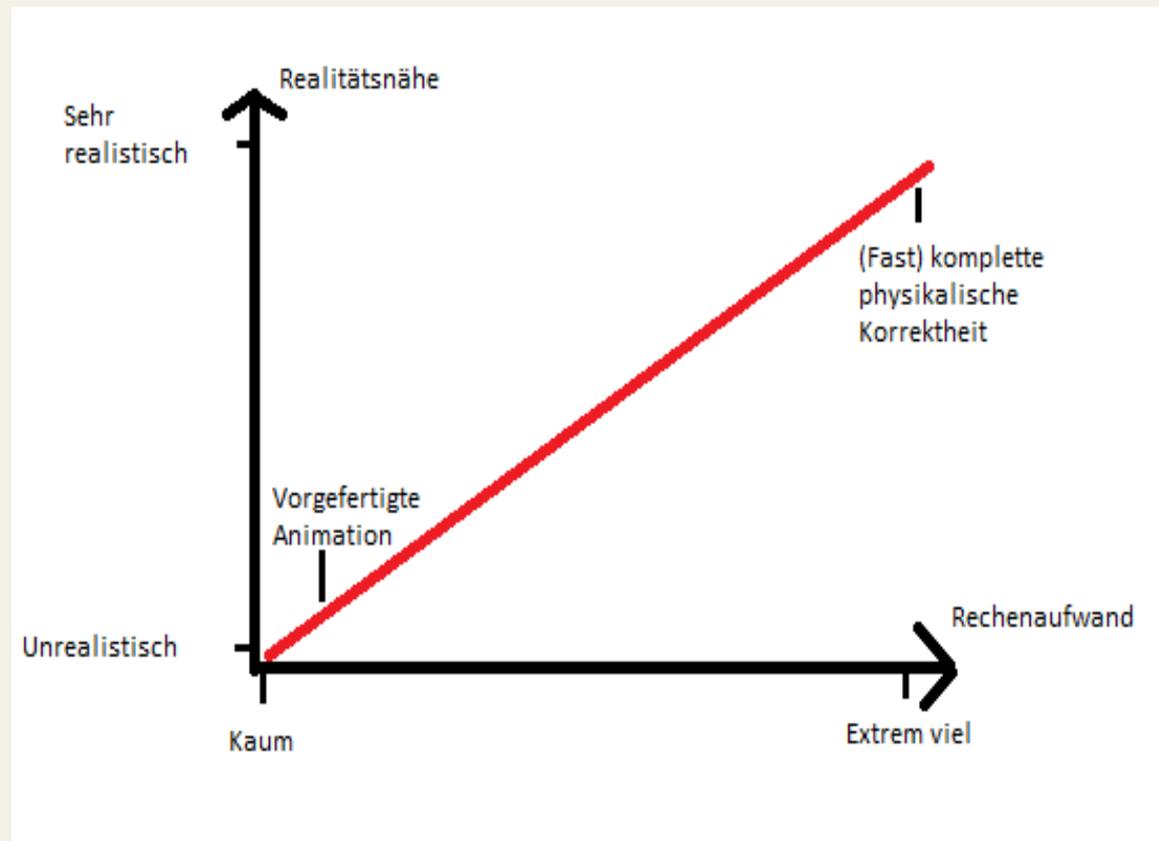
- Lichtbrechung durch Wasseroberfläche
- Nicht ruhige/glatte Oberfläche
 - Verschiedene Brechungen
 - Verschiedene Lichtkonzentrationen



Eine Diakaustik

[http://en.wikipedia.org/wiki/Caustic_\(optics\)](http://en.wikipedia.org/wiki/Caustic_(optics))

Implementierung von Diakaustiken



Paint skills

Diakaustik: Mathematik

□ Brechung von Licht muss berechnet werden

□ Snelliussches Brechungsgesetz

□ $IOR = \frac{\eta_1}{\eta_2} = \frac{\sin \theta_2}{\sin \theta_1}$ $n_1 n_2$ Brechungsindizes

□ So nicht zu implementieren

□ Umformung:

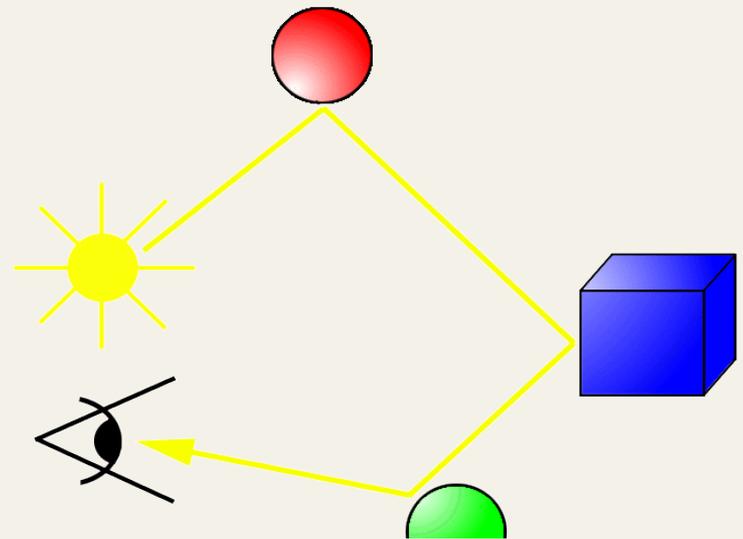
■ $T = N \left(\frac{\eta_1}{\eta_2} (E \cdot N) \pm \sqrt{1 + \left(\frac{\eta_1}{\eta_2} \right)^2 \left((E \cdot N)^2 - 1 \right)} \right) + \frac{\eta_1}{\eta_2} E.$

T = gebrochener Strahl
N = Oberflächennormale
E = Einfallender Strahl

■ Foley et al. 1996

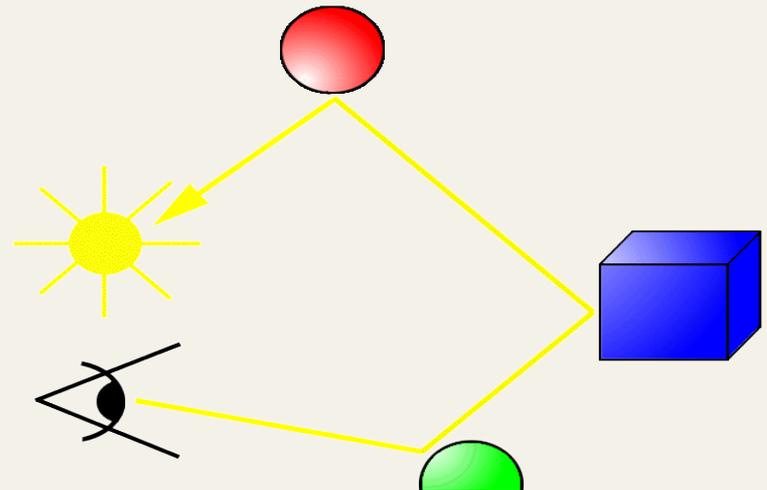
Ray Tracing

- Forward Ray Tracing
 - ▣ Verfolgen von Lichtstrahlen vom Ursprung zum Ziel
 - ▣ Nachteil: Sehr viele Lichtstrahlen landen nicht bei **unserem** Ziel



Backward Ray Tracing

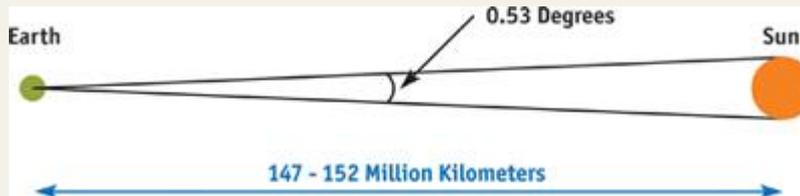
- Verfolgung vom Ziel zum Ursprung
- Von jedem Punkt aus hemisphärisches Integral berechnen
- Sehr viele Strahlen fallen von Anfang an weg



Monte-Carlo-Methode

- Zwei aggressive Annahmen

- Mittagszeit am Äquator



http://http.developer.nvidia.com/GPUGems/gpugems_ch02.html

- Nur die Strahlen, die vertikal von der Wasseroberfläche den Boden erreichen

- Diese sind die hellsten

- Realitätsverlust!

Shader

- „Shader sind mathematische Algorithmen, welche die Aufgabe haben, bestimmte Effekte beim Berechnen einer dreidimensionalen Computergrafik zu erzeugen.“

Aus: <http://www2.hs-fulda.de/caelabor/inhalte/OpenGL/Projekte/roesler/shaderprog.pdf>

- Programme

Arten von Shadern

- Geometry-Shader
 - Erzeugen neuer primitiver Geometrien aus schon vorhandenen Primitiven
- Vertex-Shader
 - Geometrische Berechnungen durch Transformation von Vertices (Eckpunkte dreidimensionaler Objekte)
- Pixel/Fragment-Shader
 - Berechnen von Farbe/Textur von Bildpunkten

Sprachen

- GLSL (OpenGL)
- CG (Nvidia)
- HLSL (DirectX)
- ...

Realisierung von Kaustiken

- Auf den Meeresboden weiteren mesh legen
- Pro Vertex 1 Lichtstrahl horizontal nach oben
- Brechung mit snellusischem Brechungsgesetz
- Winkel zwischen gebrochenen Strahlen und Vertikale berechnen
- Je kleiner dieser Winkel ist, desto heller den Ursprungsvertex färben

Godrays

- „Strahlenbüschel oder Lichtbüschel sind ein Phänomen der atmosphärischen Optik, das durch die Verdeckung der Sonne durch atmosphärische Trübungen hervorgerufen wird. „

Aus: <http://de.wikipedia.org/wiki/Strahlenb%C3%BCschel>



<http://www.donsmithblog.com/wp-content/uploads/2012/03/god-beams-san-jose-blog.jpg>



<http://de.wikipedia.org/wiki/Strahlenb%C3%BCschel>

Entstehung

- Kein Vakuum auf der Erde
- Licht wird von Teilchen absorbiert, reflektiert und gestreut
- Dadurch entsteht der Effekt, dass man Licht als Strahl sieht

Implementierung

$$L(s, \theta, \phi) = \text{exposure} \times \sum_{i=0}^n \text{decay}^i \times \text{weight} \times \frac{L(s_i, \theta_i)}{n},$$

$$L(s, \theta) = L_0 e^{-\beta_{\text{ex}} s} + \frac{1}{\beta_{\text{ex}}} E_{\text{sun}} \beta_{\text{sc}}(\theta) (1 - e^{-\beta_{\text{ex}} s}),$$

Verlust von
Helligkeit

Hinzugekommene
Helligkeit

Implementierung als post-processing
pixel-shader in <15 Zeilen

L = Helligkeit in einem Pixel

Θ (Theta) = Winkel zwischen
Strahl und Sonne

S = Weg, den das Licht durch
das Medium zurücklegt

E_{sun} = Leuchtkraft der Sonne

β_{ex} = Verlustkonstante,
bestehend aus Streuungs- und
Absorptionskomponenten

β_{sc} = Streuungskonstante

Φ (Phi) = Position des Pixels

N = Anzahl der Proben

Decay = Wichtigkeit der Probe
Exposure = Intensität des post-
processing

Weight = Intensität eines jeden
Samples



Danke!

Quellen (Auswahl)

- <http://sirkan.iit.bme.hu/~szirmay/gpucaust5.pdf>
- http://de.wikipedia.org/wiki/Kaustik_%28Optik%29
- http://help.solidworks.com/2012/German/SolidWorks/sldworks/c_caustics.htm?format=P
- https://books.google.de/books?id=LGDYuxw74vgC&pg=PA118&lpg=PA118&dq=kaustiken&source=bl&ots=fTgU66SzlU&sig=KzjQ_nTA4DYXFUWGEQE1zTFj-3E&hl=de&sa=X&ei=I083VdnbOMTLalCWgdgl&ved=0CCgQ6AEwAjkK#v=onepage&q=kaustiken&f=false
- <http://www2.hs-fulda.de/caelabor/inhalte/OpenGL/Projekte/roesler/shaderprog.pdf>
- <http://de.wikipedia.org/wiki/Shader>
- http://de.wikipedia.org/wiki/Snelliusches_Brechungsgesetz
- http://en.wikipedia.org/wiki/Normal_%28geometry%29#Calculating_a_surface_normal
- http://http.developer.nvidia.com/GPUGems/gpugems_ch02.html
- http://http.developer.nvidia.com/GPUGems3/gpugems3_ch13.html
- <http://amd-dev.wpengine.netdna-cdn.com/wordpress/media/2012/10/ATI-LightScattering.pdf>
- <http://fabiansanglard.net/lightScattering/>
- <http://de.wikipedia.org/wiki/Tyndall-Effekt>
- <http://de.wikipedia.org/wiki/Strahlenb%C3%BCschel>
- [...]