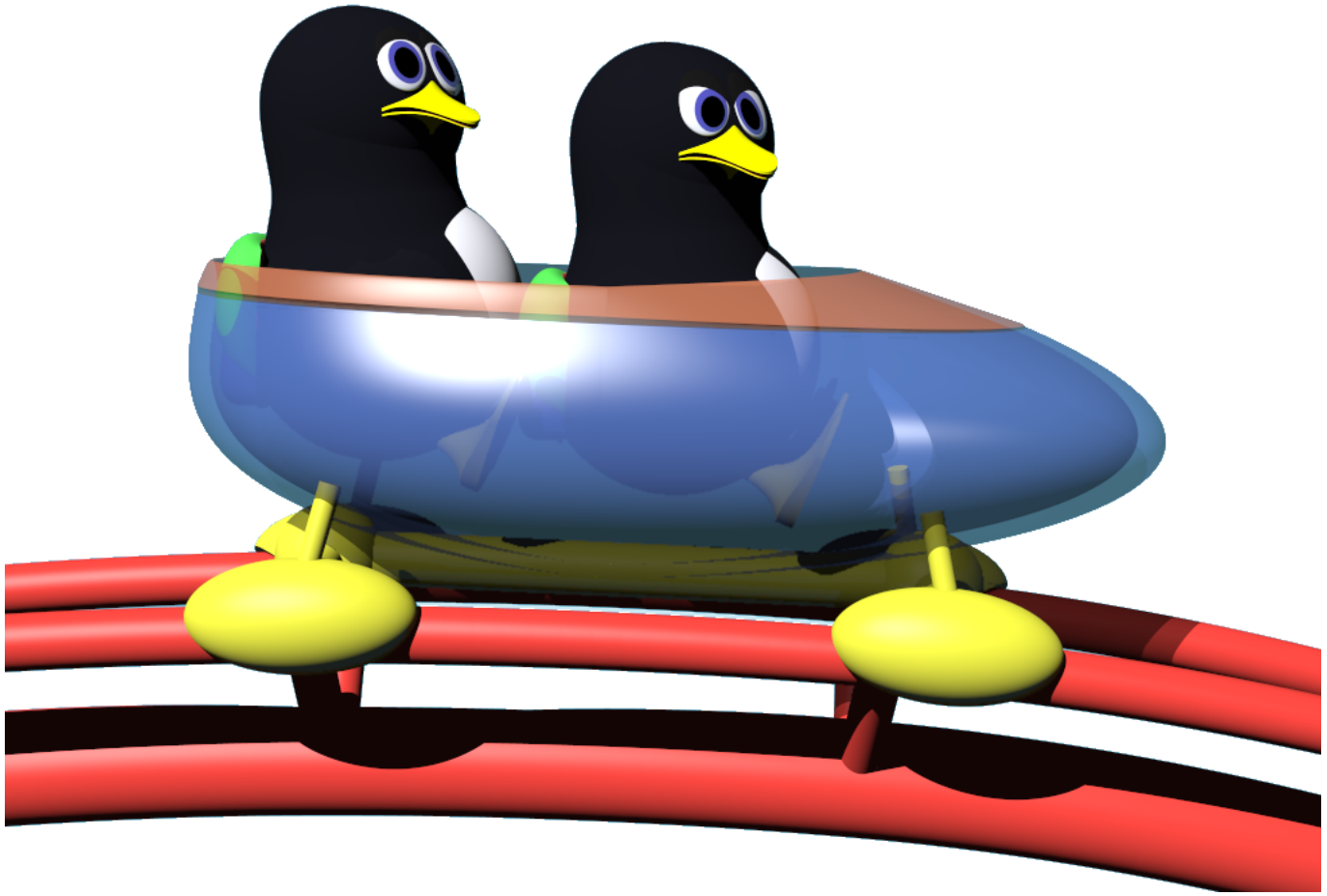
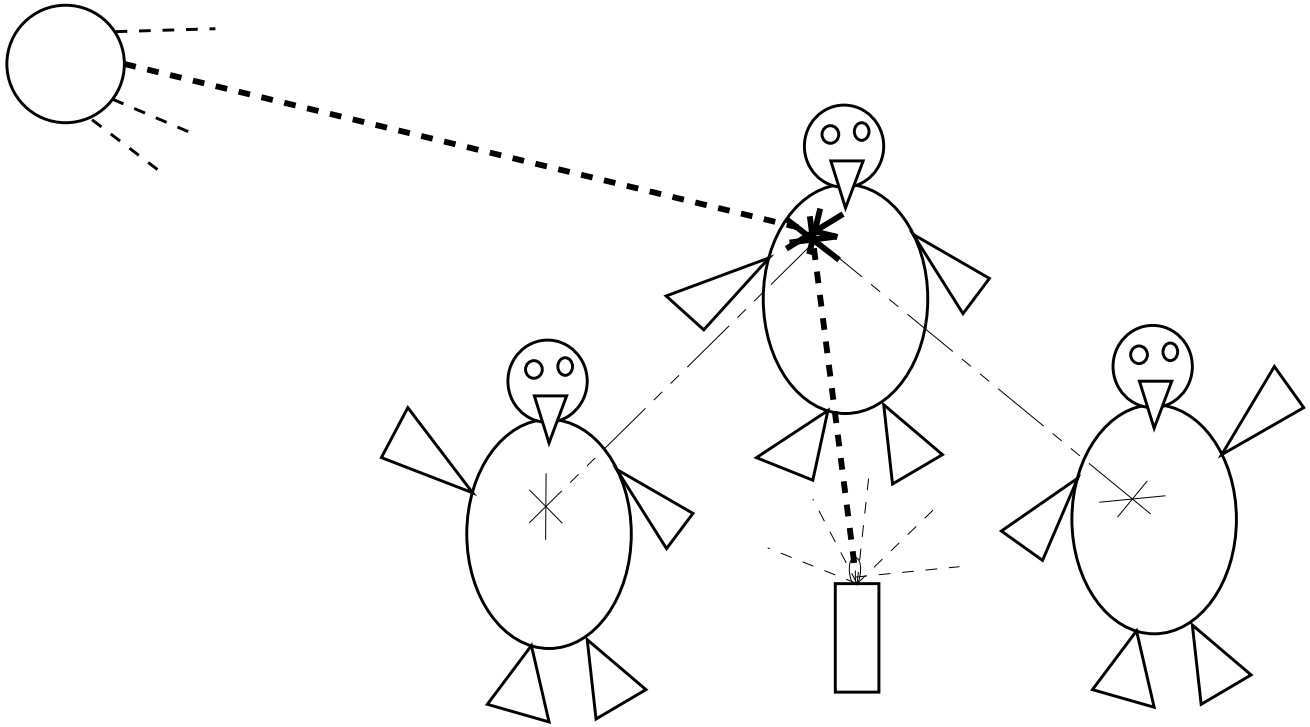
Aus Zahlen werden Bilder

Jan Tobias Mühlberg
<muehlber@fh-brandenburg.de>

Quelle: http://www.emperor-penguin.com



Modellierung einer Realität



Ein endlich genaues Modell der realen Welt...

Modellierung einer Realität

- Beschreibung der einzelnen Objekte einer Szene
- Zuordnung physikalischer Material-eigenschaften
- Definition von Lichtquellen
- Festlegung einer Perspektive, aus der die Realität betrachtet werden soll

Modellierung einer Realität

- Modellierungs-Software erlaubt die Verwendung einfacher geometrischer Formen:
  - Ebenen und Flächen
  - Kisten, Kegel, Zylinder und Kugeln
  - häufig auch „Doughnuts“ und Schnittkörper
- zusetzlich gibt es Lichtquellen und Kameras

Modellierung einer Realität

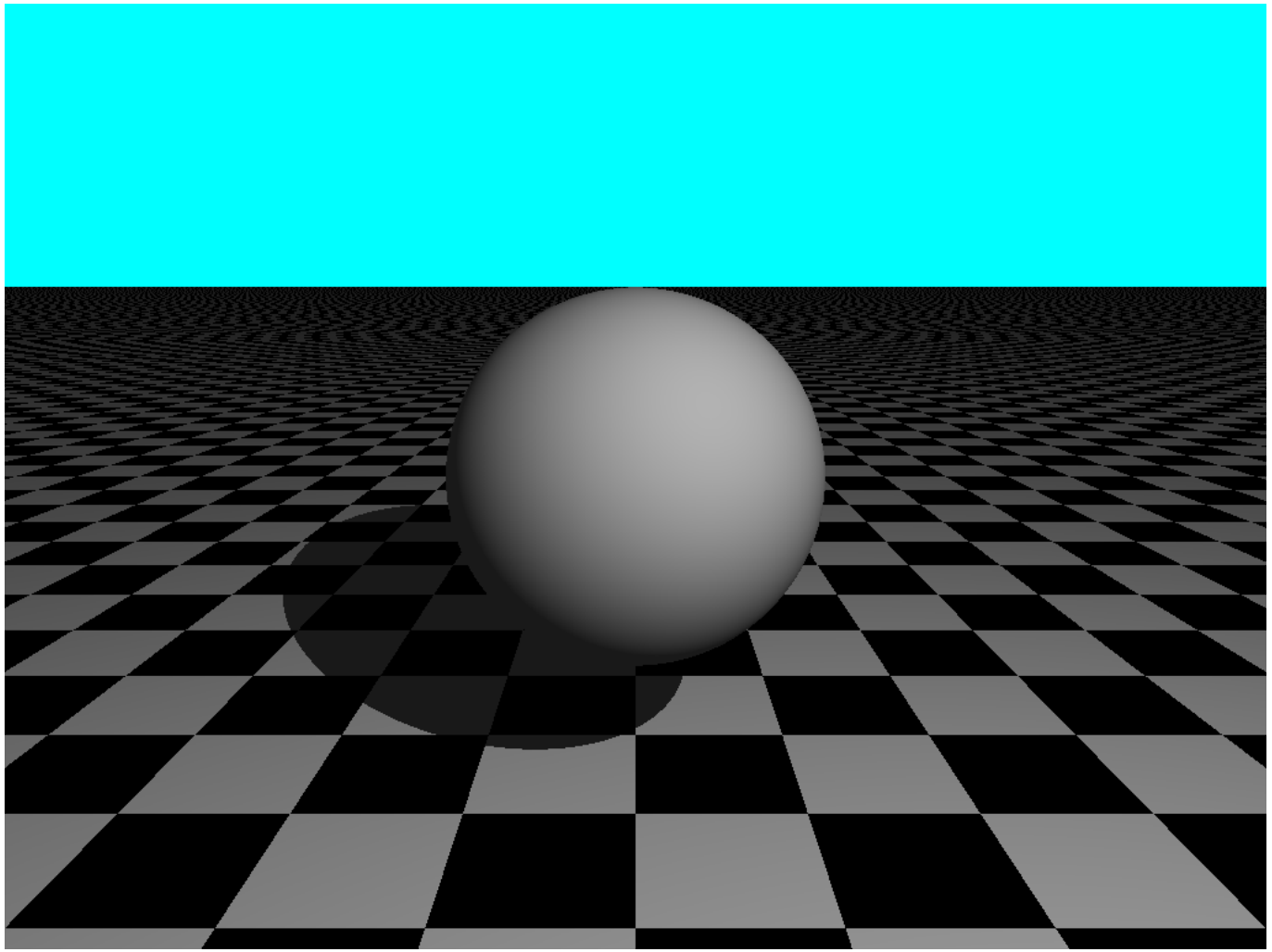
- „mathematische“ Repräsentation unserer Realität innerhalb des Computers
- Objekte einer Szene werden in Form von Vektoren oder einfachen Formeln abgelegt
- ganz toll in PovRAY: der Anwender kann Objektdefinitionen direkt in Form von Vektoren und zusätzlichen Attributen eingeben

Ein Beispiel mit PovRAY...

```
#include "colors.inc"

background { color Cyan }

camera {
 location <0, 2, -3>
 look_at  <0, 1,  2> }

light_source { <2, 4, -3> color White}
```

Ein Beispiel. . . (cont'd)

```
sphere {
 <0, 1, 2>, 1
 texture { pigment { color White } } }
plane { <0, 1, 0>, -1
 pigment { checker color White, color Black } }
```



Raytracing

- sehr einfaches Verfahren: wir verfolgen simulierte Lichtstrahlen und schauen, was ihnen unterwegs passiert
- Anwendung der Gesetze der Strahlenoptik: Spiegelung, Brechung und Absorbtion
- Der Strahlenweg ist umkehrbar!

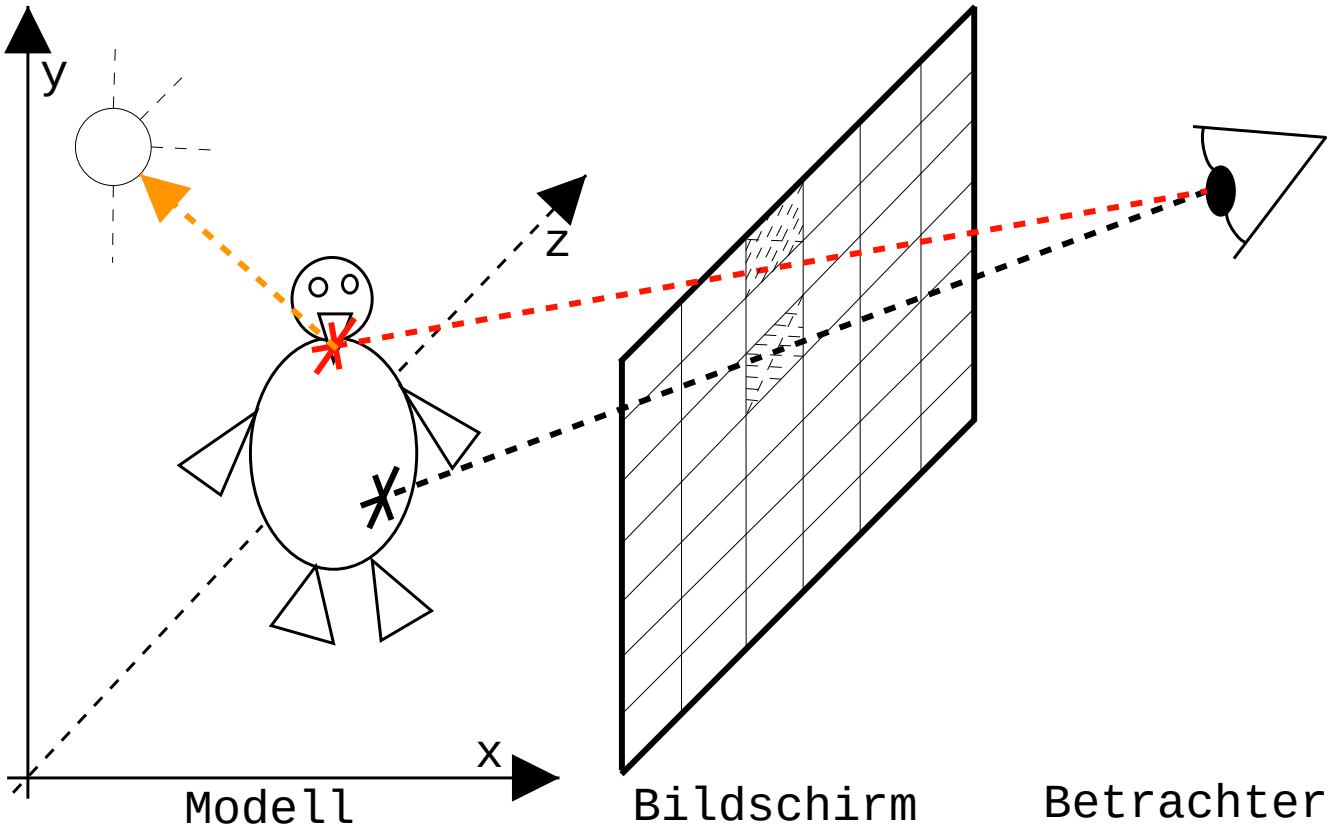
Raytracing - das Problem

- von einer Lichtquelle gehen unendlich viele Strahlen aus
- Strahlen können auf ihrem Weg sehr oft reflektiert, gebrochen oder sogar diffus reflektiert werden
- unendlich große Probleme können von Computern nicht in endlicher Zeit gelöst werden

Raytracing - die Lösung

- da jeder Bildpunkt nur eine eine Farbe annehmen kann, brauchen wir eigentlich nur soviele Strahlen zu verfolgen, wie wir Bildpunkte haben
- die Reflexionstiefe ist begrenzbar
- diffuse Reflexionen kann man zu einem „Umgebungslicht“ aufaddieren

Raytracing



Raytracing – Algorithmus

```
farbe raytrace (strahl) {
  objekt, schnittpunkt = suche_objekt (strahl);
  if (objekt == 0) farbe = hintergrund;
   else farbe = berechen_farbe (strahl,
    schnittpunkt, objekt, lichtquelle);
  return (farbe); }
```

Raytracing – Algorithmus

- `suche_objekt()` ist für Szenen mit vielen Objekten nicht trivial
- `berechen_farbe()` muß wiederum `raytrace()` aufrufen – mehrfache Reflexionen und Brechungen
- `berechen_farbe()` muß auch für jede Lichtquelle einmal aufgerufen werden

Raytracing – Algorithmus

- zusätzliche Berechnungen für Schatten: wenn ein Strahl auf ein Objekt trifft, muß geprüft werden, ob von dortaus die Lichtquelle direkt erreichbar ist oder nicht
- falls nich $\longrightarrow$ der Punkt liegt in einem Schattenbereich
- Schattenbereiche gibt es oftmals viele

Warum Cluster?

- ein Bild in der Auflösung 1024 x 768 hat 786432 Bildpunkte
- für jeden Bildpunkt müssen ca. 10 Teilstrahlen verfolgt werden
- einen Bildpunkt zu berechnen kostet auf moderner Technik im Schnitt ca. 1/10.000 s, ein Bild dementsprechend ca. 80 s

Warum Cluster?

- wir wollen einen Film berechnen, der ca. 5 Minuten läuft
- dazu brauchen wir 7500 Einzelbilder. . . das macht ca. 200 Stunden Rechenzeit für einen PC
- außerdem kühlt die Halle hier so schnell aus

;-))

Danke für die Aufmerksamkeit.

Quellen

- Carsten Fuchs: *Amiga Reflections*, M&T, 1989
- Toni Lama: *3D-Welten*, Carl Hanser Verlag, 2004
- PovRAY-Website: http://www.povray.com