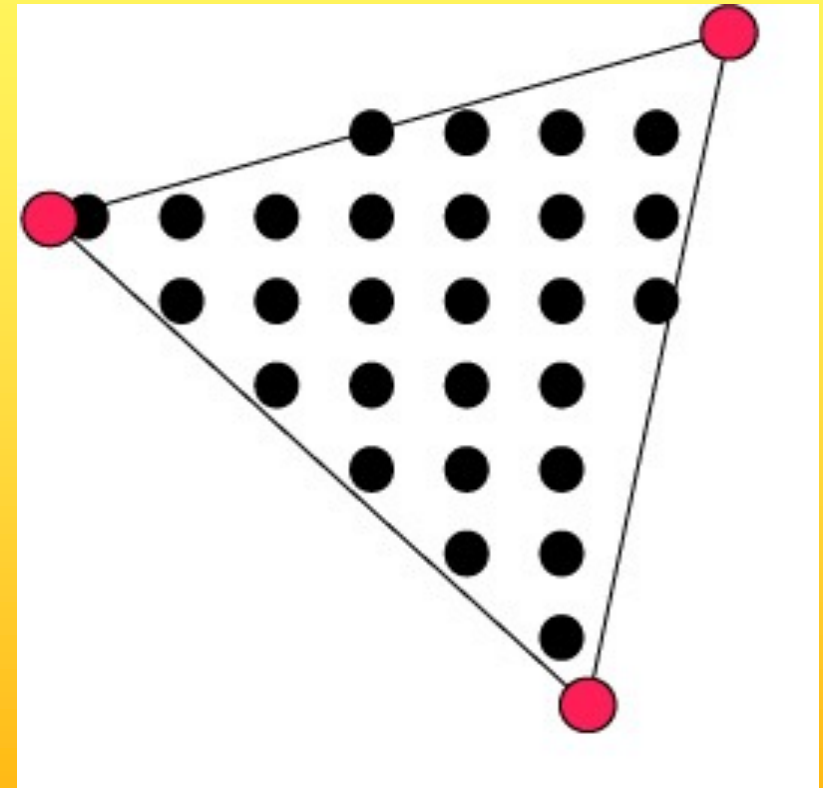
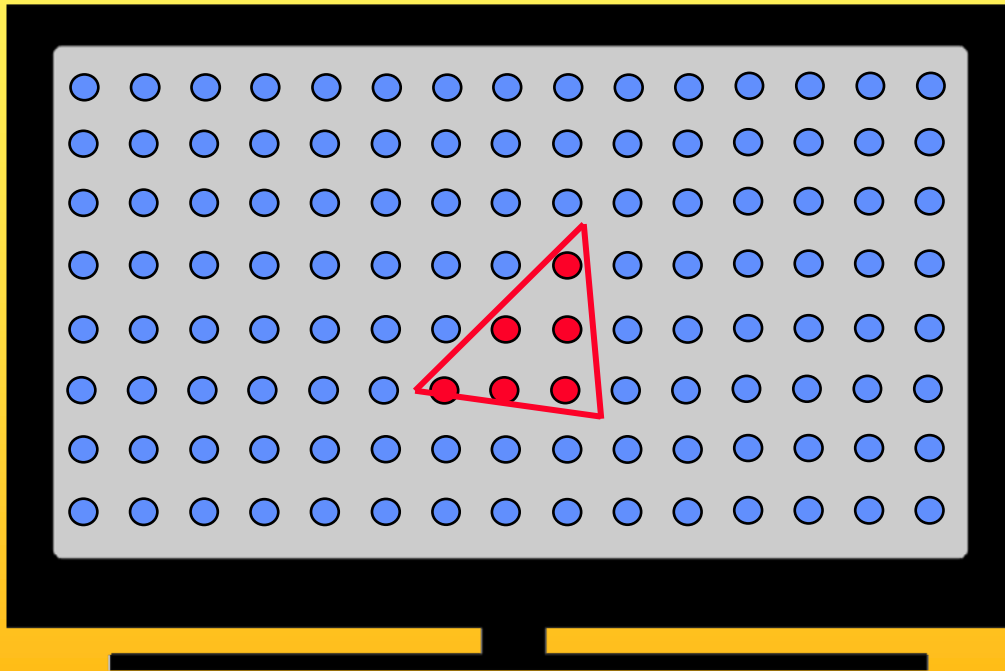


# Realistic Real-time Rendering Today and in the Future

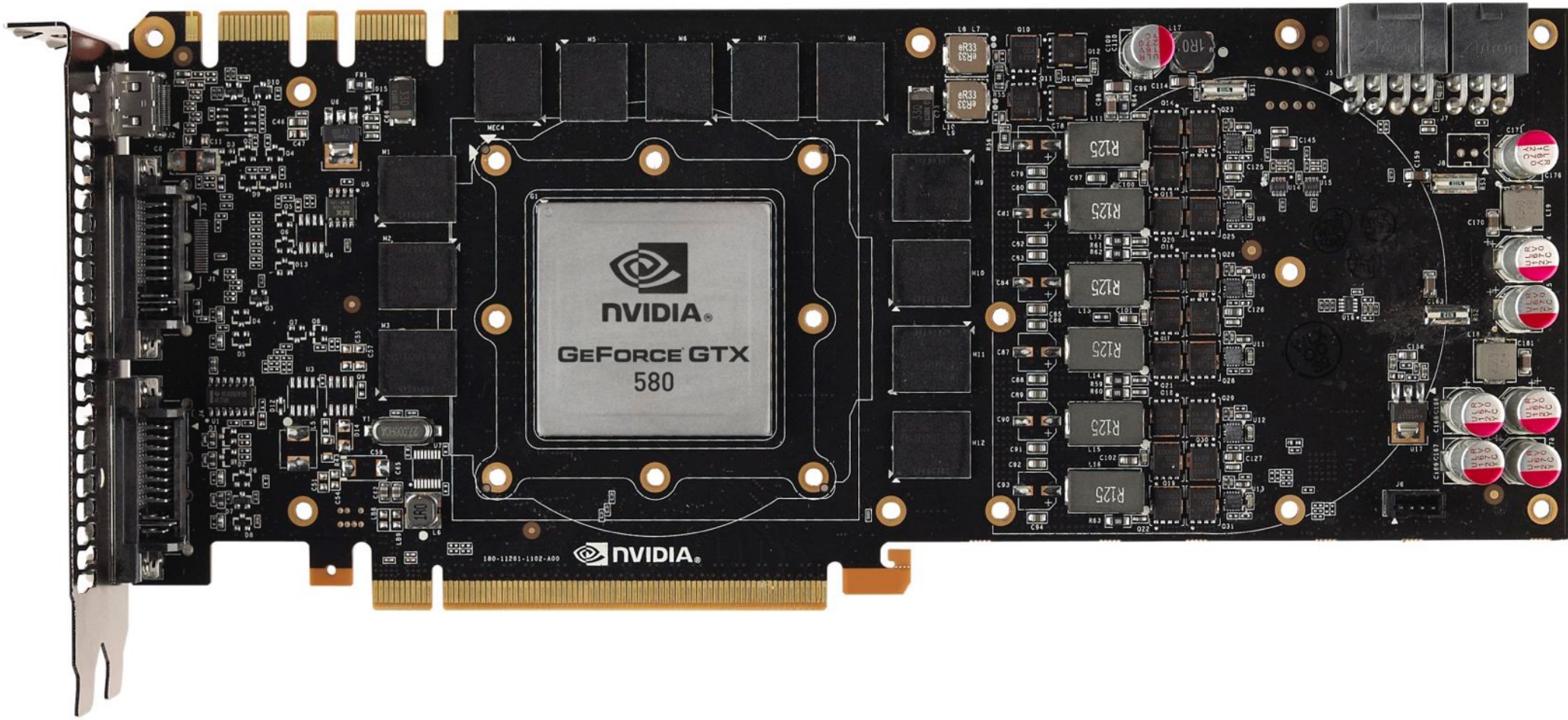
Ulf Assarsson

Chalmers University of Technology

# The screen consists of pixels

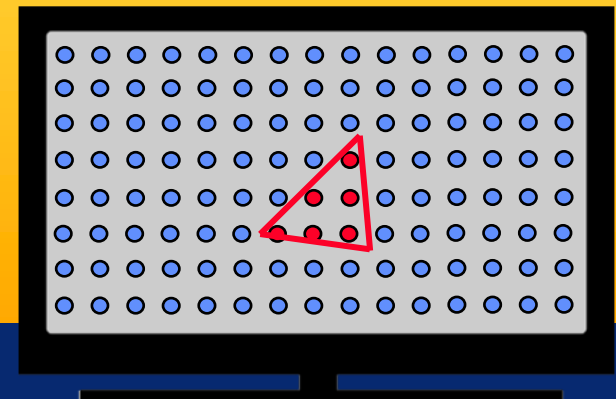
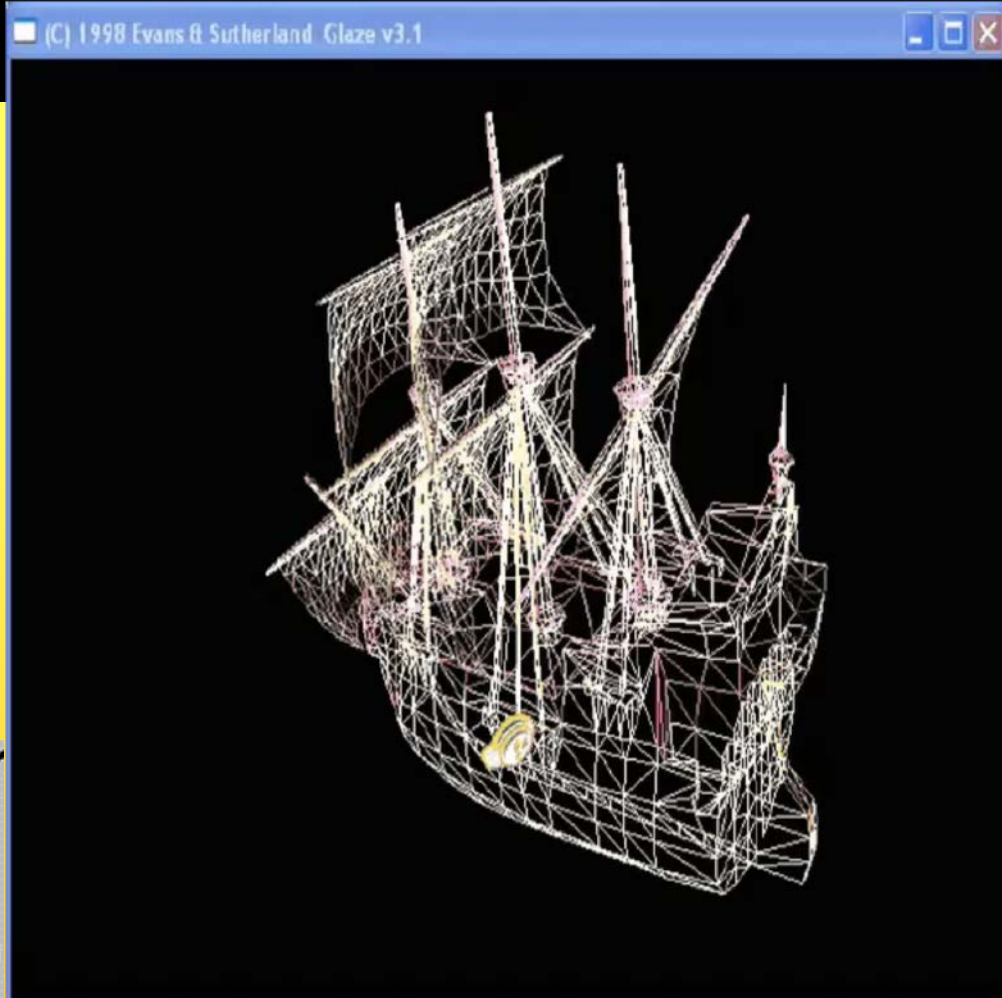
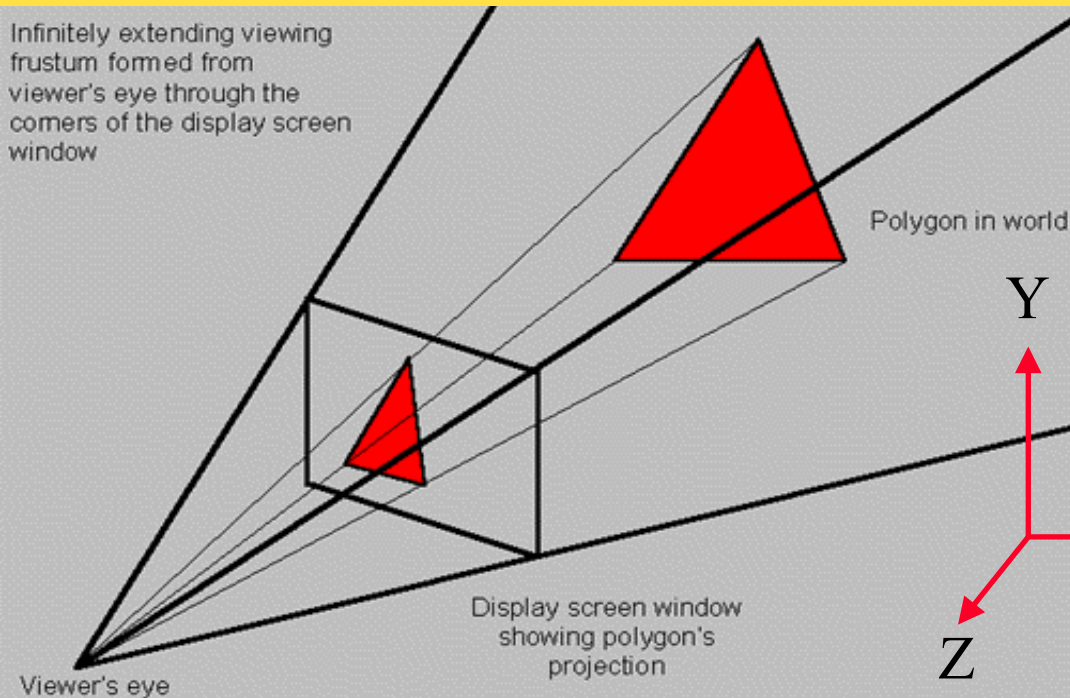


# Grafikkort



# 3D-Rendering

- Objects are often made of triangles
- $x, y, z$ - coordinate for each vertex



# 4D Matrix Multiplication

$$\begin{bmatrix} s_x & \bullet & \bullet & t_x \\ \bullet & s_y & \bullet & t_y \\ \bullet & \bullet & s_z & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

# Real-Time Rendering



# Textures

- One application of texturing is to "glue" images onto geometrical object



+

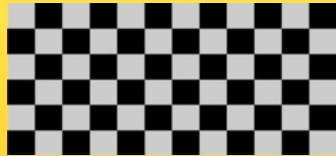


=

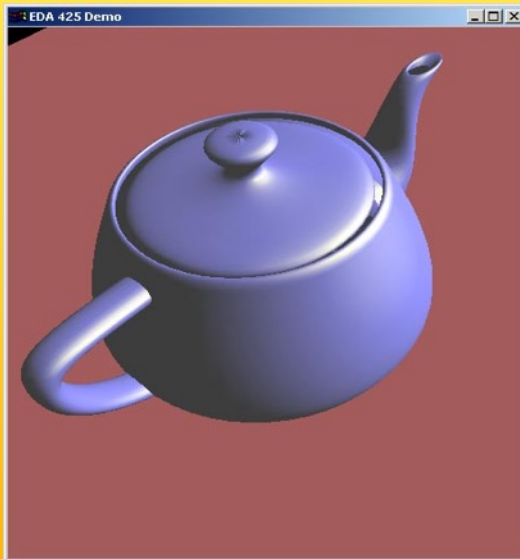


# Texturing: Glue images onto geometrical objects

- Purpose: more realism, and this is a cheap way to do it



+

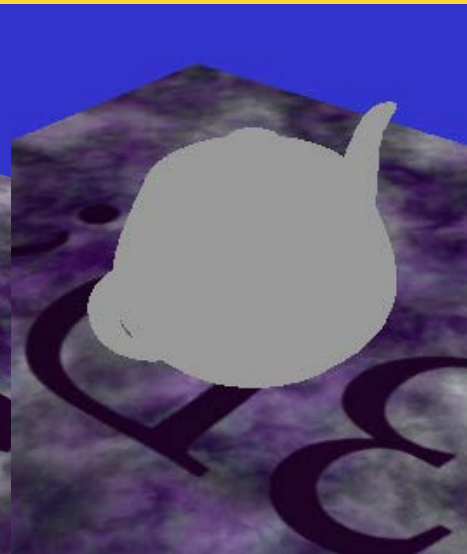
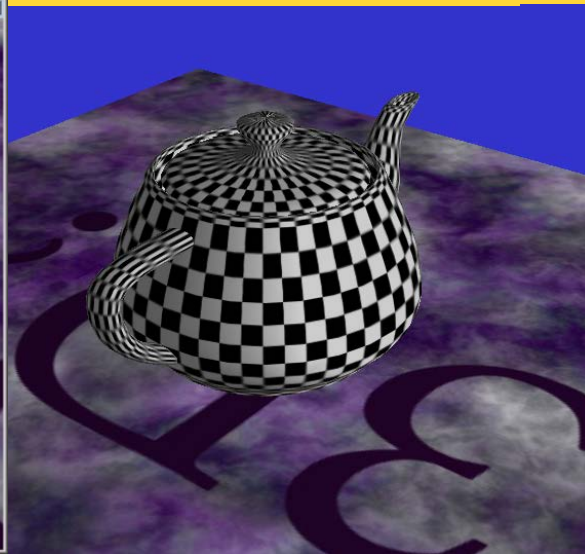
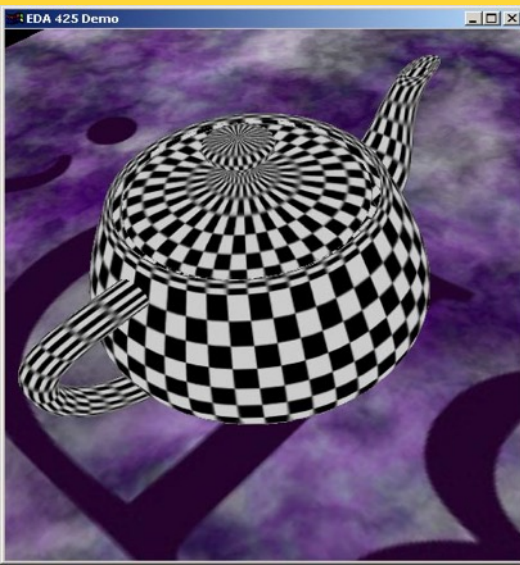
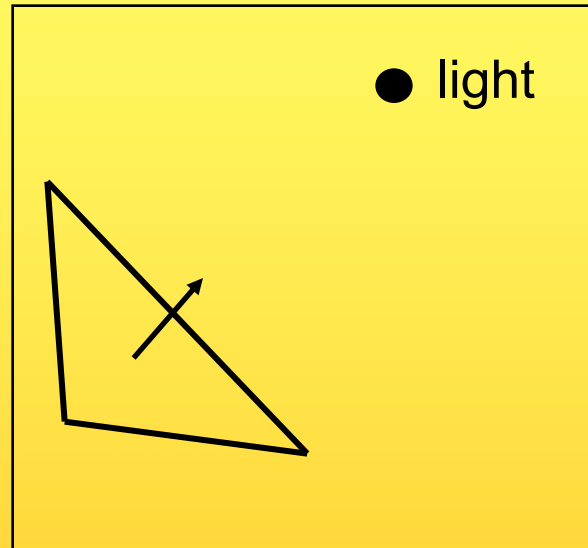


=





# Light computation per triangle

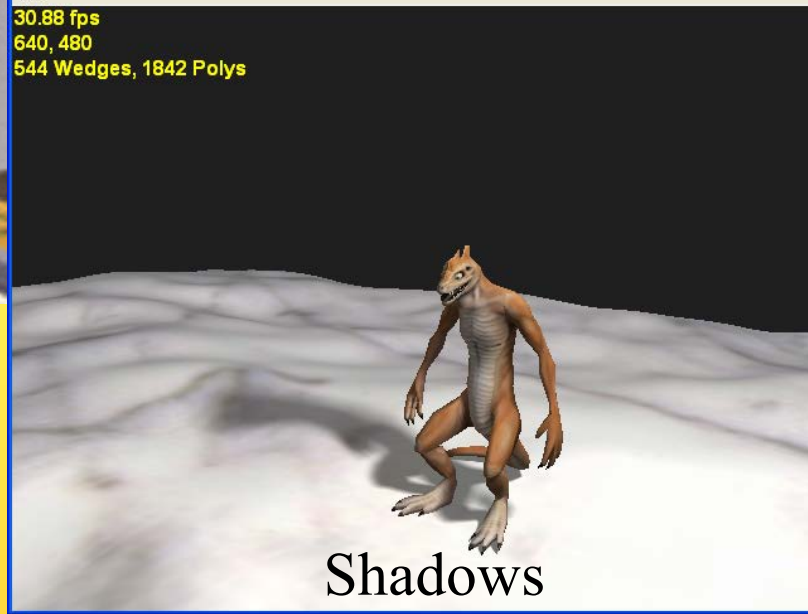


# More



Reflections

30.88 fps  
640, 480  
544 Wedges, 1842 Polys



Shadows



Materials



Water



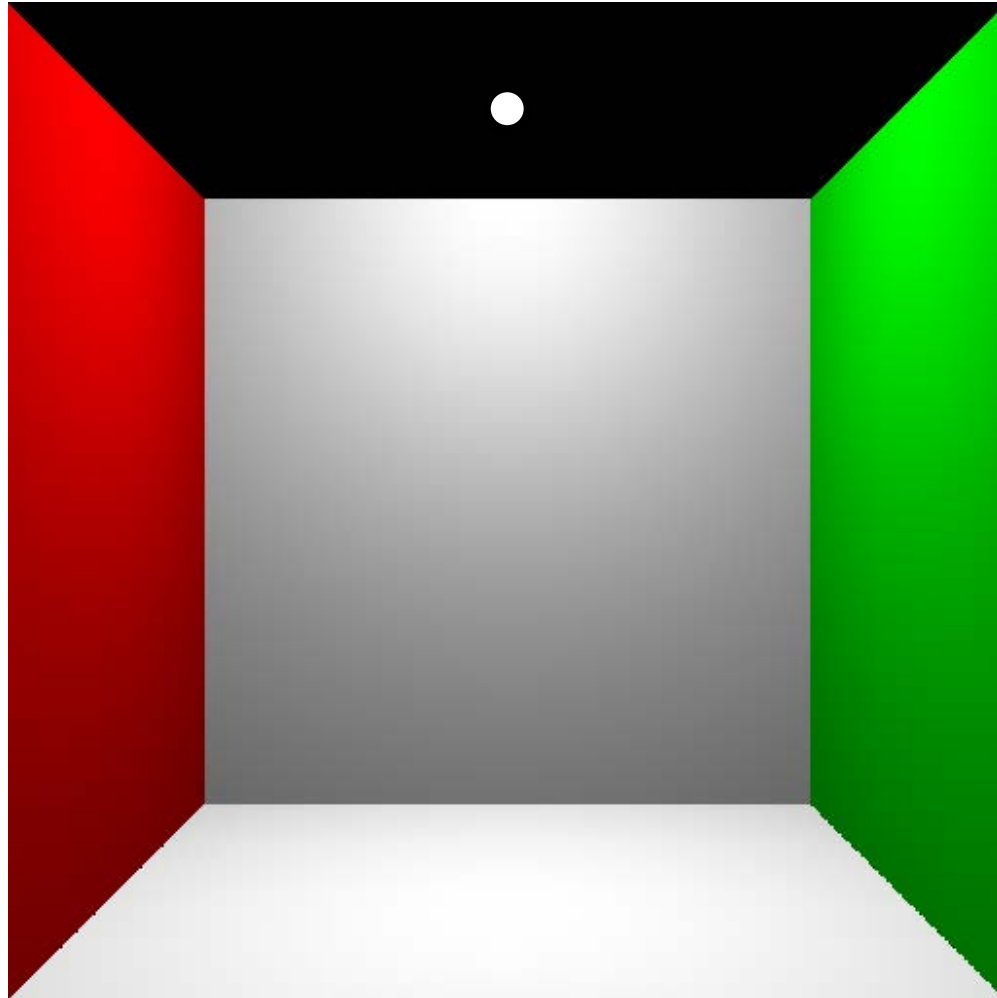
Airlight



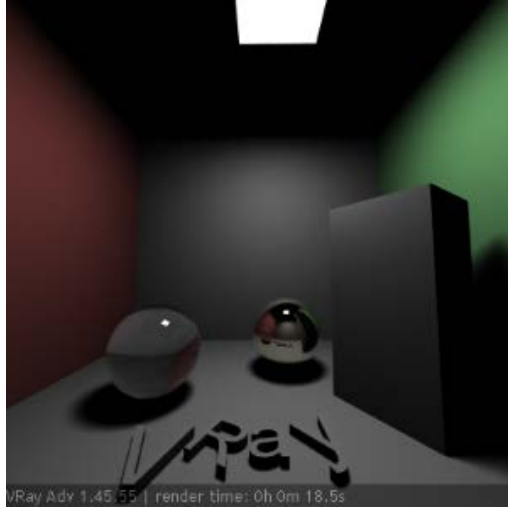
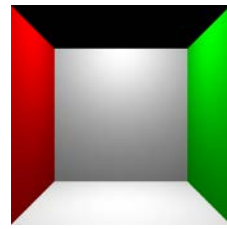
Fire

# Light Bounces

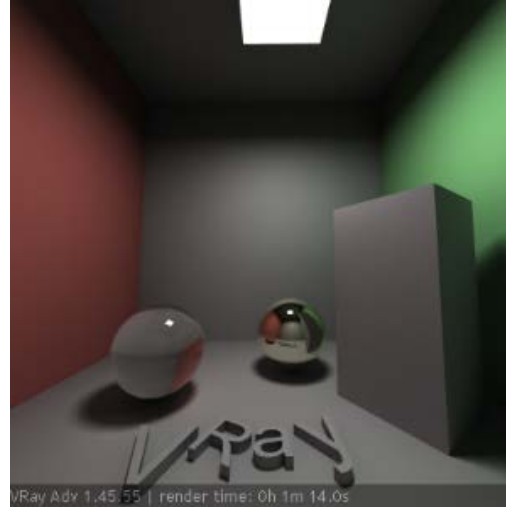
Typical test box (Cornell box), often compared to actual photograph:



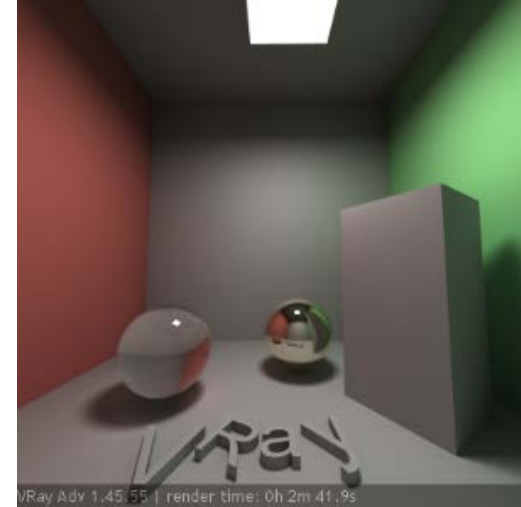
# Light Bounces



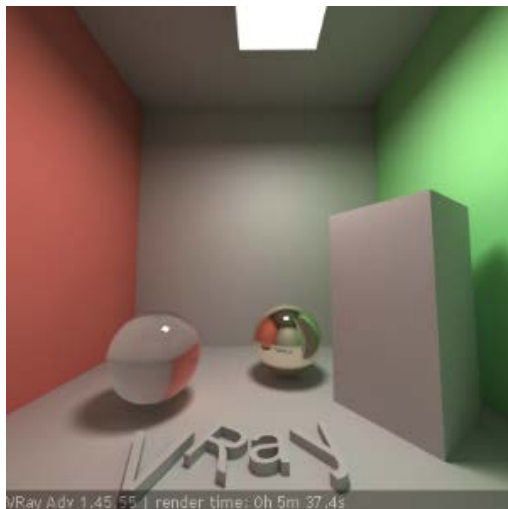
1 bounce



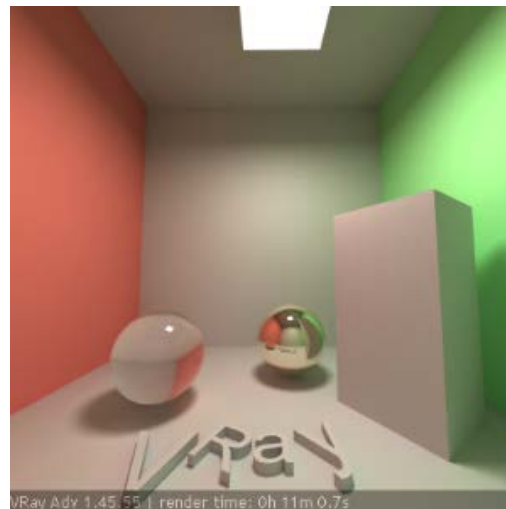
2 bounces



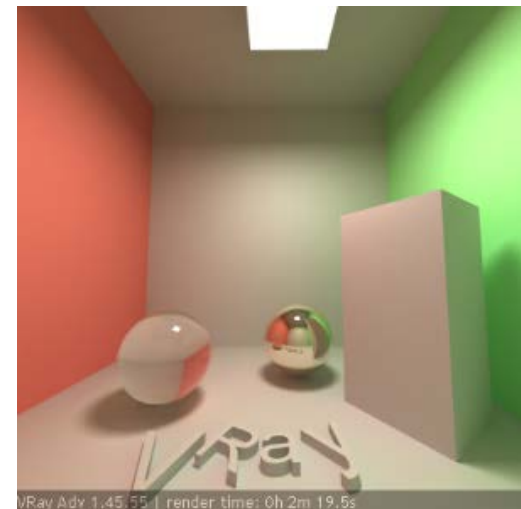
3 bounces



5 bounces



9 bounces



infinite bounces

# The Problem of Computer Graphics

- Is not CG soon a solved problem?
- Will not computers soon be fast enough?

Probably not...

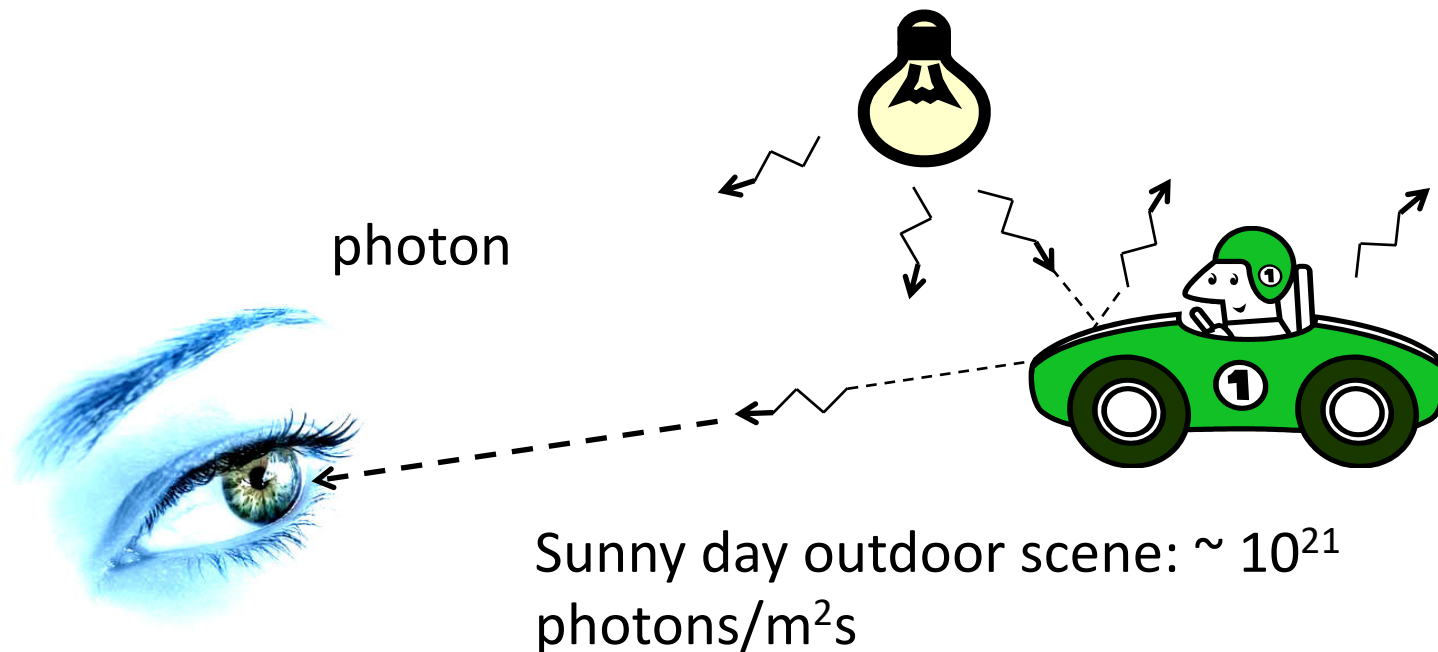


~20 to ~10<sup>15</sup> photons/s

The eye has a resolution of 130M receptors:  
120M gray scale (rods / stavar)  
7M color (cones / tappar)

# The problem of Computer Graphics

- Eye sensitivity:  $\sim 20$  photons/s to  $\sim 10^{15}$  photons/s
- So, if we could trace only the photons that hit the eye, the problem would be limited.
- But, the only really guaranteed 100% correct way (still) is tracing photons from light to eye.



## Facts:

- Eye sensitivity:  $\sim 20$  to  $\sim 10^{15}$  photons/s
- Sensitivity is logarithmic
  - i.e., difference between 100 or 200 photons is as noticeable as for  $10^{10}$  or  $2 \cdot 10^{10}$  photons
- $\sim 10^{21}$  photons/m<sup>2</sup>s
- 1 photon costs  $\sim 10,000$  cycles

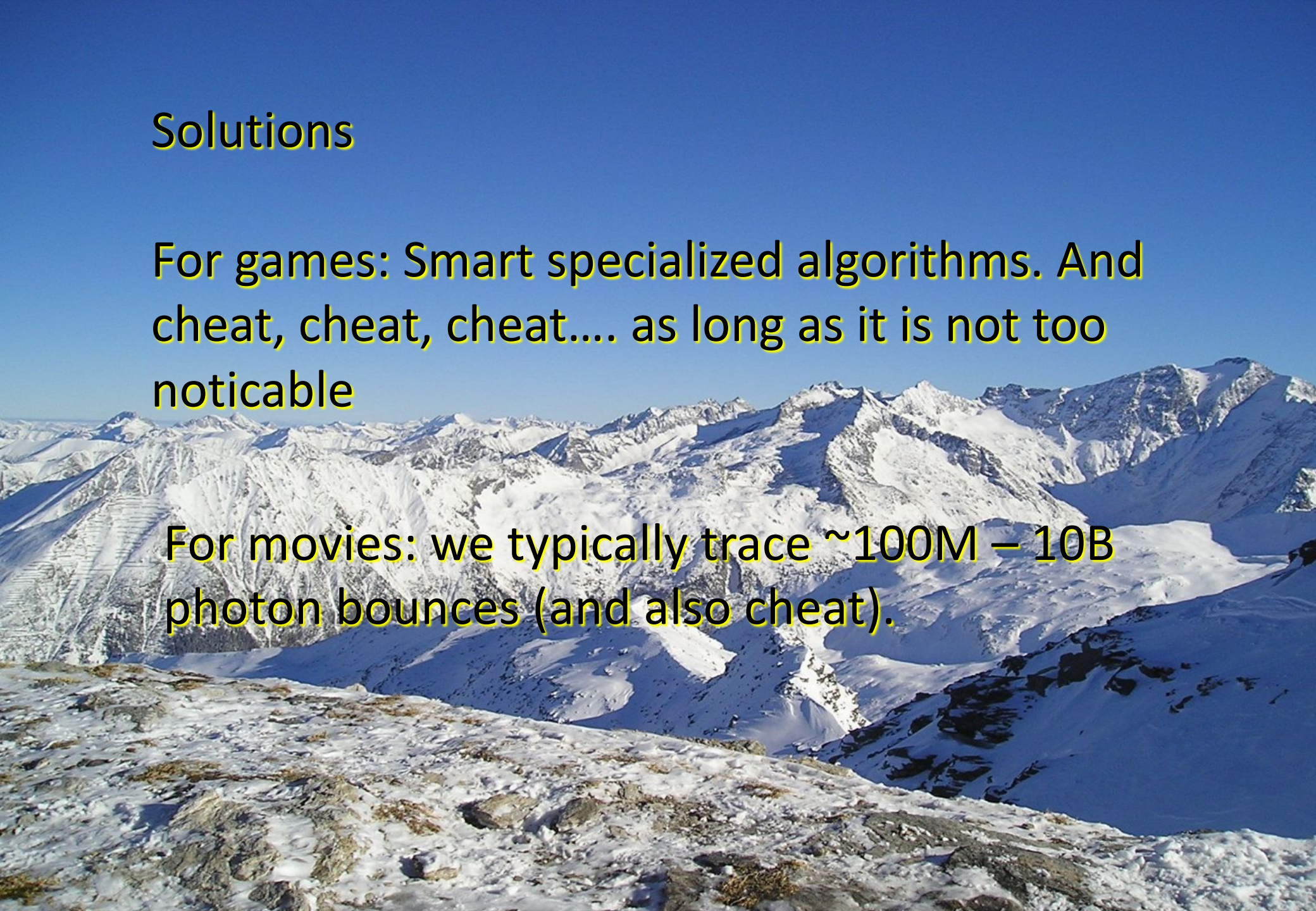
10 billion years per square meter for 1 computer



# Solutions

For games: Smart specialized algorithms. And cheat, cheat, cheat.... as long as it is not too noticable


For movies: we typically trace  $\sim 100M - 10B$  photon bounces (and also cheat).





Typically: 100M – 1B photon bounces





So, what is the state-of-the-art for  
real-time graphics today?

**6** WZPZ  
EVENING NEWS

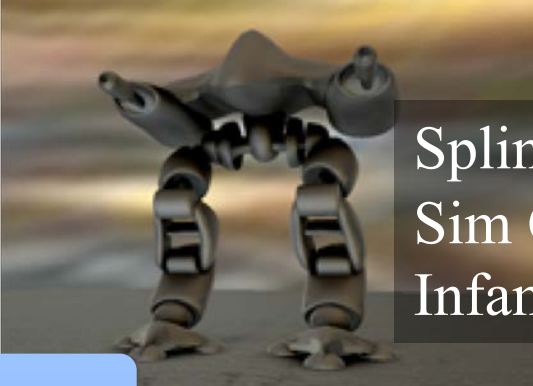
**LIVE AT THE MEADE BUILDING**



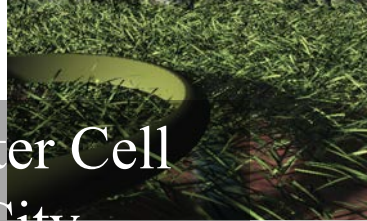
Fallout 4,  
NVIDIA



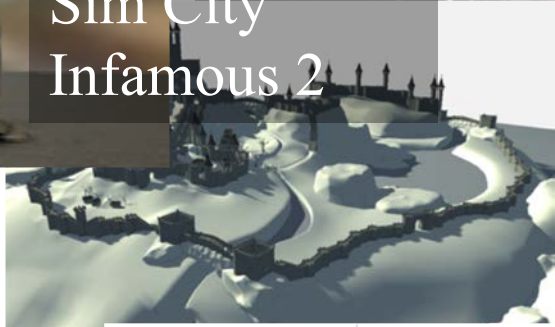
Hair and Fur  
(games)



Shadows  
(games)



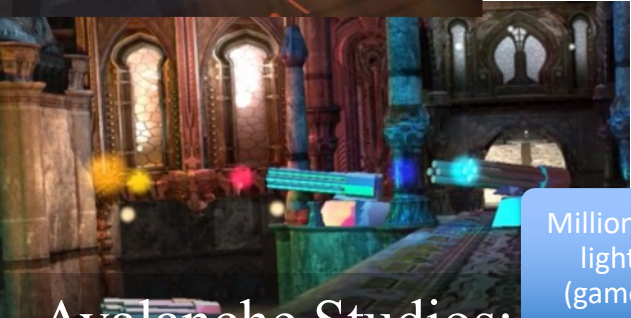
Splinter Cell  
Sim City  
Infamous 2



Airlight  
(games)

Our  
Research  
Projects

Scene  
compression



Millions of  
lights  
(games)

Free  
Viewpoint  
Video



GPGPU



Avalanche Studios:

- Just Cause 3
- Doom (latest)
- Bosch
- Intel

e.g. sorting:

19 5 100 1 63 79  
 ↓  
 1 5 19 63 79 100

# Beast + Unreal Engine



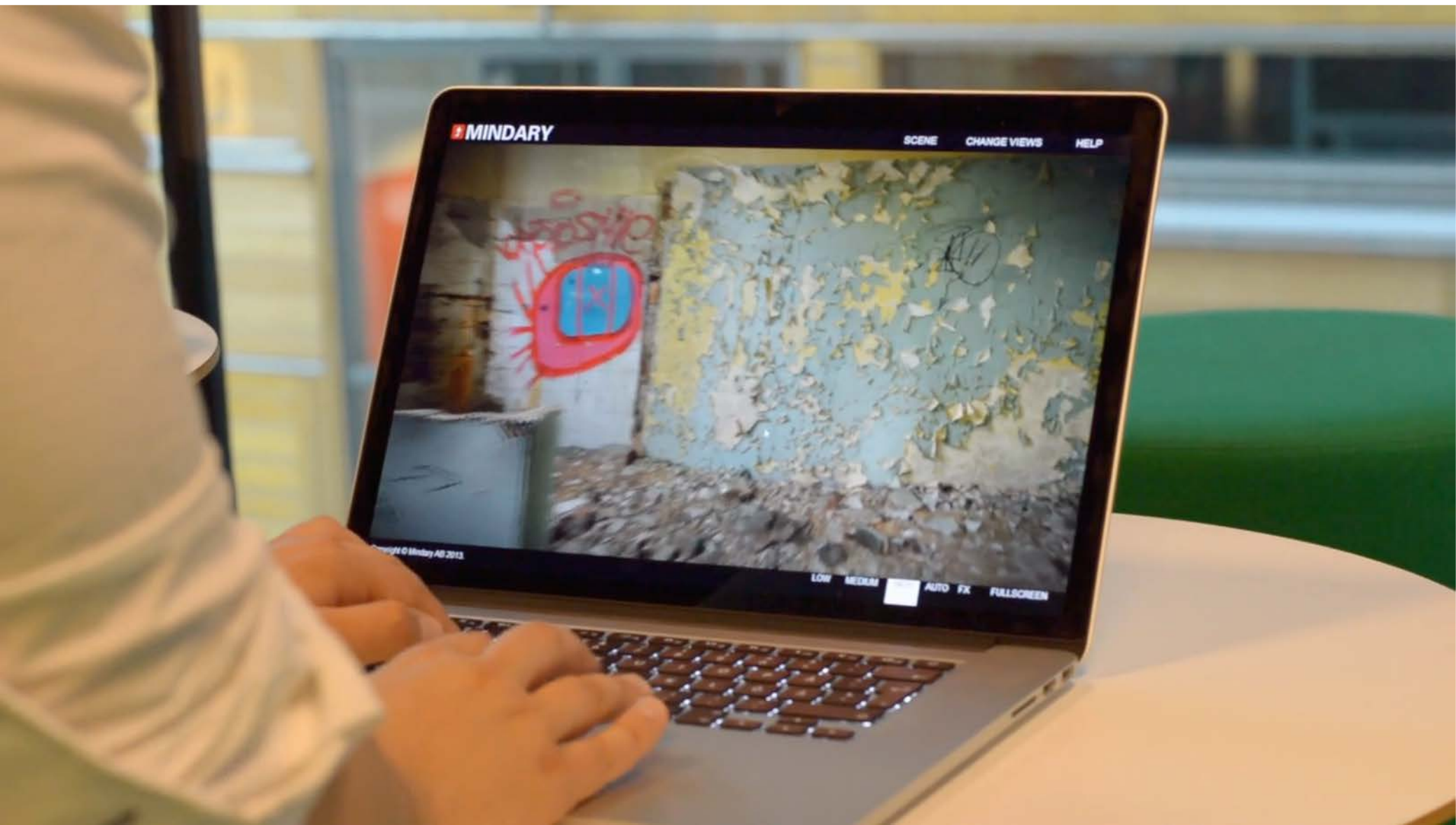
# Mix of graphics and photographing

Textures from photographs



# Mix of graphics and photographing

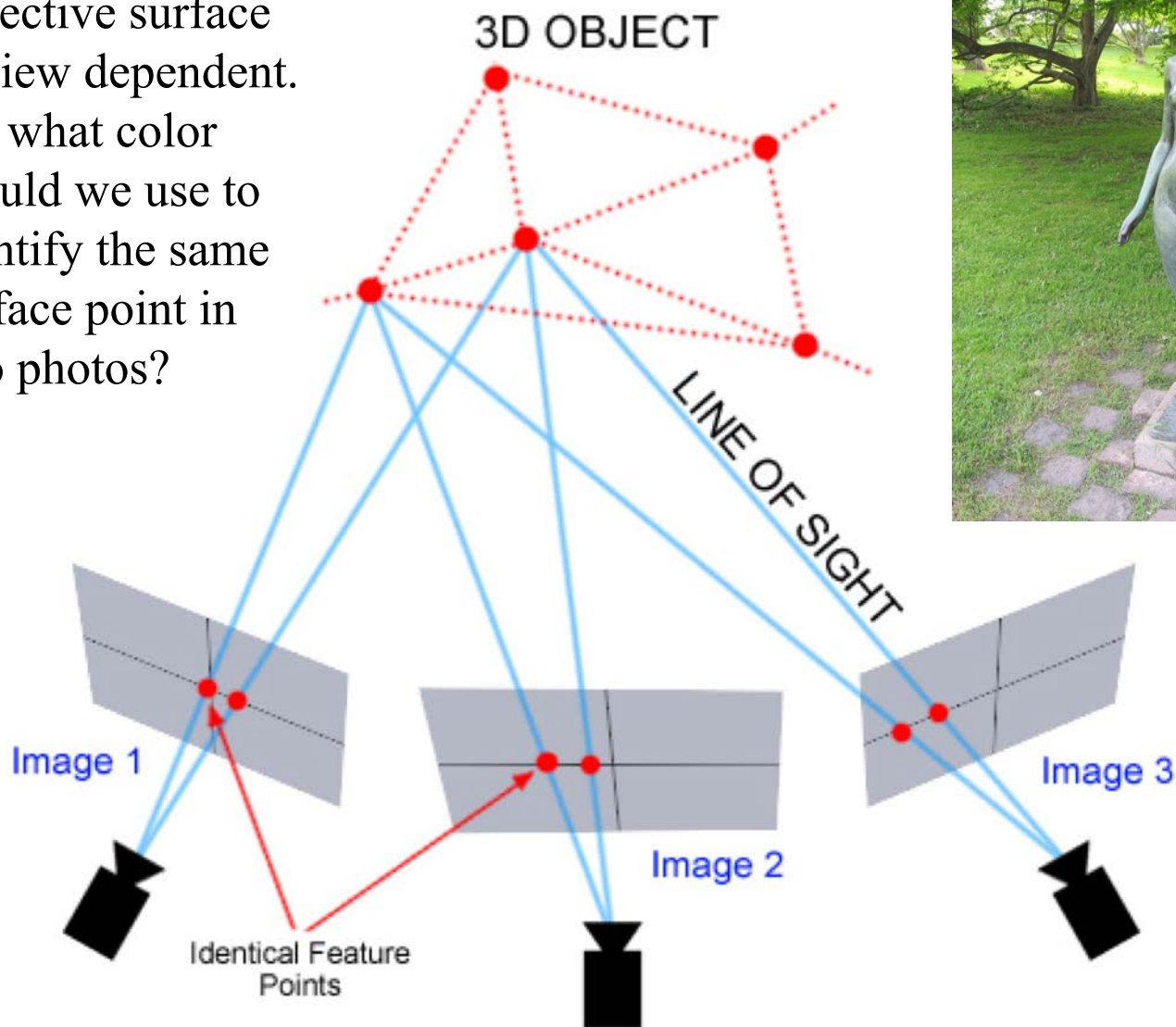
- Istället för att modellera 3D-grafik och beräkna realistiskt utseende:
  - Fotografera/filma verkligheten och konvertera den till 3D-grafik.
- Ofta vill man mixa modellerade och verkliga konverterade objekt.



<http://www.cse.chalmers.se/~uffe/mindary/demo/v2.html>

# Reflective surfaces (=view-dependent colors) are a problem

The color of a reflective surface is view dependent. So, what color should we use to identify the same surface point in two photos?





Colors are view-dependent.  
The more reflective surface,  
the larger the problem.

“Solutions”, e.g.:

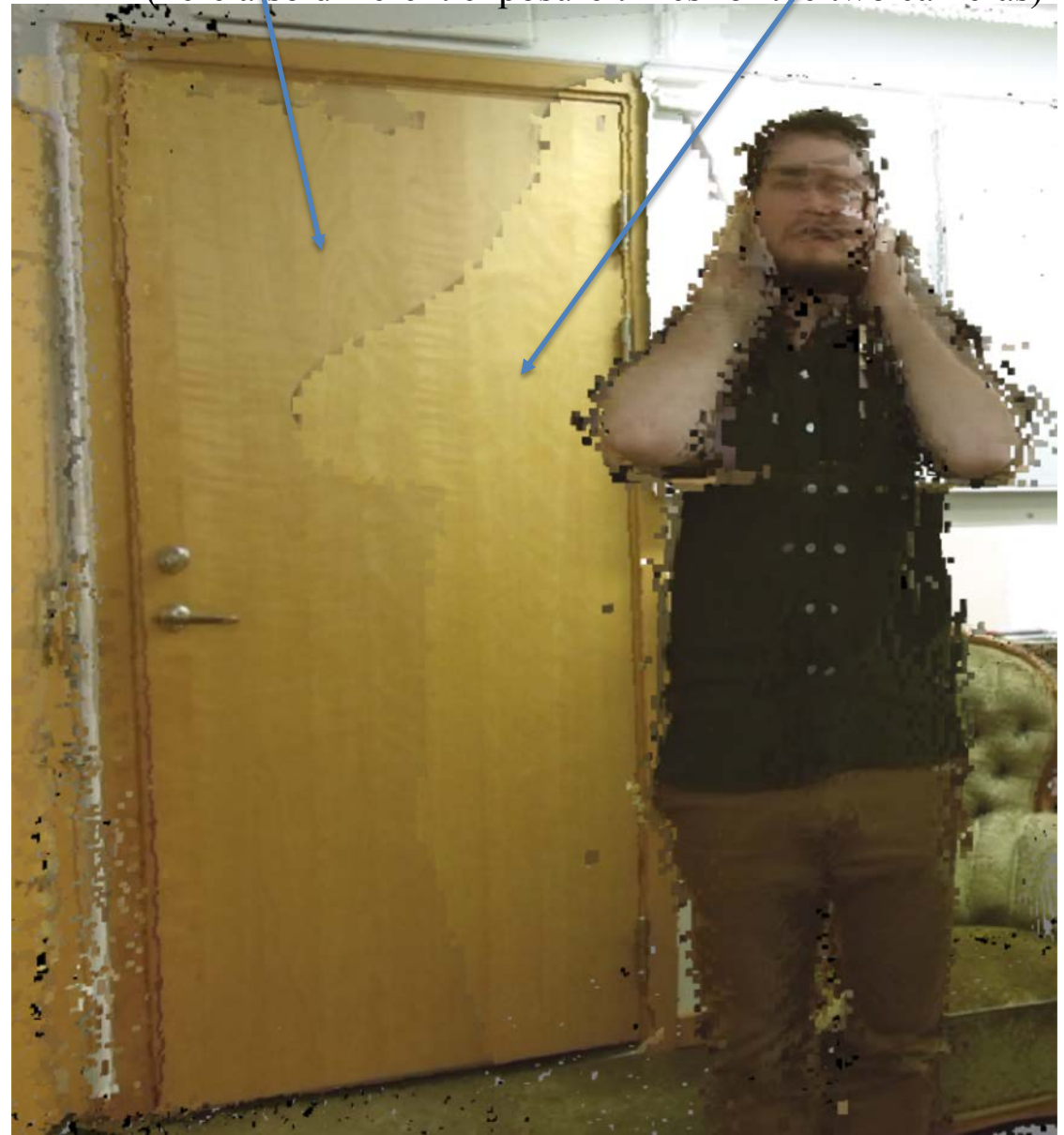
- Use color gradients instead of raw colors.
- Suppress reflections with polarization filters when photographing.
  - Instead, computer-generate the reflections when visualizing the 3D scene.

Neither works perfectly - very reflective surfaces (e.g. mirror) not at all.

Average color from 2 cameras

Color from 1 camera

(here also different exposure times for the two cameras)



Anyways,

1. Generate static (non view-dependent) color textures
2. Add computer-generated reflections (most important view-dependent effect)



Now with **photo textures** (view-independent)  
and computer-generated view-dependent reflections

# Unreal Engine 4

Combining photo textures and computer-generated view-dependent reflections

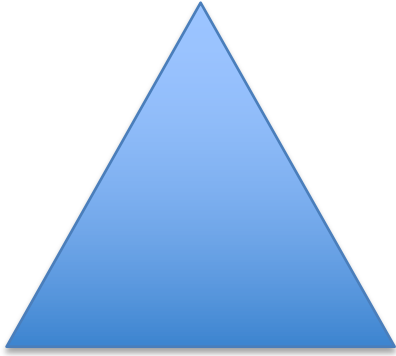


Increasing the amount of geometric  
details

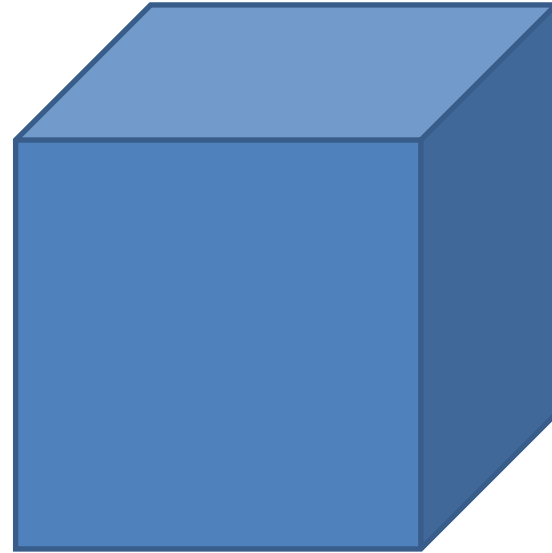
# Triangles



# Voxels



Triangle  
36 bytes



Voxel  
Volume – element  
1 bit

# Voxels

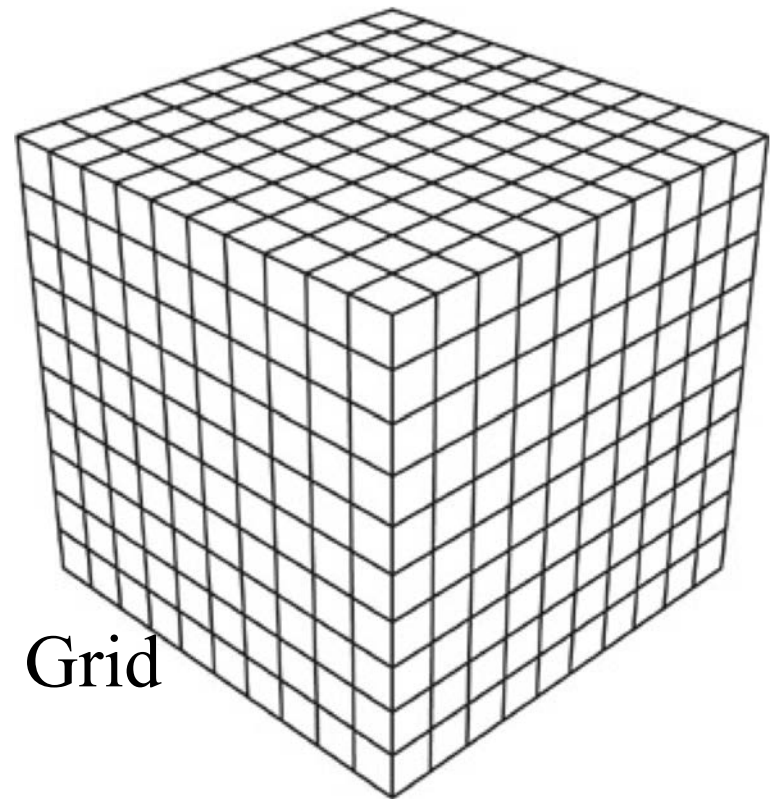
- Desirable to be able to use very high resolutions



# Voxels

One possible data structure:

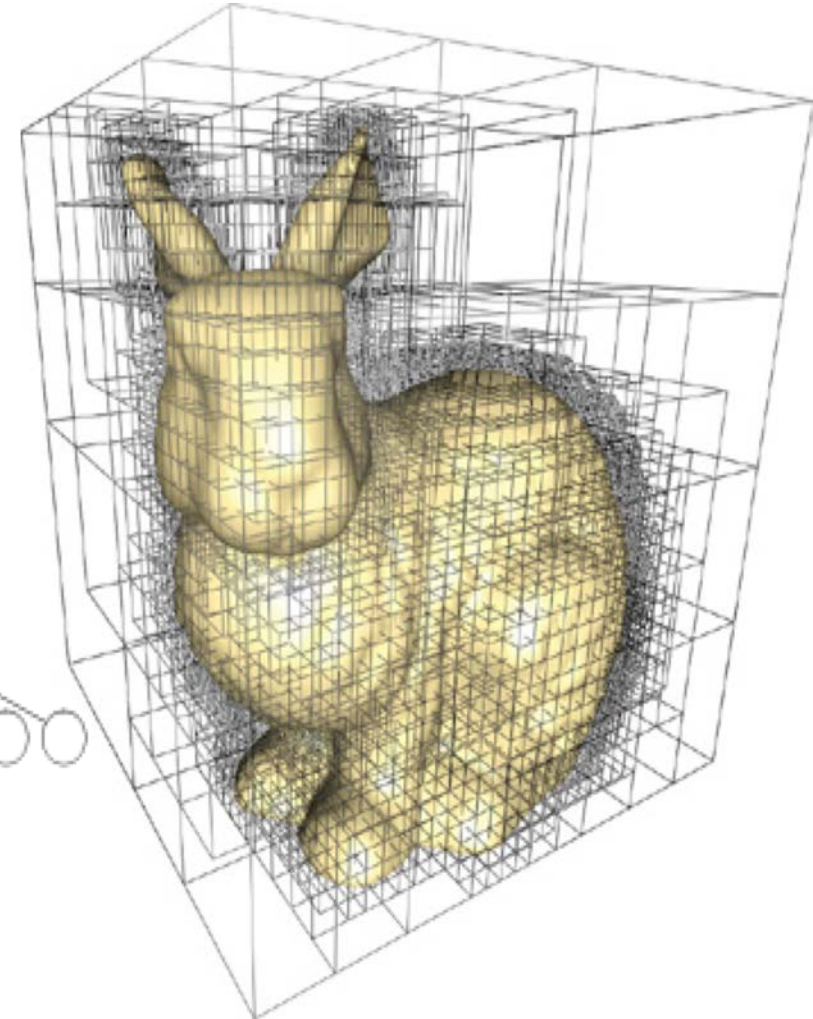
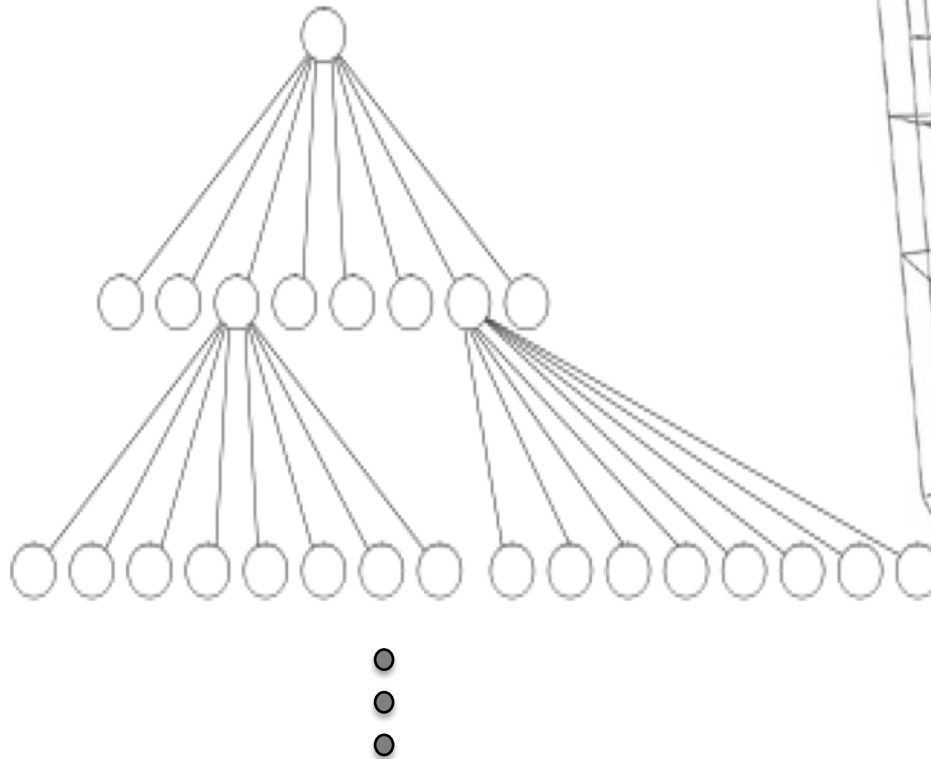
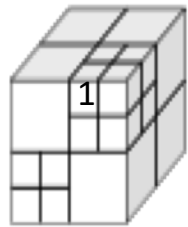
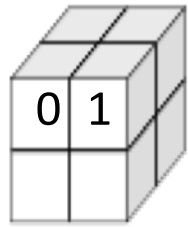
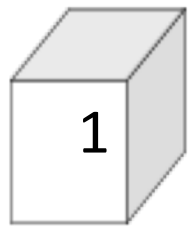
- Grids – 3D array of 0:s and 1:s





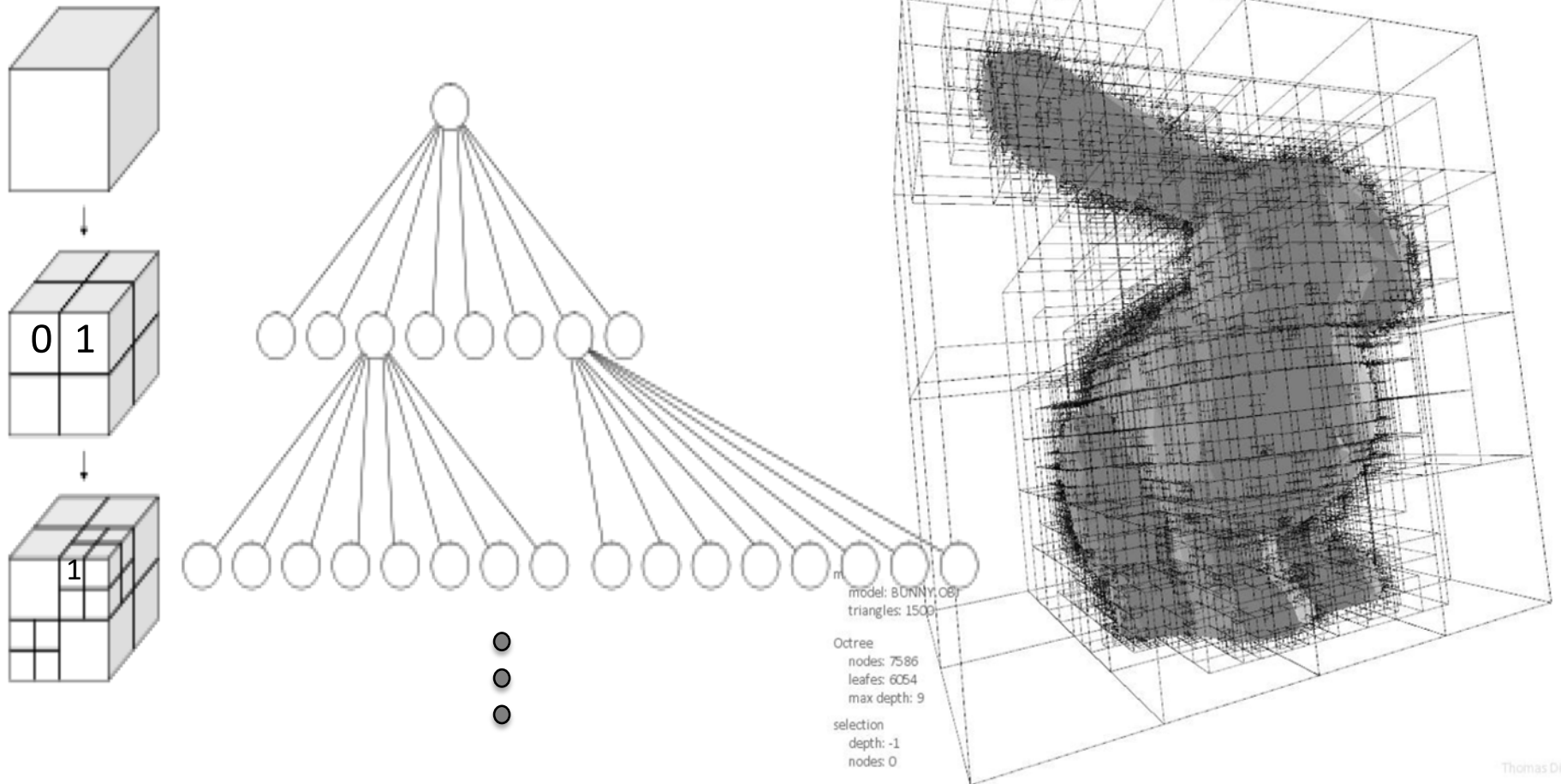
# Sparse Voxel Octree

Each node has eight children, representing an octant of the parent node's volume.



# Sparse Voxel Octree

Each node has eight children, representing an octant of the parent node's volume.



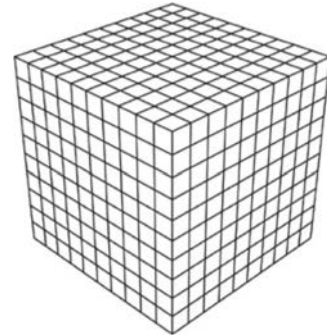
# Sparse Voxel Octree

- SVO: Id Software, page 6
- 1.15 bits/ non-empty voxel
- DAG: e.g., 0.08 bit/non-empty voxel



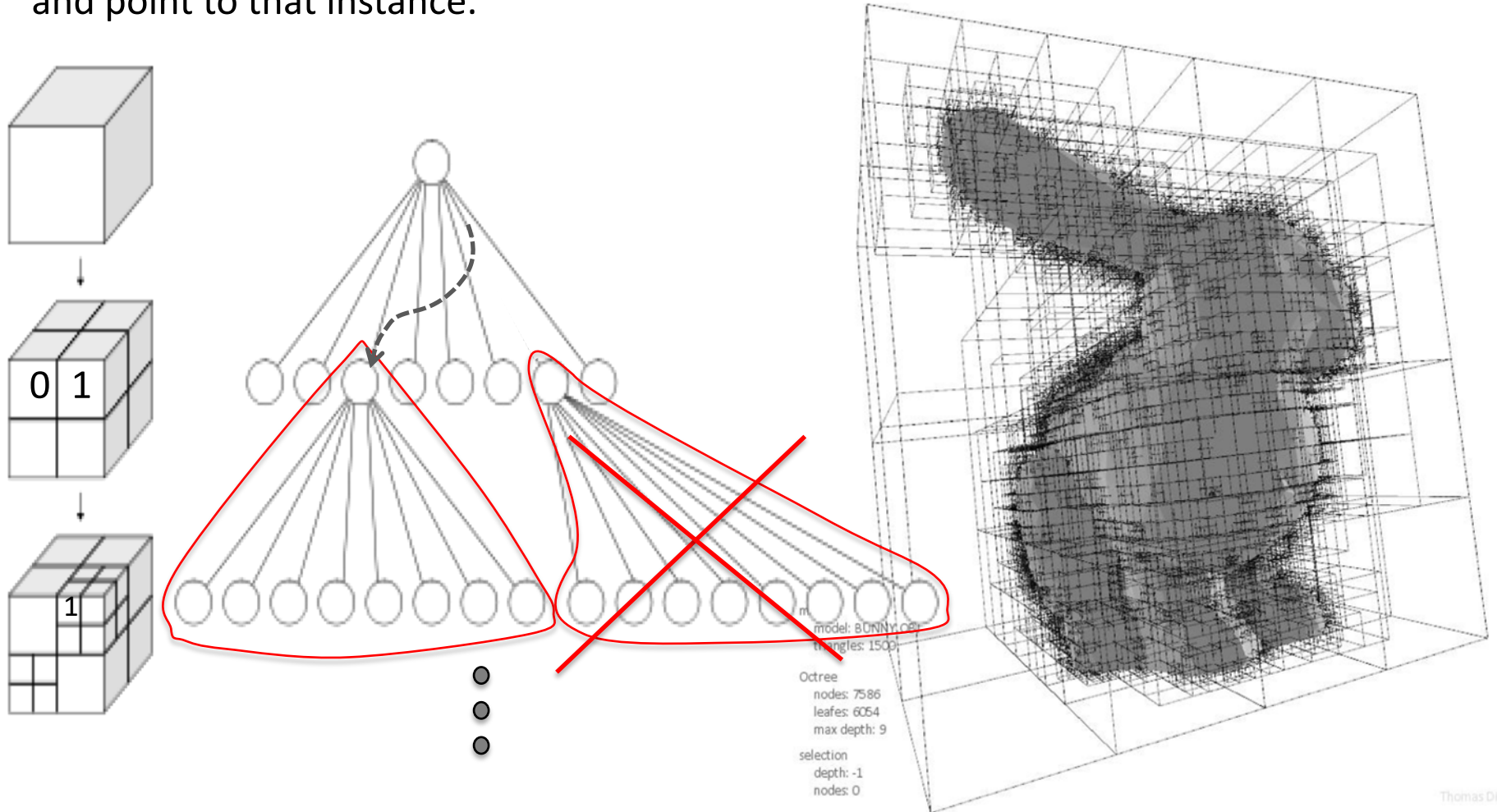
# Voxels

- Voxel = 1 bit.
- We currently handle scene of res =  $128.000^3$ 
  - Naively: 262 TB
  - We => < 1GB



# Our Voxel DAGs

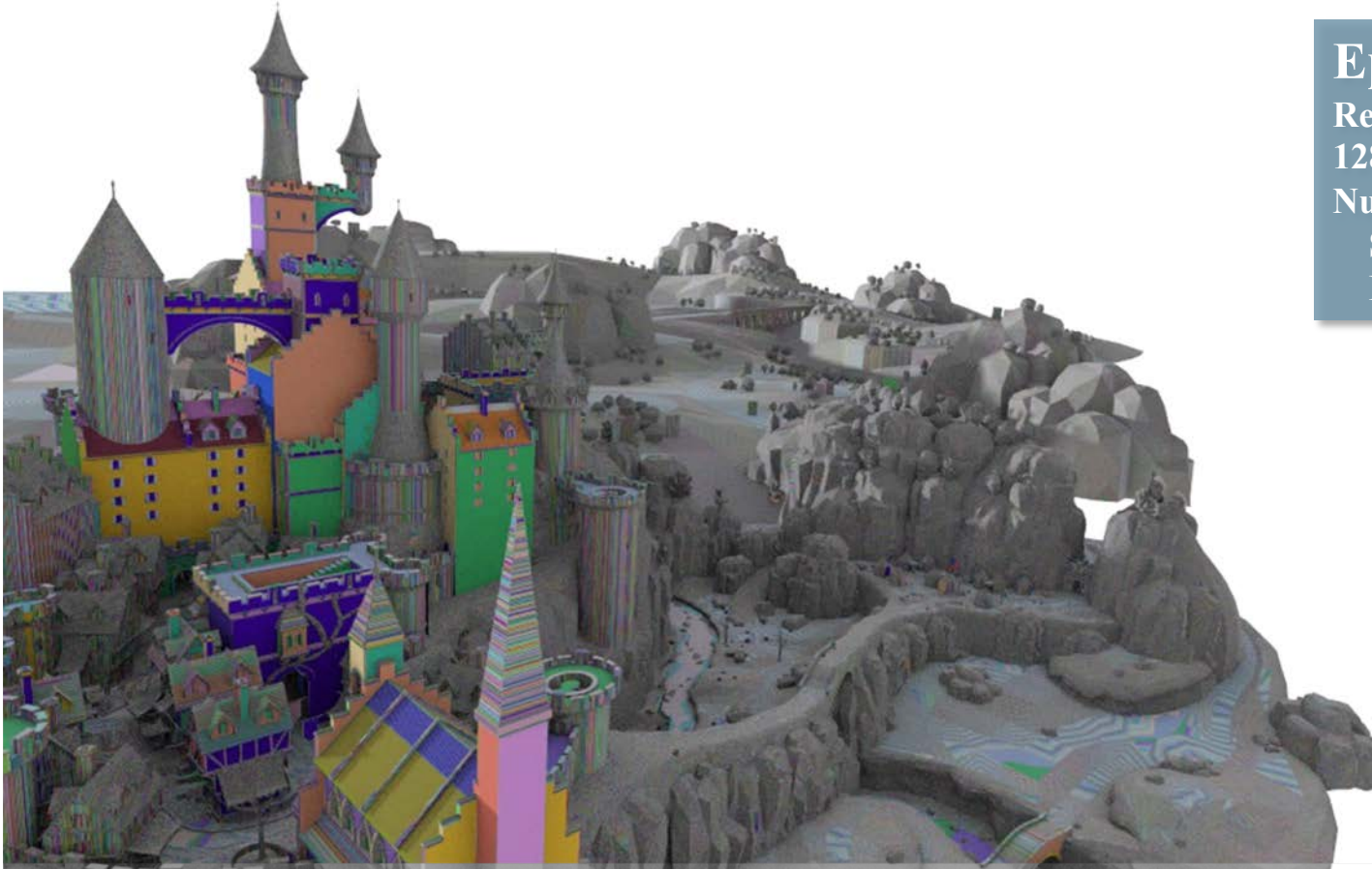
For identical subgraphs, only store one instance, and point to that instance.





**Resolution:**  
**131072<sup>3</sup>**  
**(~900MB)**

# Visualizing Identical Subtrees



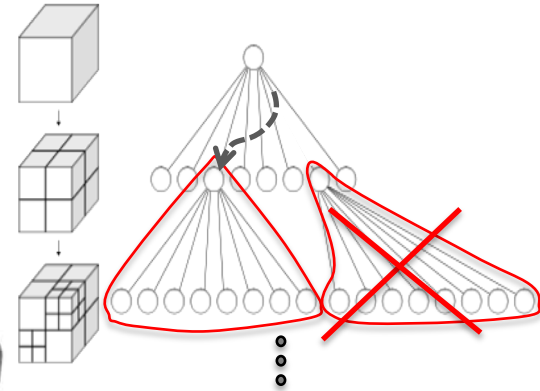
## Epic Citadel

Resolution:  $128K \times 128K \times 128K$

Number of nodes

SVO: 5.5 billion

DAG: 45 million (0.8%)



# Visualizing Identical Subtrees

## Hairball

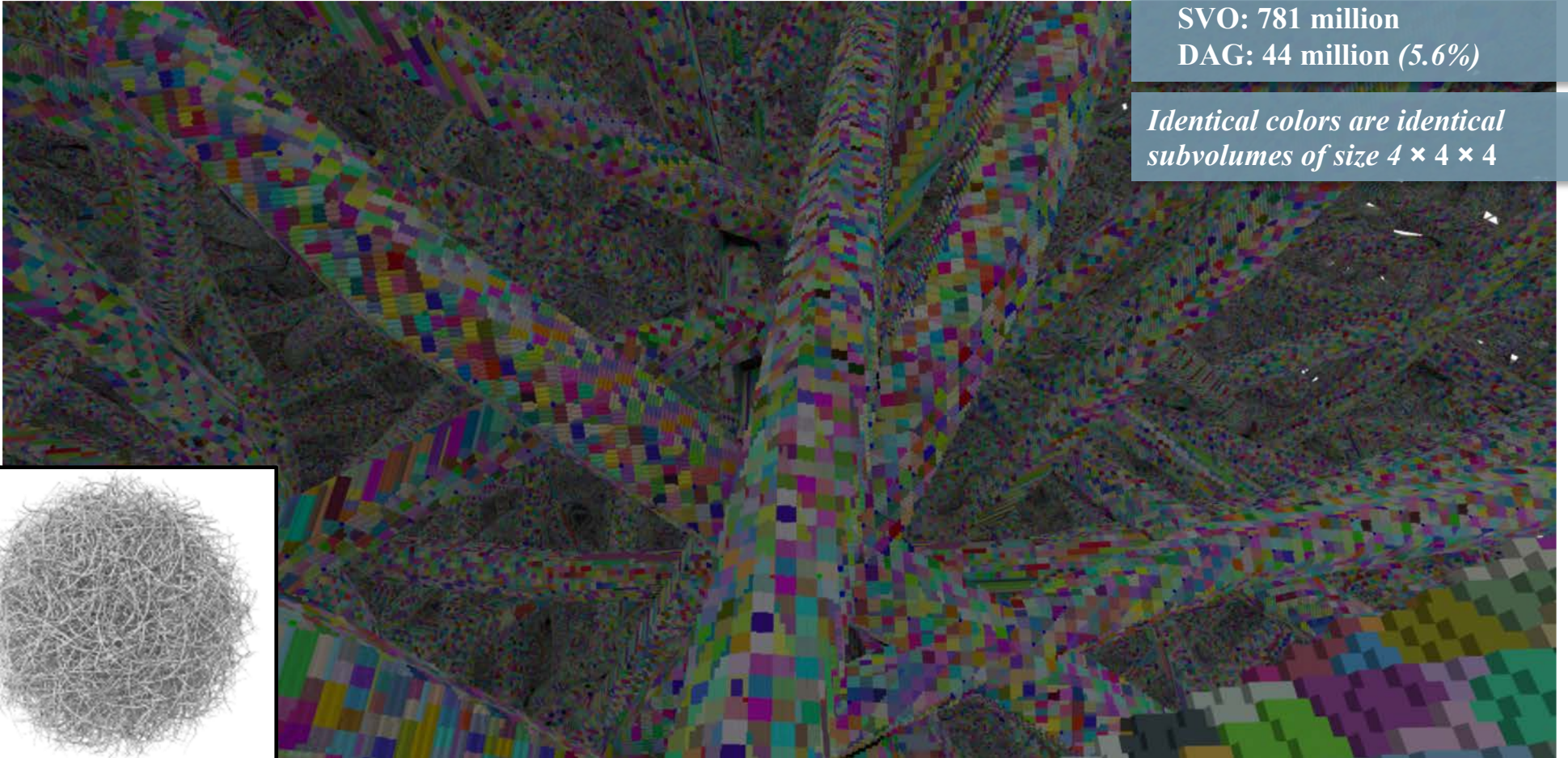
Resolution: 8K × 8K × 8K

Number of nodes

SVO: 781 million

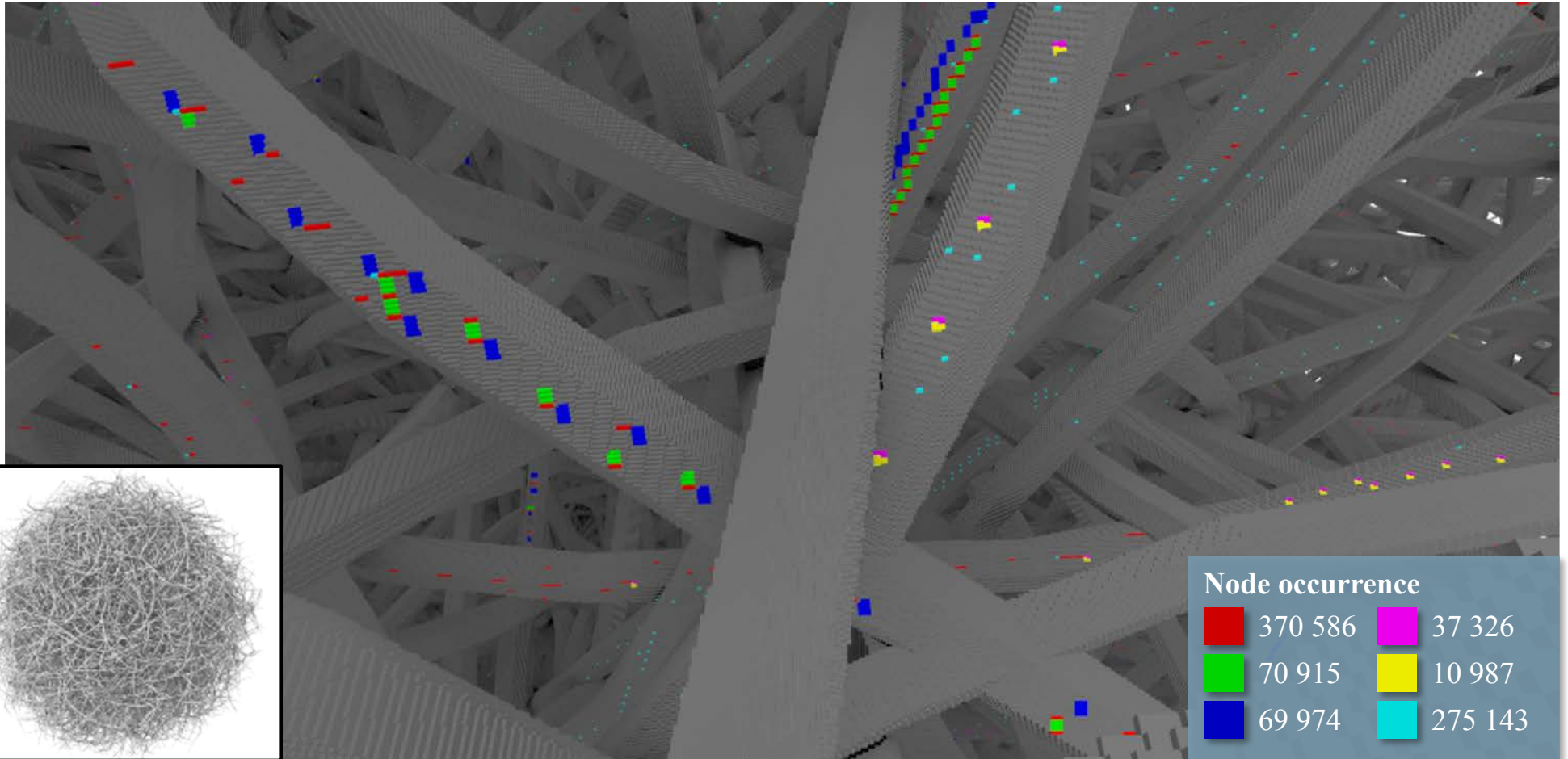
DAG: 44 million (5.6%)

*Identical colors are identical  
subvolumes of size 4 × 4 × 4*





# Visualizing Identical Subtrees



**From static scenes  
to dynamic (moving) scenes,  
i.e., Free Viewpoint Video**

# Free Viewpoint Video

Want:

1. convert a real scene to 3D graphics, 24 times/second.
2. Render scene from any viewpoint.



Med 2 eller fler kameror kan man beräkna djup – precis som våra ögon.



Med 2 eller fler kameror kan man beräkna djup – precis som våra ögon.



Med 2 eller fler kameror kan man beräkna djup – precis som våra ögon.

Varje kub  $\sim 1\text{cm}^3$ . Önskar  $\sim 1\text{mm}^3$



# Free Viewpoint Video

## KINECT

Input: Three depth streams from Kinect cameras  
Voxel grid resolution: 512x512x512  
Frames: 480



## Cameras:



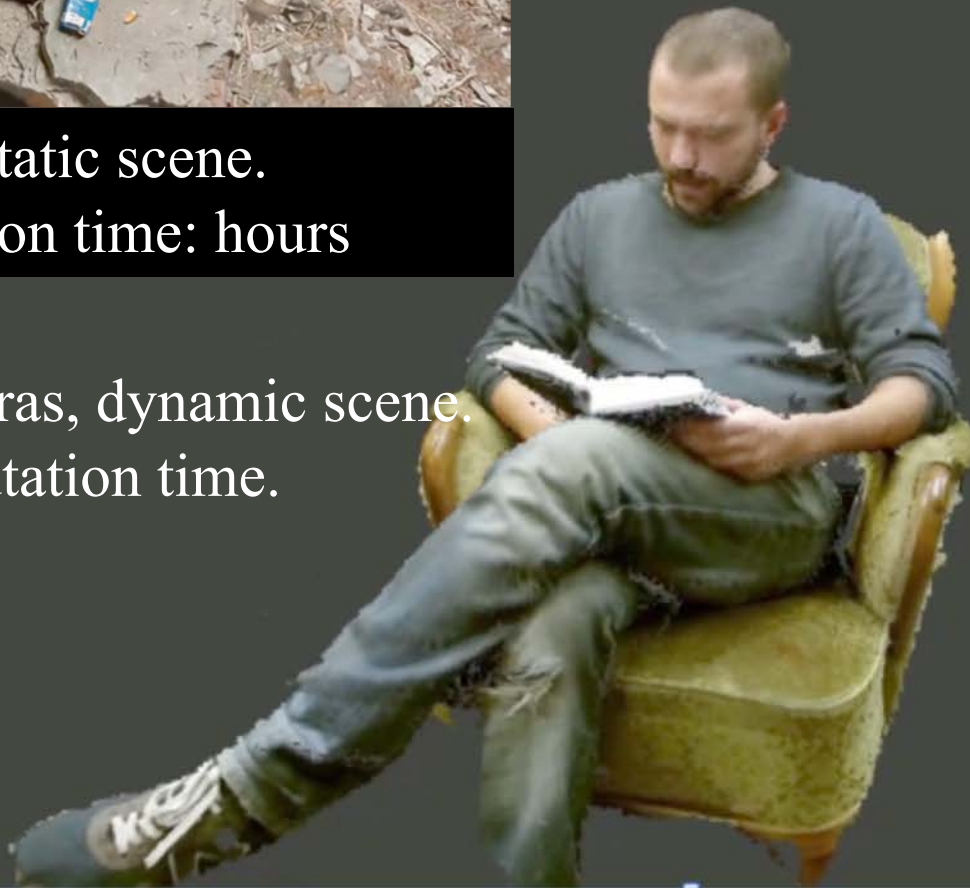
480 frames  
512<sup>3</sup> grid  
20 sec  
@24Hz

5.2 MiB  
0.9  
GiB/hour  
2.1 Mbits/s



500 photos, static scene.  
Precomputation time: hours

Here: 3 cameras, dynamic scene.  
No precomputation time.



Cameras:





Two days ago...





LG/LT: 16 cams geometry + 8 cams texture

Microsoft: ~100 cameras, triangles  
Precomputation time: ~100 hours.



But we still have no view-dependent colors.  
I.e., no reflections that change with the view position



# Deadpool (to demonstrate view-dependent reflections)



# Deadpool

(to demonstrate desired quality of FVV with view-dependent reflections)



# Framtidens mediatekniker

- Filma (4-10 kameror)
- 3D-rekonstruera varje frame
  - För varje tidssteg i filmen:
    - 3D rekonstruera scenen från kamerornas foton.
- Komprimera från TB till GB.
  - streambar över internet
- Spela upp filmen från valfri synvinkel
  - Dvs vi kan gå omkring i filmen medan den spelar.



# Framtidens mediatekniker

- Science Fiction visar vägen
  - Visar vad vi vill ha
  - Människan skaffar det hon vill ha (bland annat...)



60:ies



2000



# Star Trek - Tablets

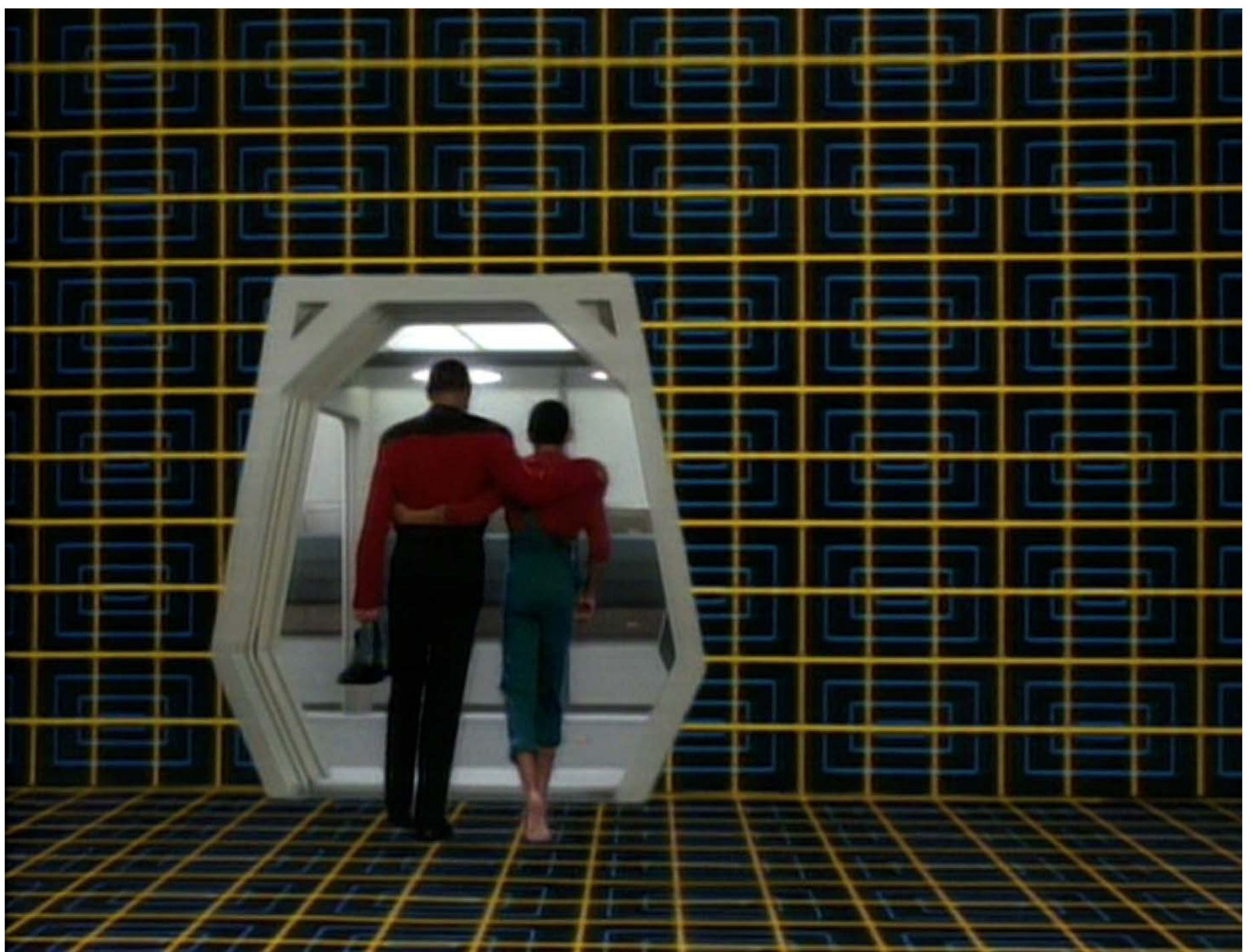
80:ies

2010

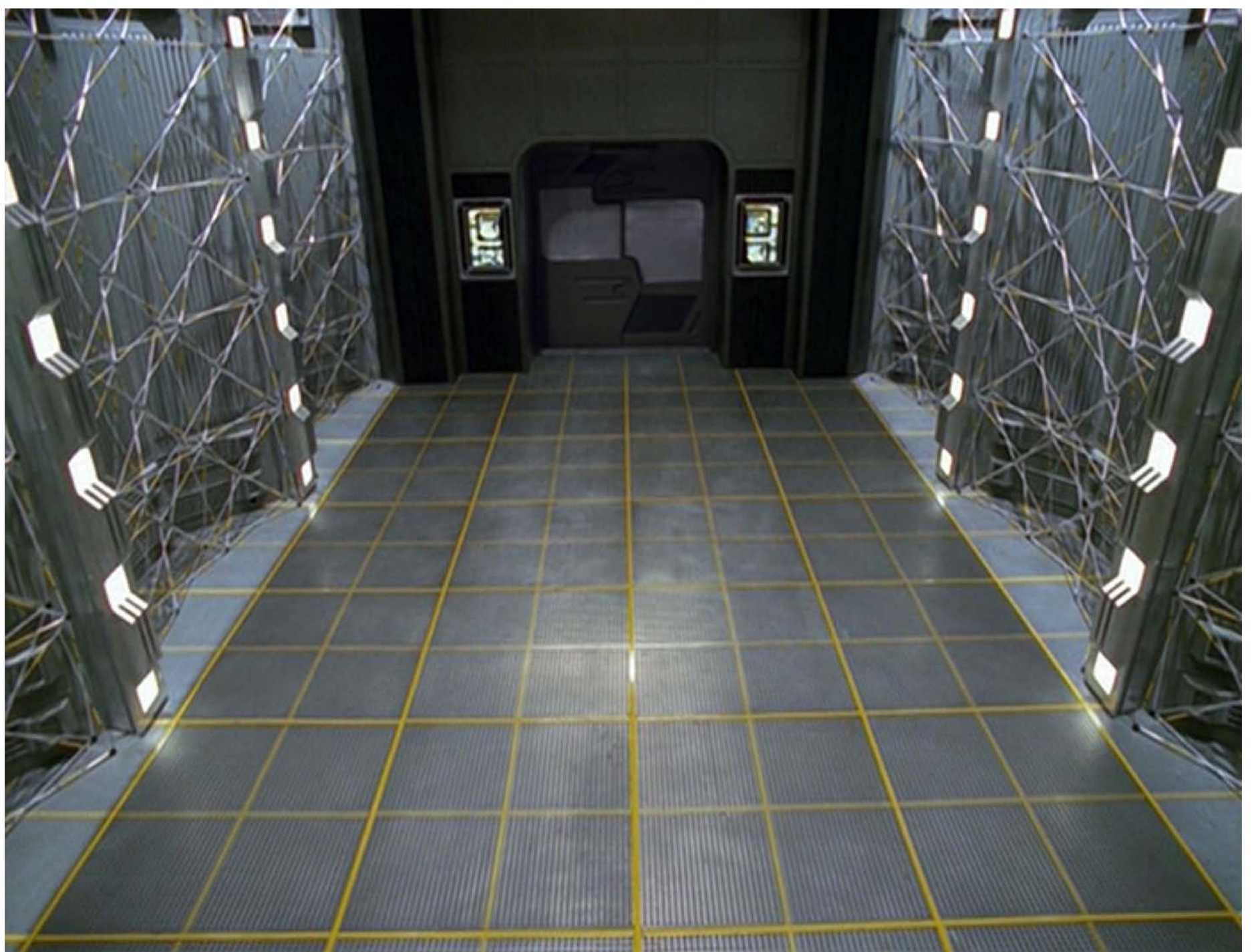


# Star Trek - Holodeck









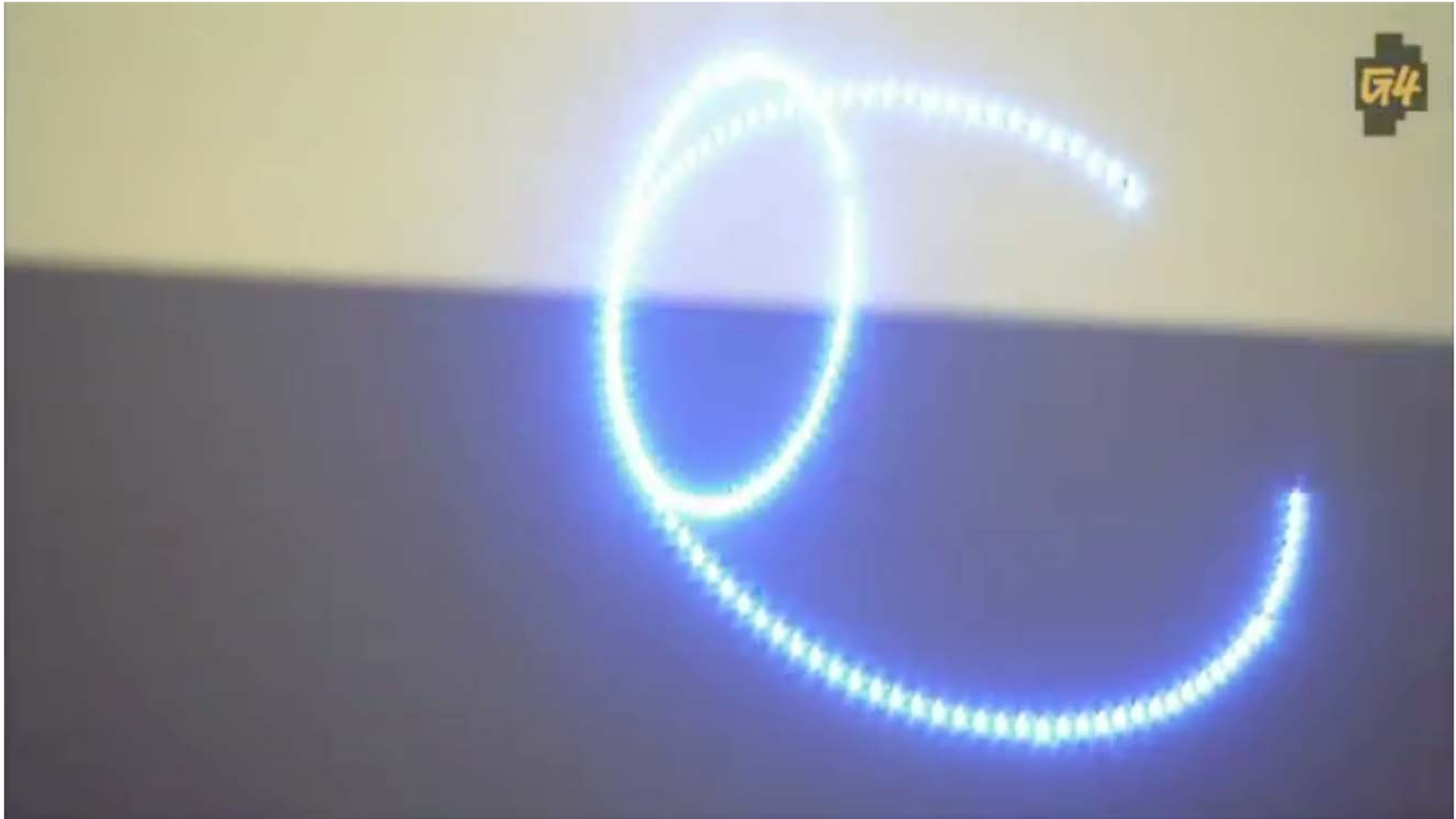
Oculus Rift



HTC Vive



# Mid air display



# Mid air display





# Mid air displays 2015



This video contains an Audio Explanation.

**Fairy Lights in Femtoseconds:**  
Aerial and Volumetric Graphics  
Rendered by Focused Femtosecond Laser  
Combined with Computational Holographic Fields

Yoichi Ochiai, Kota Kumagai, Takayuki Hoshi,  
Jun Rekimoto, Satoshi Hasegawa, Yoshio Hayasaki

0:11 / 3:19

Settings HD Full Screen

# Mid air displays 2015

This video contains  
an Audio Explanation.

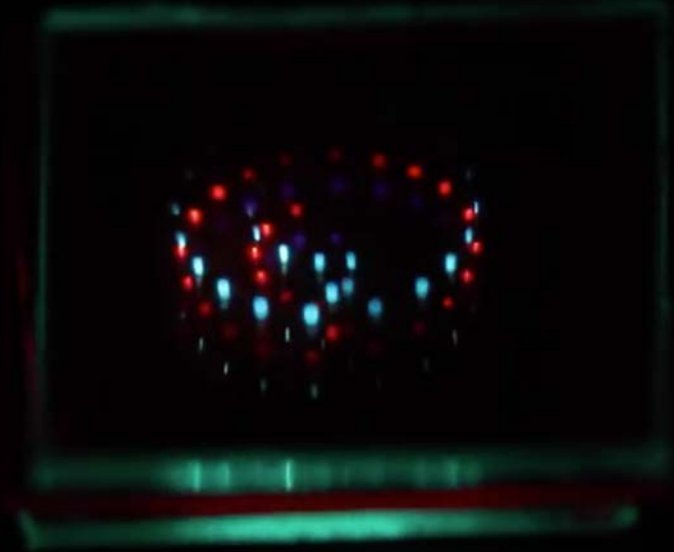


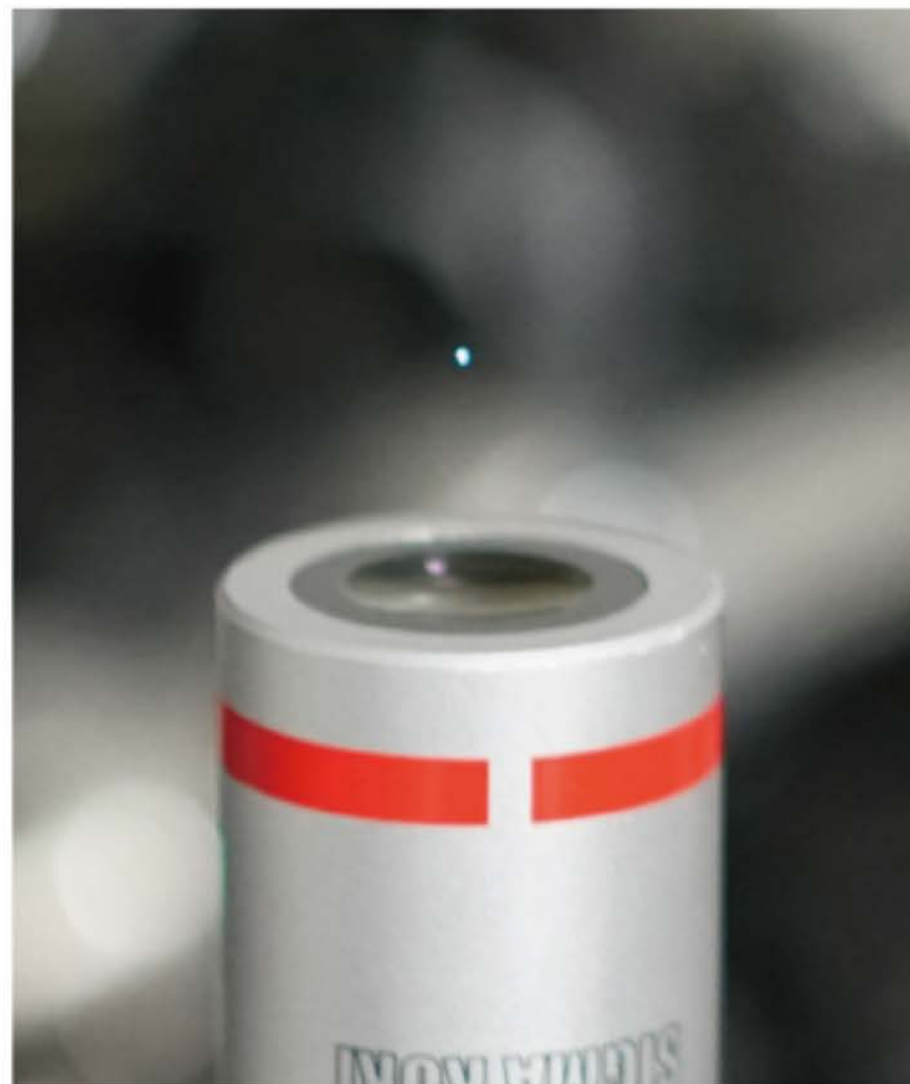
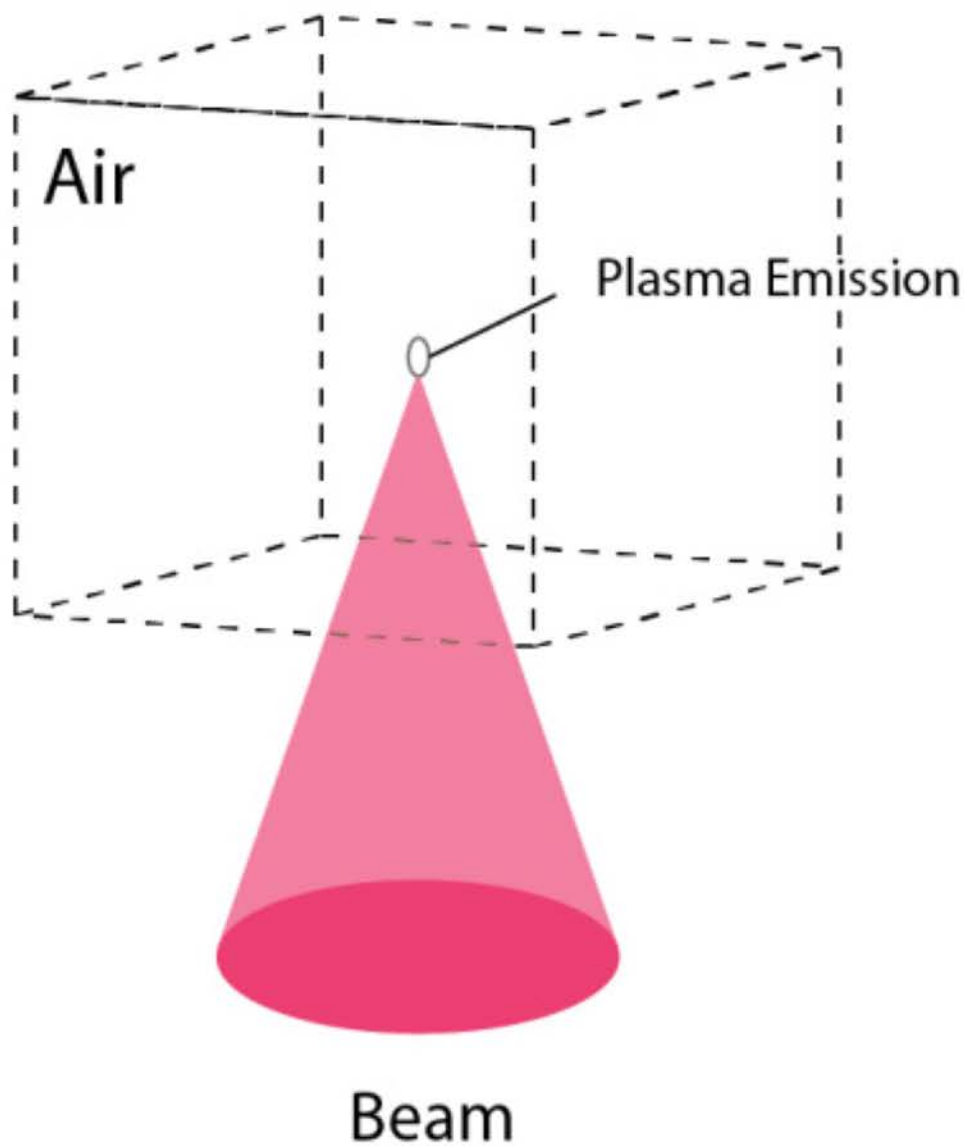
## Fairy Lights in Femtoseconds:

Aerial and Volumetric Graphics  
Rendered by Focused Femtosecond Laser  
Combined with Computational Holography

Yoichi Ochiai, Kota Kumagai, Takayuki Hoshi,  
Jun Rekimoto, Satoshi Hasegawa, Yoshio Hayashi

0:14 / 3:19





**Figure 4:** *Laser plasma induced by focused femtosecond laser.*



Rendering Volumetric Haptic Shapes in Mid-Air using Ultrasound

# Inget nytt under solen

- Titta på Automan från 1983-1984.



# Inget nytt under solen

- T



# The Future?

- Much science fiction will become possible
- We want to enter computer-generated virtual worlds
  - Holodeck (maybe in a few decades)
  - Or a plug into brain like in Matrix...
- We want computer-generated objects to enter our real world
  - 3D printers
  - Mid-air displays
  - Virtual matter (particles)

**Välkommen till TDA362 Computer Graphics, Lp2**