



Computer-Graphik II

Beschleunigung des Ray-Tracing

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

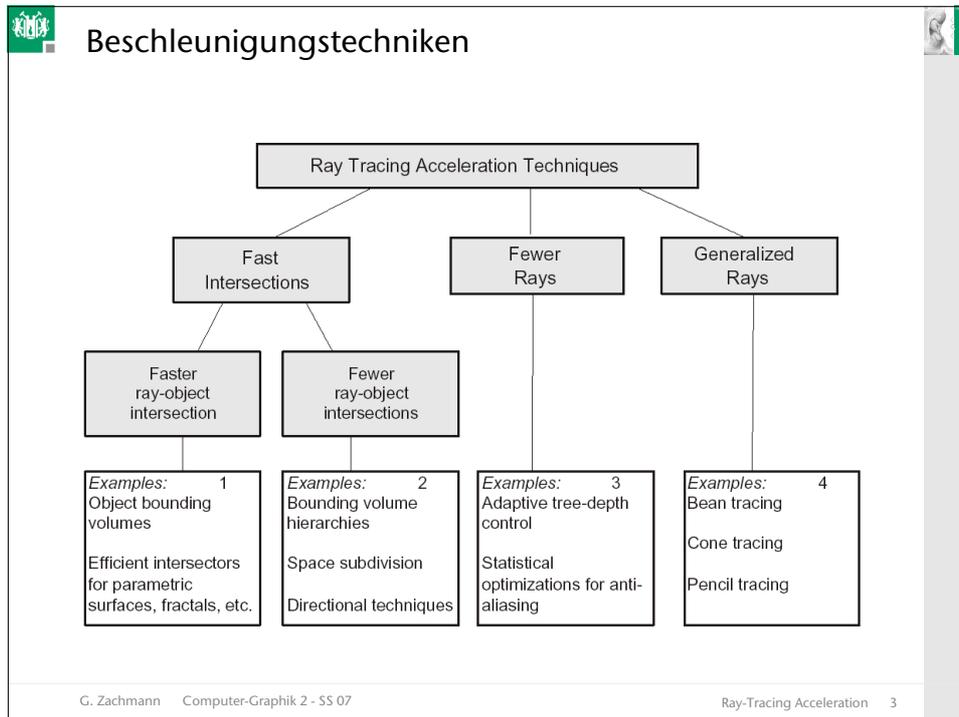


Kosten des Ray-Tracing

cost \approx height * width *
num primitives *
intersection cost *
size of recursive ray tree *
num shadow rays *
num supersamples *
num glossy rays *
num temporal samples *
num focal samples *
...

Kann man das verringern?

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



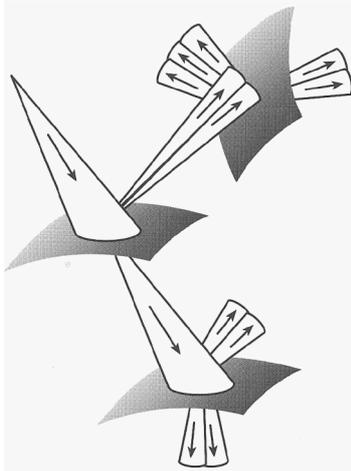
Der *Light Buffer*

- Beobachtung: bei der Verfolgung von Schattenstrahlen reicht es, **irgendeinen** Schnittpunkt mit einem opaken Objekt zu finden
- Idee: speichere bei jeder Lichtquelle und für jede Raumrichtung eine Liste von Polygonen, die in dieser Richtung liegen
 - Datenstruktur des **Light Buffer**: "**Richtungswürfel**"
 - Entweder als Preprocessing (scan conversion auf die Würfelseiten), oder "on demand" (eintragen in Zelle falls Occluder gefunden)

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

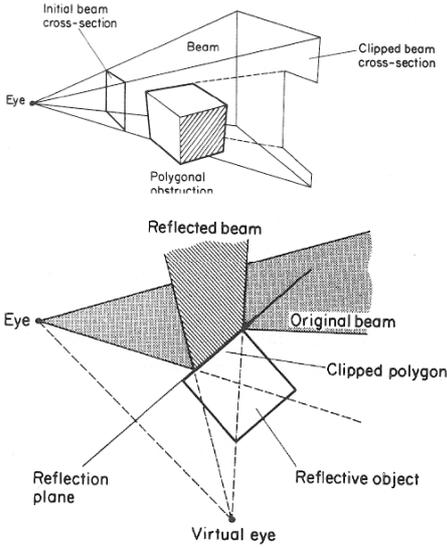
Beam und Cone Tracing

- Allgemeine Idee: versuche Beschleunigung durch Verschießen mehrerer oder "dickerer" Strahlen auf einmal
- Beam Tracing:
 - Ein Strahlbündel mit Pyramide genau darstellen
 - Neue Beams an den Oberflächen (Polygone) erzeugen
- Cone Tracing:
 - Ungefähre Approximation eines Strahlbündels mit Kegeln mit Kegeln
 - Wenn notwendig, in kleinere Kegel unterteilen
- Probleme:
 - Ausschnitt der Strahlen?
 - Gute Approximation?
 - Wie berechnet man Schnitte mit Flächen?
- Nicht wirklich praktikabel, viel zu teuer!



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 5

Beam Tracing

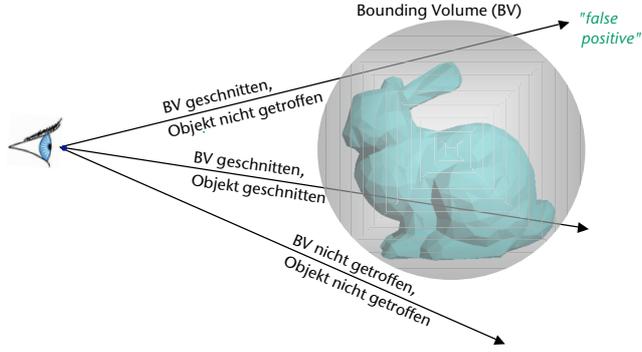


Initial beam cross-section
Eye
Beam
Clipped beam cross-section
Polygonal object
Reflected beam
Original beam
Clipped polygon
Reflection plane
Reflective object
Virtual eye

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

Bounding Volumes (BVs)

- Grundidee: spare Kosten durch Vorberechnungen mit der Szene und Filterung der Strahlen zur Laufzeit

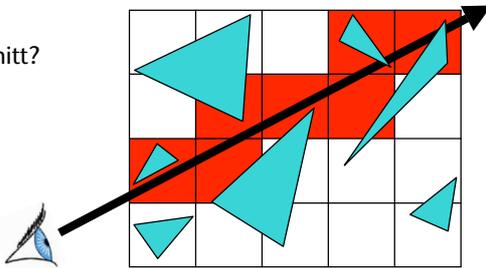


- Verfehlt der Strahl das Bounding Volume, so kann man auf den Schnitt mit der Teilszene verzichten

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

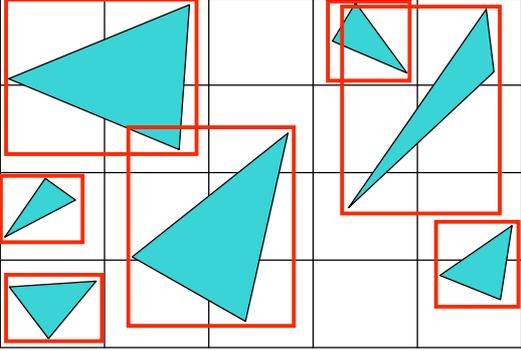
Regelmäßiges Gitter

- Erstellung des Gitters:
 - Bestimme BBox der Szene
 - Bestimme gute Gitter- Auflösung (n_x, n_y, n_z)
- Für jede Zelle entlang eines Strahls:
 - Enthält die Zelle einen Schnitt?
 - Ja: liefere Schnitt zurück
 - Nein: fortfahren



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 8

- Primitive in Gitter einfügen:
 - Benutze Objekt-BBox
 - I.a. mehrfache Einfügung in versch. Zellen
- Jede Zelle enthält Liste mit Zeigern auf Objekte

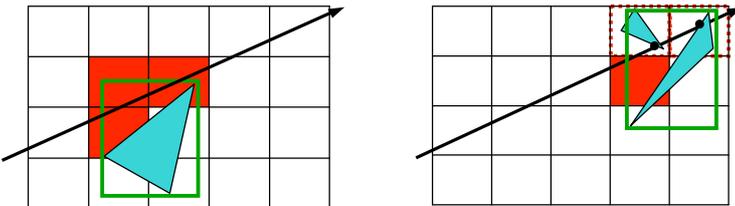


G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 9

Die Mailbox-Technik

- Nach der Unterteilung des 3D-Raums können die Objekte in mehreren Voxeln liegen und müssen dann in jedem von diesen Voxeln referenziert werden

1. Problem: Schnitt muß nicht der nächste sein (r.u.)
 - Lösung: wenn Schnittparameter t nicht im Innen der Zellestrecke ist, dann weitermachen (es kann etwas näheres geben)
2. Problem: wie vermeidet man, dass der Strahl 3x gegen das Obj getestet wird? (l.u.)



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 10




- Lösung: jedem Objekt in der Szene wird eine **Mailbox** und jedem Strahl eine eindeutige **Strahl-ID** zugeordnet
 - Einfach im Konstruktor der Strahl-Klasse einen Zähler hochzählen
- Nach jedem Schnittpunkttest wird die Strahl-ID in die Mailbox des Objekts gespeichert
- Vor jedem neuen Schnittpunkttest wird die Strahl-ID des aktuellen Strahls mit der Strahl-ID in der Mailbox des Objektes verglichen:
 - die IDs sind gleich → das Ergebnis des Schnittpunkttests kann ohne weitere Berechnungen aus der Mailbox ausgelesen werden;
 - sonst → führe neue Schnittpunktberechnung durch und speichere das Ergebnis in der Mailbox (mit Strahl-ID)

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



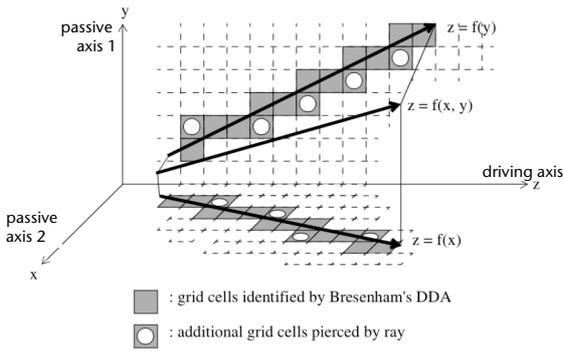

Optimierungen der Mailbox-Technik

- Probleme der naiven Methode:
 - Schreiben der Mailbox im Dreieck zerstört Cache
 - Man kann nicht mehrere Strahlen parallel testen
- Lösung: speichere Mailbox getrennt von den Dreiecksdaten
 - Kleine Hash-Table zu jedem Strahl, die die Dreiecks-IDs enthält
 - Nur wenige Dreiecke werden von jedem Strahl berührt
 - Hashtable kann hauptsächlich im Level-1-Cache bleiben
 - Einfache Hashing-Funktion reicht
 - Paralleles Testen mehrere Strahlen auf versch Prozessoren trivial
- Dahinter steckt das alte Problem: soll man
 "Array of Structs" (AoS) oder *"Struct of Arrays"* (SoA)
 implementieren?

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

Traversierung eines 3D-Gitters

- Einfache Idee: verwende 2 synchronisierte DDA's → 3D-DDA
 - Wie im 2D gibt es eine "driving axis"
 - Im 3D gibt es aber **zwei** "passive axes"

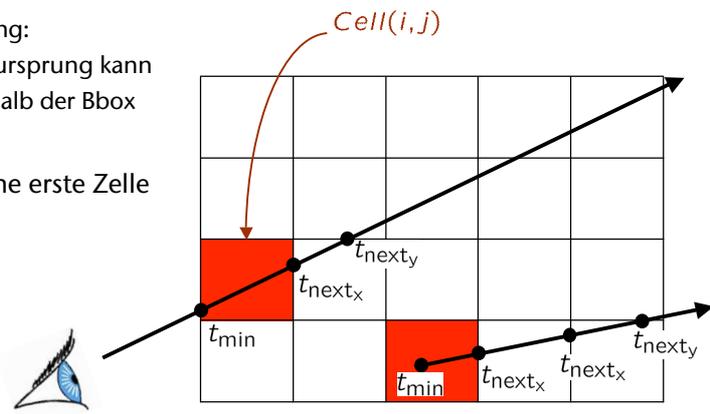


: grid cells identified by Bresenham's DDA
 : additional grid cells pierced by ray

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

Besserer Gitter-Traversierungs-Algorithmus

- Schneide Strahl mit Bbox der Szene
 - Achtung: Strahlursprung kann innerhalb der Bbox sein!
- Bestimme erste Zelle



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

- Gibt es ein Muster für die Zellenübergänge?
- Ja, horizontale und vertikale Übergänge haben regelmäßigen Abstand

$dt_x = g_x / d_x$

$dt_y = g_y / d_y$

g_x

g_y

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

Der Algorithmus

```

if tnext_x < tnext_y :
    i += sx
    tmin = tnext_x
    tnext_x += dtx
else:
    j += sy
    tmin = tnext_y
    tnext_y += dty

```

(d_x, d_y)

dt_y

dt_x

$$s_x = \begin{cases} 1 & , d_x > 0 \\ -1 & , d_x \leq 0 \end{cases}$$

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

Speicherung

- Viele leere Zellen → stelle Gitter durch eine Hashtabelle dar

Hash-Funktion $h(i,j,k)$

Hash-Tabelle

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 17

- Dicht besetztes Gitter → verwende Blocking (aka "memory bricking")
 - Teile Gitter auf in Blocks, speichere jeden Block in zusammenhängenden Speicherbereich, so daß 1 Block = 1 L1-Cache-Zeile
 - Fasse Blocks zu "Macro-Blocks" zusammen, so daß 1 Macro-Block komplett in den L2-Cache passt

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 18

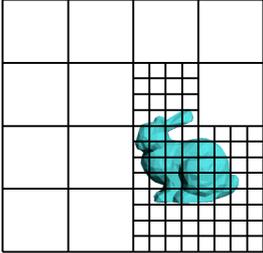
Optimale Zahl der Voxel

- Zu viele Zellen → langsame Traversierung, großer Speicherverbrauch (schlechte Cache-Ausnutzung)
- Zu wenig Zellen → zu viele Primitive in einem Zellen
- Gute Daumenregel: Seitenlänge der Zellen so groß wie die durchschnittliche Seitenlänge der Dreiecke (Objekte)
- Kennt man die nicht (oder ist zu teuer zu berechnen): wähle Seitenlänge = $\sqrt[3]{N}$
- Weitere Daumenregel: möglichst würfelförmige Voxel erzeugen

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

Rekursives Gitter [1989]

- Problem: reguläres Gitter passt sich nicht gut unterschiedlichen lokalen Dichten an ("teapot in a stadium")
- Idee:
 - Erzeuge zunächst nur grobes Gitter
 - Unterteile "dichte" Zellen wieder durch ein (grobes) Gitter
 - Abbruchkriterium: weniger als n Objekte in Zelle oder max. Tiefe erreicht
- Ergibt n^3 -Wege-Baum
 - Evtl. Problem der effizienten Speicherung
- Zusätzliches Feature: Unterteilung "on demand"
 - Erzeuge zunächst nur 1-2 Levels
 - Falls Strahl zur Laufzeit Zelle trifft, die Abbruchkriterium nicht erfüllt, erzeuge dann weitere Levels

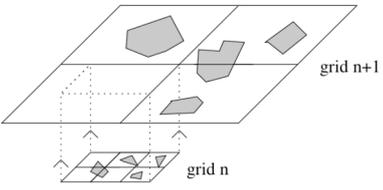


Nested Grids

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

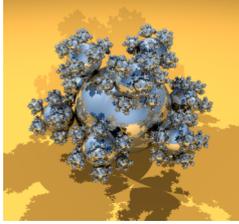
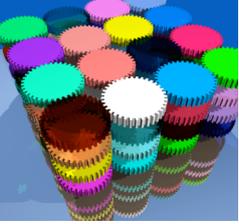
Hierarchical Uniform Grid [1994]

- Problem: Anpassung der Zellengröße an die Objektgröße, wenn viele unterschiedliche Größen dabei sind
- Idee:
 - Gruppieren Objekte nach Größe → Cluster
 - Gruppieren Objekte innerhalb jedes Clusters nach Entfernung → kleinere Cluster
 - Baue Gitter für jedes dieser Cluster
 - Konstruiere Hierarchie über diese elementaren Gitter
- Beispiel:



G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 21

Vergleich einiger hierarchischer Gitter (Aufbau)

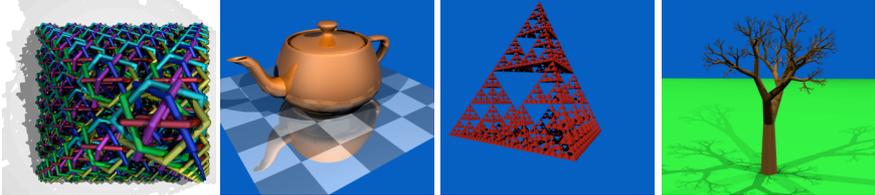




	balls	gears	mount
Uniform - D = 1.0	0.19	0.38	0.26
Uniform - D = 20.0	0.39	1.13	0.4
Rekursiv	0.39	5.06	1.98
HUG	0.4	1.04	0.16

$$D = \frac{\text{Anzahl Voxel}}{\text{Anzahl Objekte}}$$

Quelle: Vlastimil Havran, Ray Tracing News vol. 12 no. 1, June 1999, <http://www.acm.org/tog/resources/RTNews/html>

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 22



	rings	teapot	tetra	tree
Uniform - $D = 1.0$	0.35	0.3	0.13	0.22
Uniform - $D = 20.0$	0.98	0.65	0.34	0.33
Rekursiv	0.39	1.55	0.47	0.28
HUG	0.45	0.53	0.24	0.48

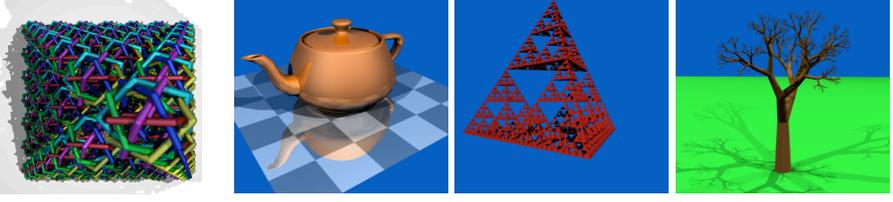
G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 23

Laufzeit



	Kugeln	Getriebe	Fassung
Uniform - $D = 1.0$	244.7	201.0	28.99
Uniform - $D = 20.0$	38.52	192.3	25.15
Rekursiv	36.73	214.9	30.28
HUG	34.0	242.1	62.31

G. Zachmann Computer-Graphik 2 - SS 07 Ray-Tracing Acceleration 24



	Ringe	Teekanne	Tetra	Baum
Uniform - D = 1.0	129.8	28.68	5.54	1517.0
Uniform - D = 20.0	83.7	18.6	3.86	781.3
Rekursiv	113.9	22.67	7.23	33.91
HUG	116.3	25.61	7.22	33.48
Anpassungsfähig	167.7	43.04	8.71	18.38

G. Zachmann Computer-Graphik 2 - SS 07

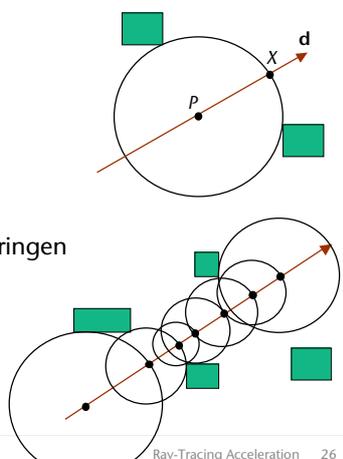
Ray-Tracing Acceleration 25

Proximity Clouds

[1994]

- Gedankenexperiment:
 - Annahme: wir stehen auf dem Strahl an Punkt P, und wissen, daß sich in einer Kugel um P mit Radius r kein Objekt befindet
 - Dann können wir direkt zum Punkt

$$X = P + \frac{r}{\|d\|} \mathbf{d}$$
 weiterspringen
 - Annahme: diesen "clearance" Radius wissen wir in jedem Pukt des Raumes
 - Dann kann man von Punkt zu Punkt springen

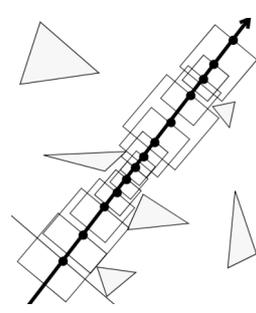


G. Zachmann Computer-Graphik 2 - SS 07

Ray-Tracing Acceleration 26

- Das funktioniert genau so mit jeder anderen Metrik
- Problem: man kann nicht in **jedem** Punkt diese Info speichern
- Idee: diskretisiere
 - Speichere für jede Gitterzelle, wieviele (leere) Zellen man überspringen kann, unabhängig von der Strahlrichtung
- Beispiel:

	1	1	1	1					
	2	2	2	2					
	3	3	3	3					
	4	4	4	3					
		3	3						
			1	1	1				



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