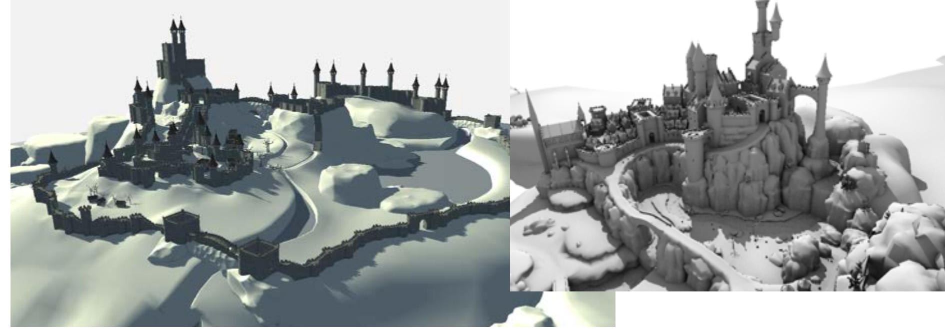
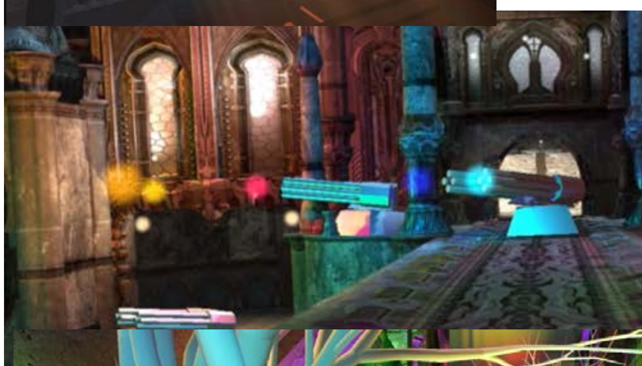




Realistic Real-time Rendering Today and in the Future

Ulf Assarsson

Chalmers University of Technology

Översikt

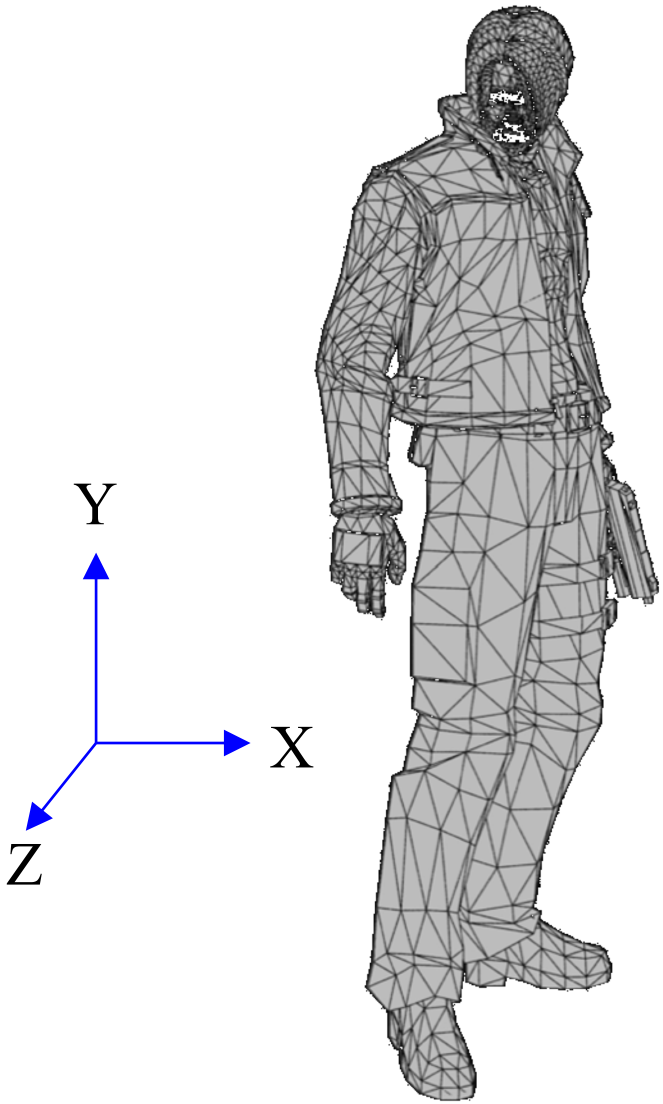
- Grafik i spel (grunderna i realtidsgrafik mha grafikkort)
- Hur man beräknar realistisk 3D grafik för film
- Hur man kan göra i spel
- Volumetric Video
- Framtidens mediatekniker

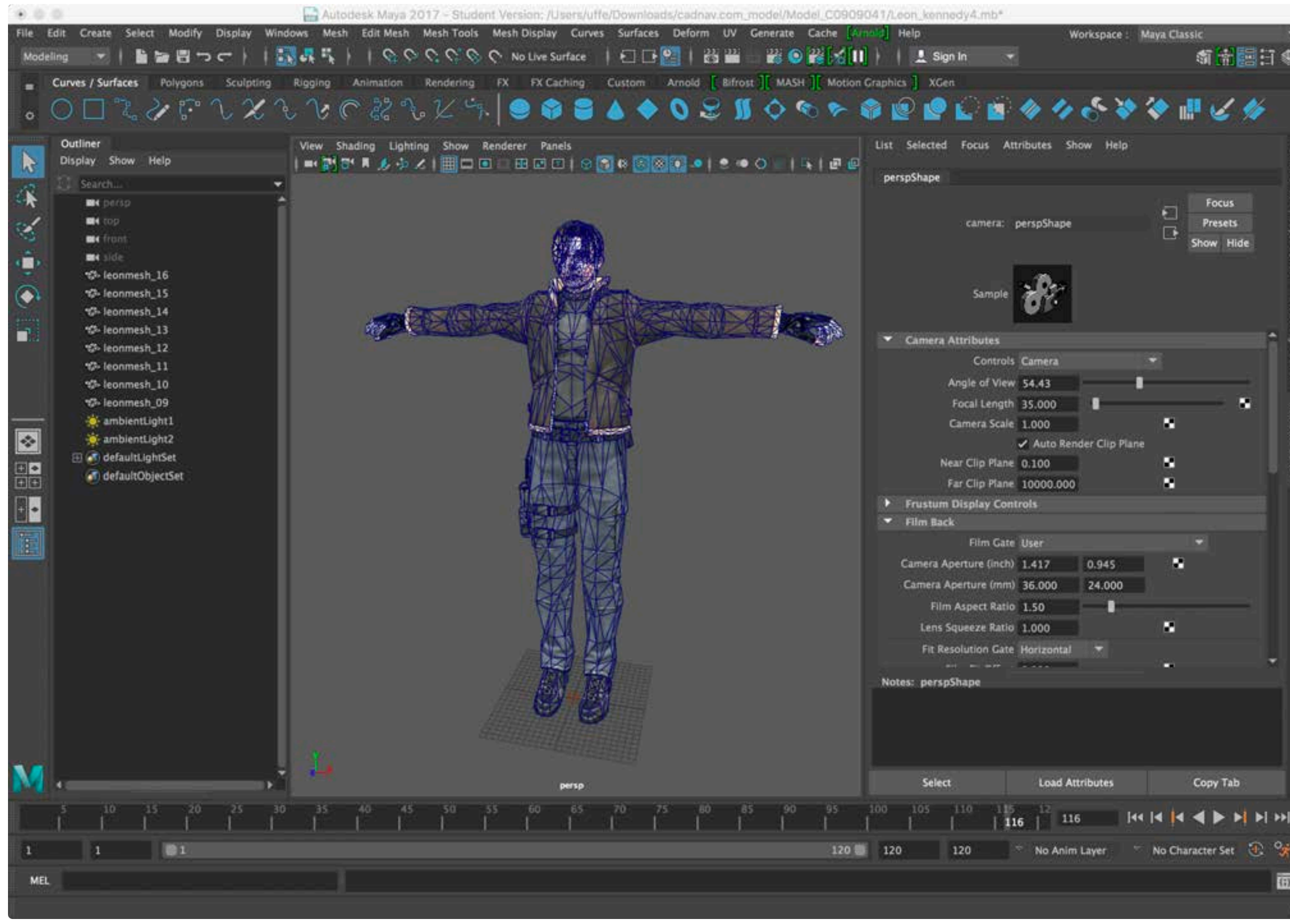
Realistic Real-time Rendering

- Today and in the Future

Triangelrendering för realtidsgrafik (t ex spel) med grafikkort

Kontext: datorgrafik, för t ex spel, film.
3D-objekt: ytor modelleras med trianglar i 3D.





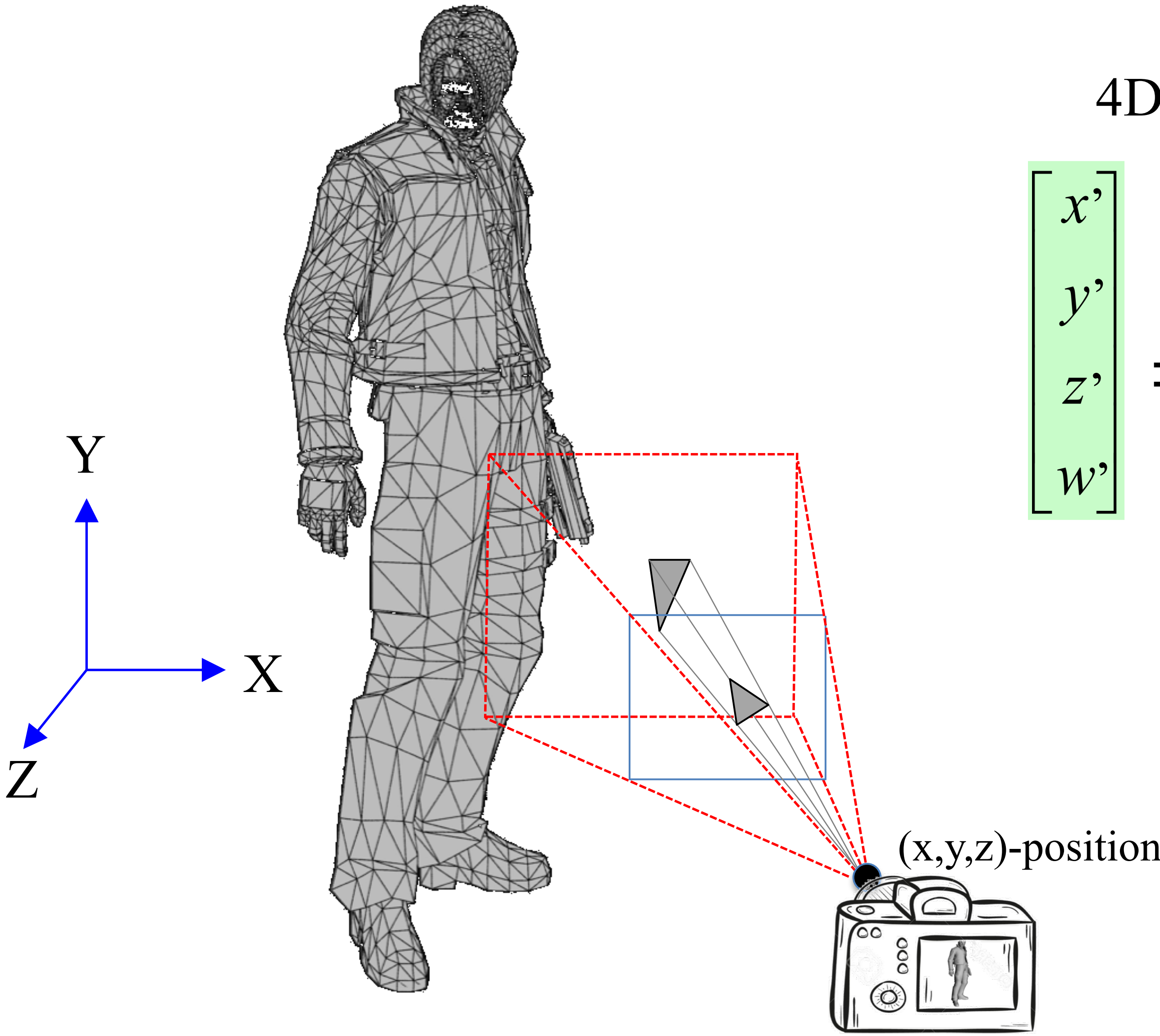
4926 triangles

Kontext: datorgrafik, för t ex spel, film.

3D-objekt: ytor modelleras med trianglar i 3D.

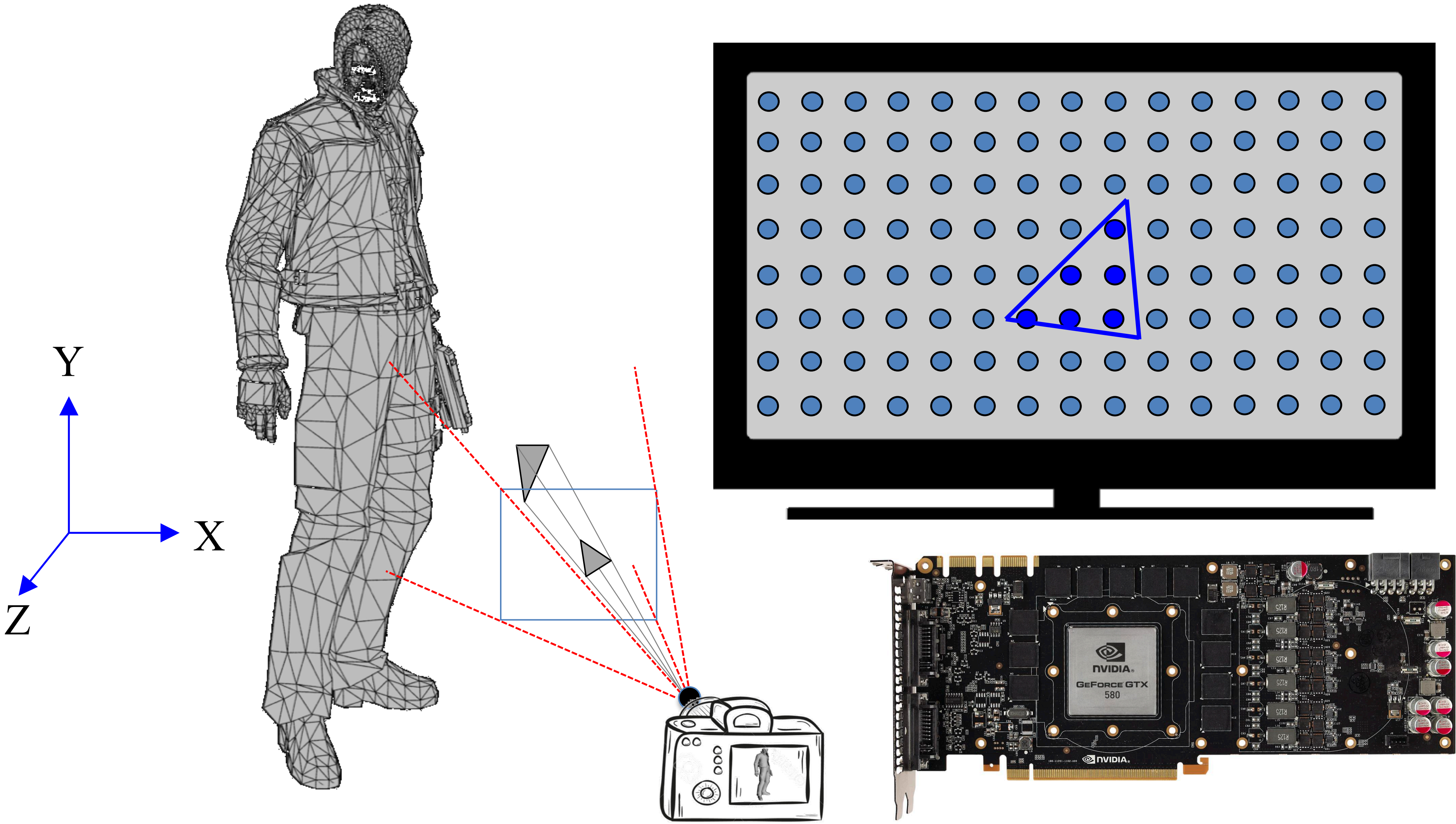
4D Matrix Multiplication

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

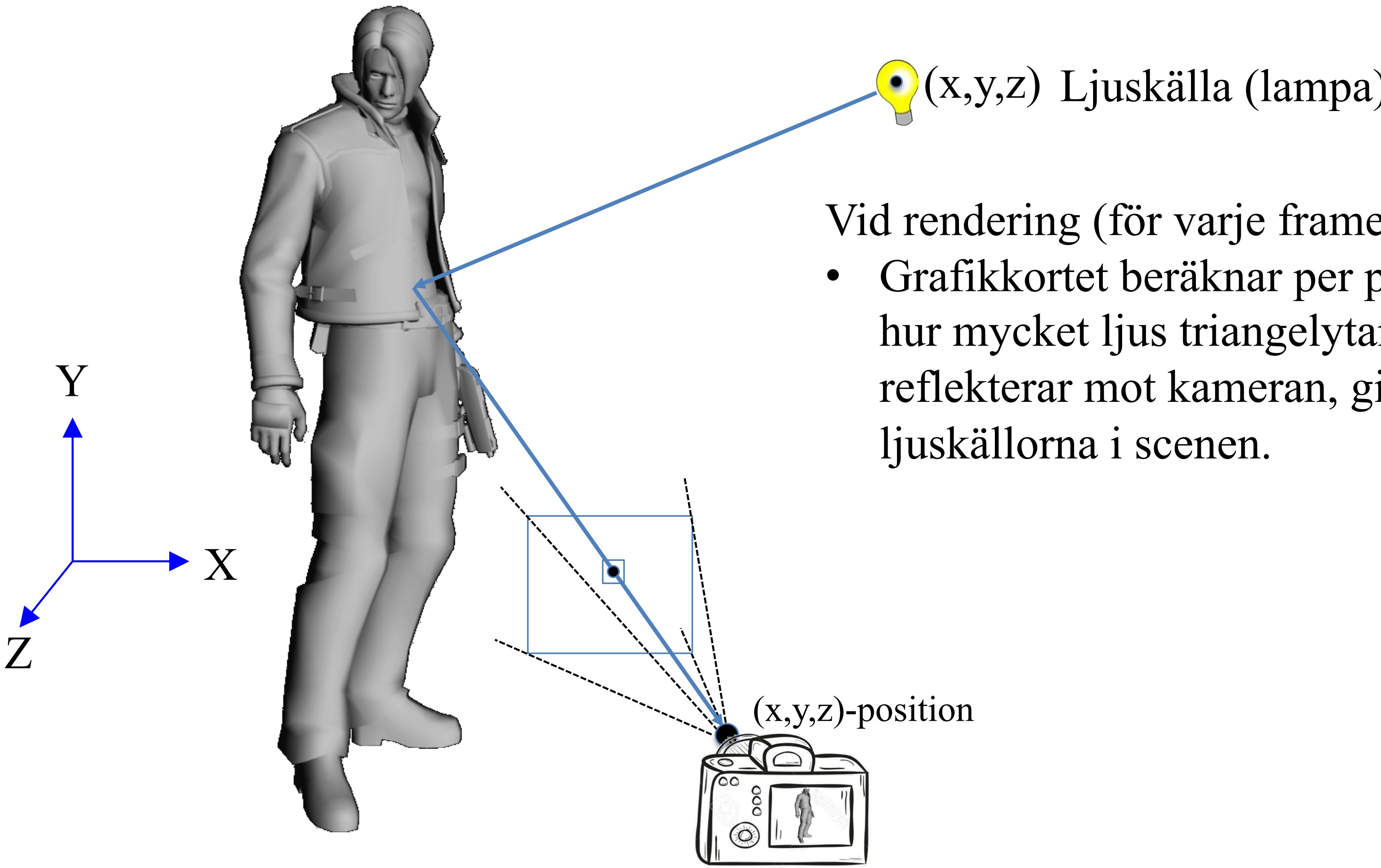


Kontext: datorgrafik, för t ex spel, film.

3D-objekt: ytor modelleras med trianglar i 3D.



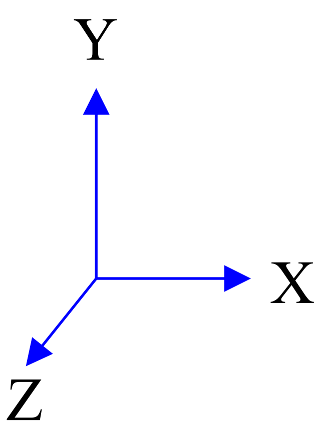
Kontext: datorgrafik, för t ex spel, film.
3D-objekt: ytor modelleras med trianglar i 3D.



Vid rendering (för varje frame):

- Grafikkortet beräknar per pixel, hur mycket ljus triangelytan där reflekterar mot kameran, givet ljuskällorna i scenen.

Kontext: datorgrafik, för t ex spel, film.
3D-objekt: ytor modelleras med trianglar i 3D.

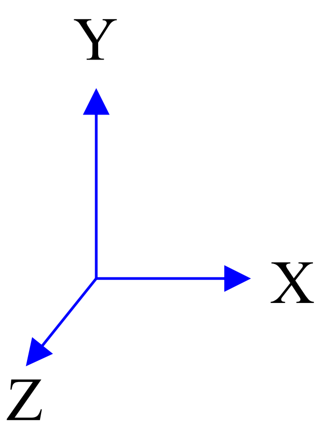


(x,y,z) Ljuskälla (lampa)

Vid rendering (för varje frame):

- Grafikkortet beräknar triangelytans **belysning** i varje pixel, givet ljuskällorna i scenen.

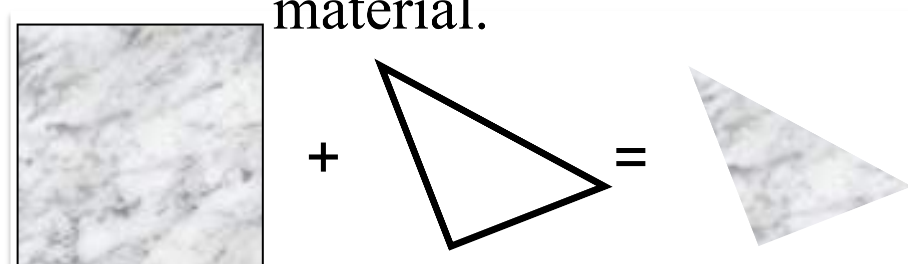
Kontext: datorgrafik, för t ex spel, film.
3D-objekt: ytor modelleras med trianglar i 3D.



(x,y,z) Ljuskälla (lampa)

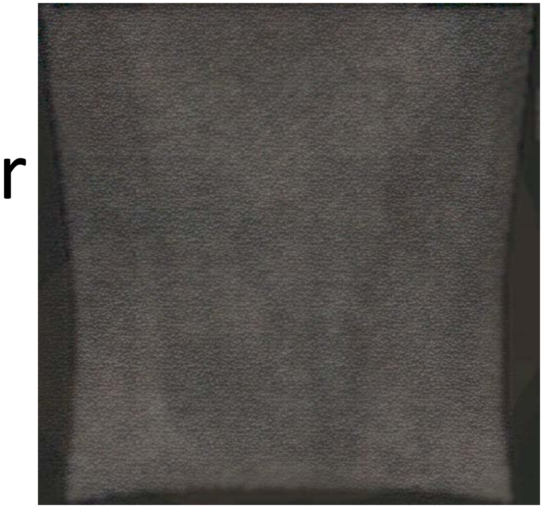
Vid rendering (för varje frame):

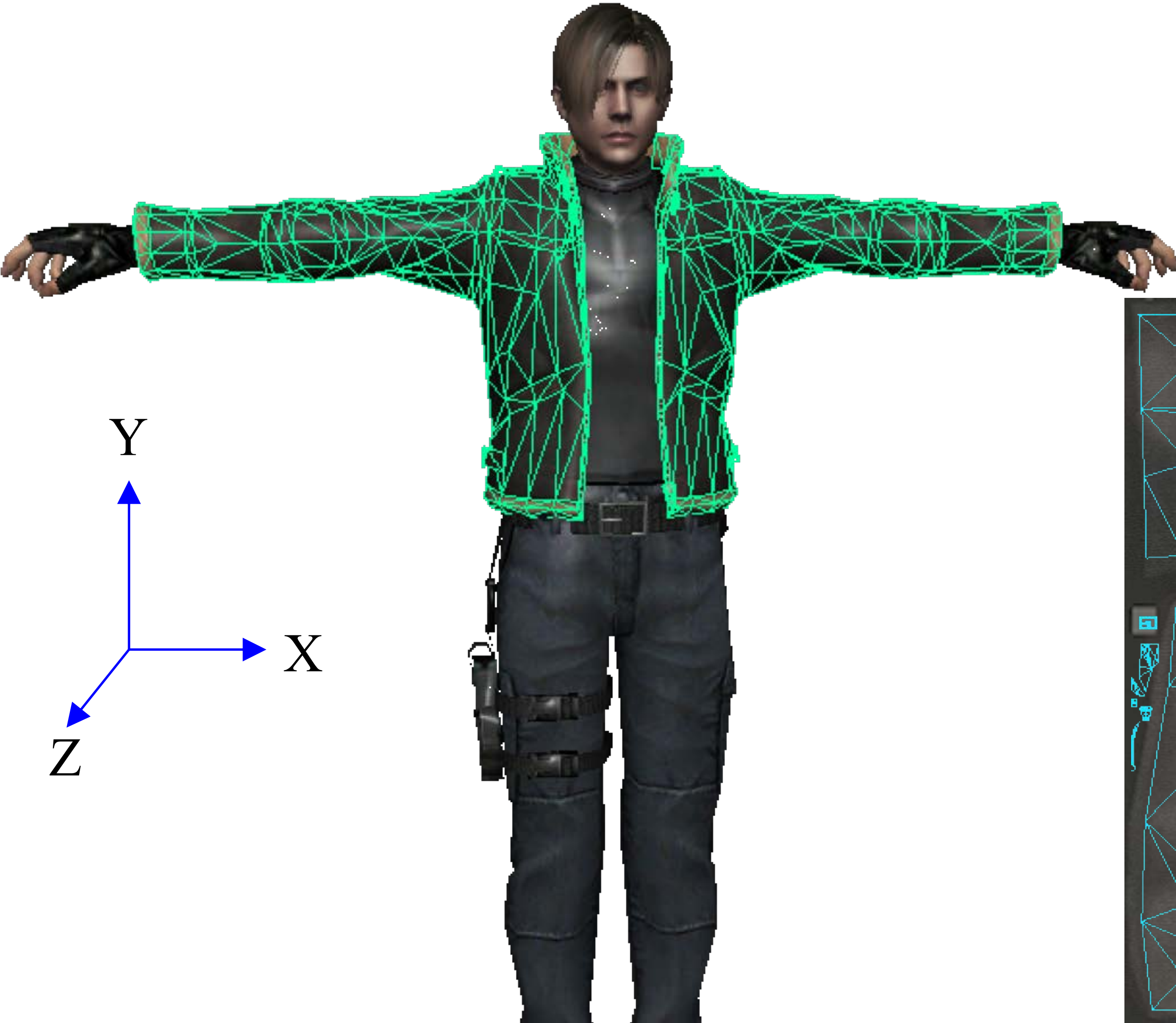
- Grafikkortet beräknar triangelytans **belysning** i varje pixel, givet ljuskällorna i scenen.
- och lägger även på **texturer** (=bilder) på trianglarna (modulerade med ljusintensiteten) för att simulera ytdetaljer och olika material.

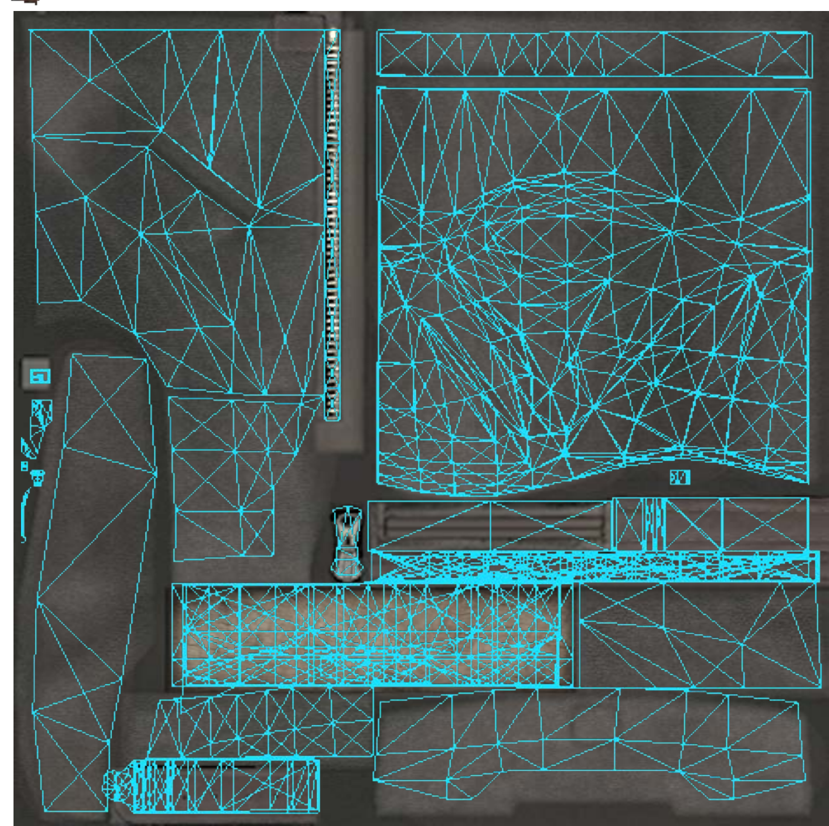


Kontext: datorgrafik, för t ex spel, film.

3D-objekt: ytor modelleras med trianglar

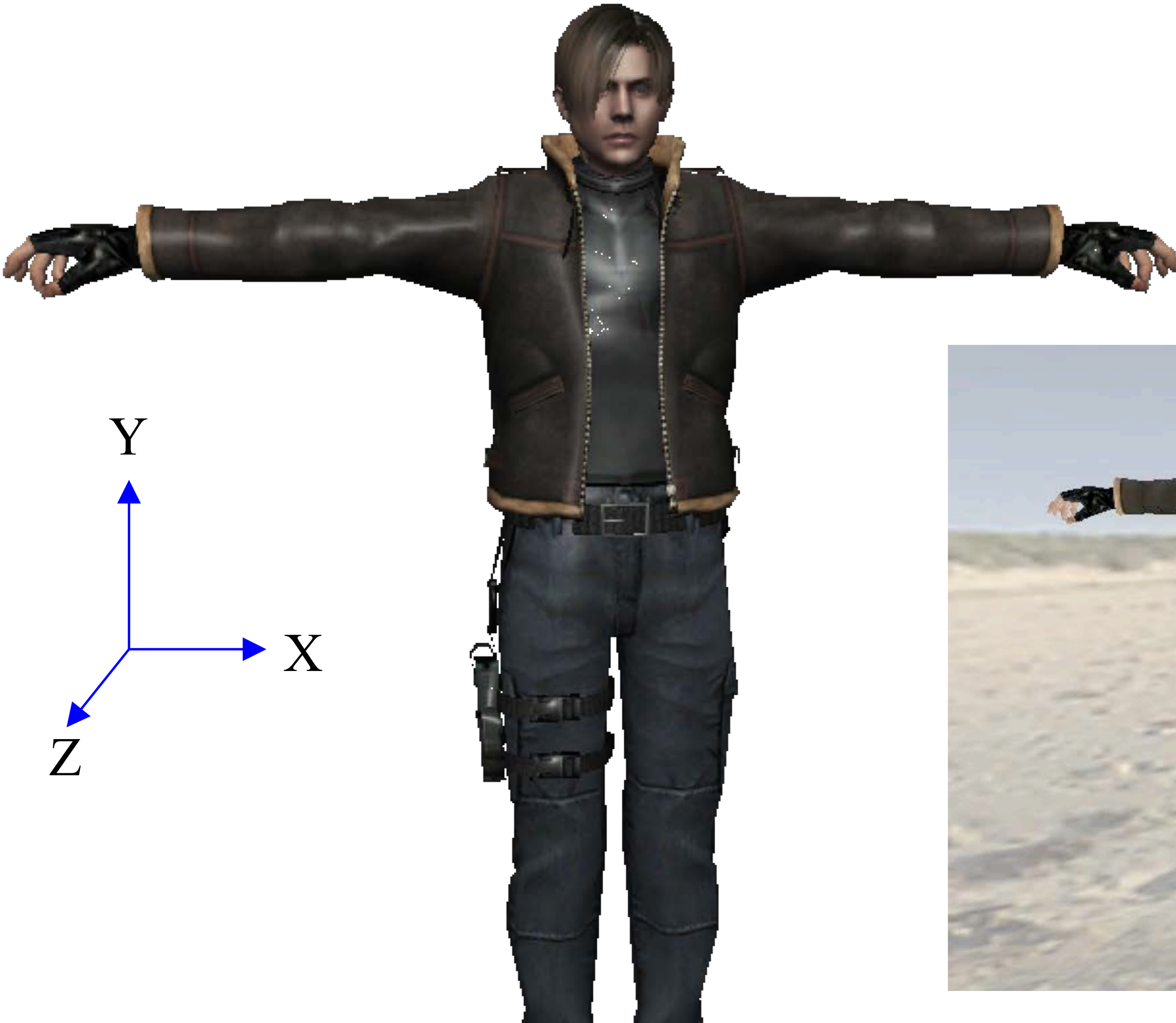




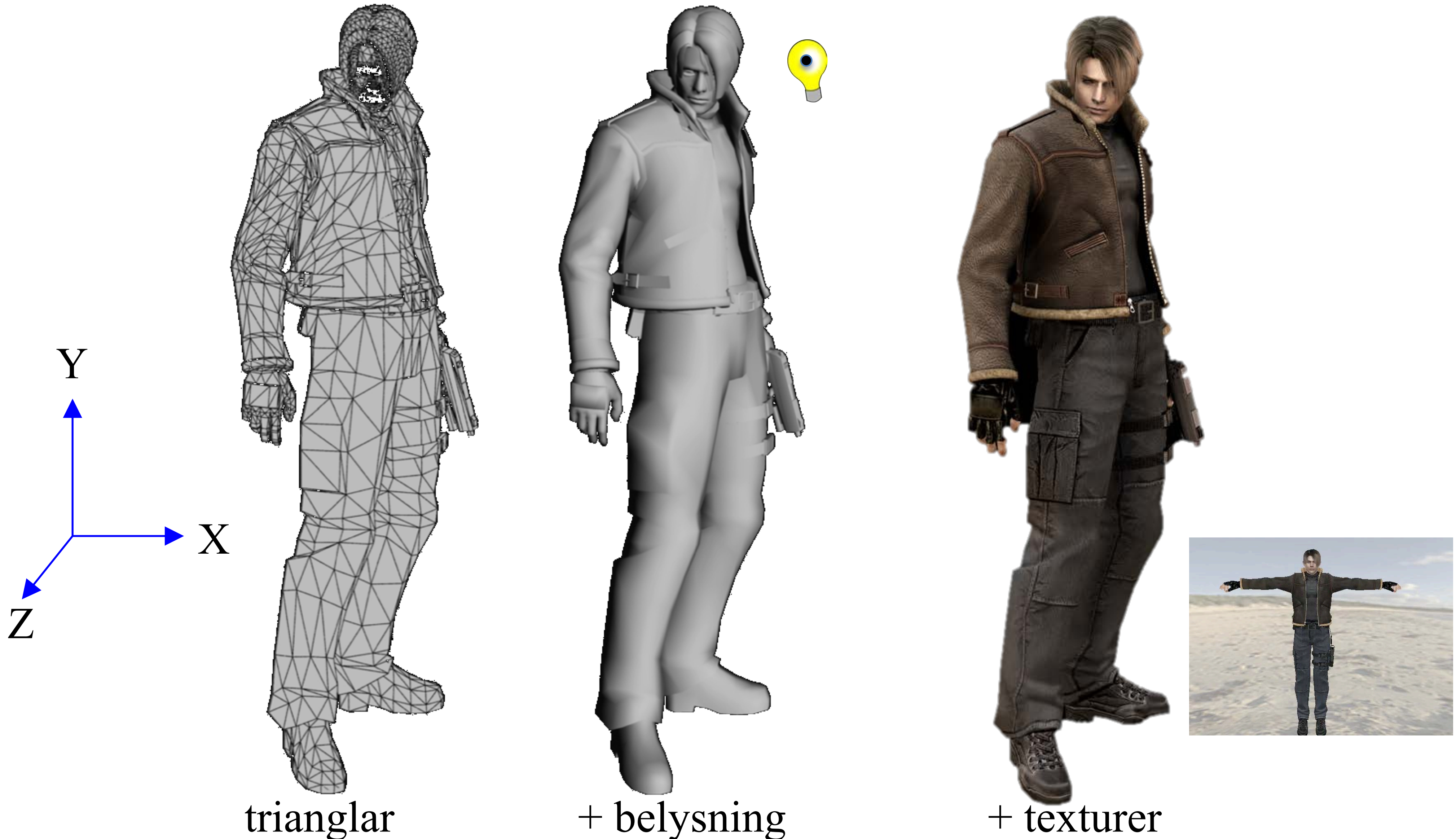


Kontext: datorgrafik, för t ex spel, film.

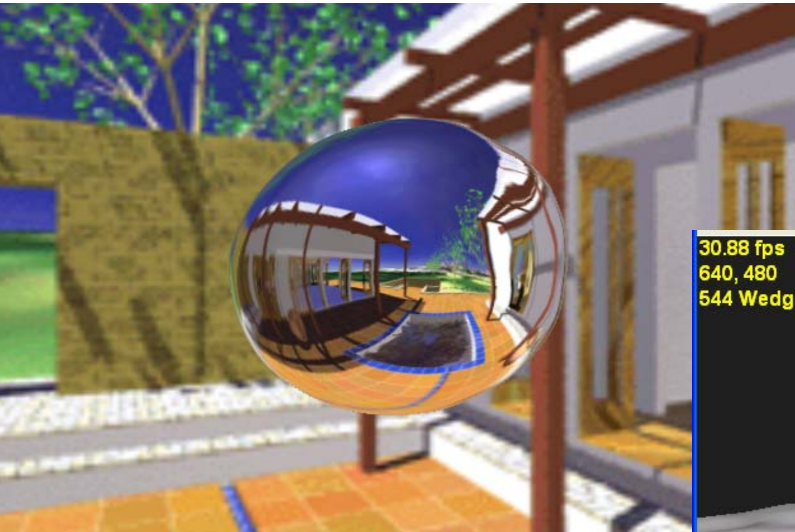
3D-objekt: ytor modelleras med trianglar i 3D.



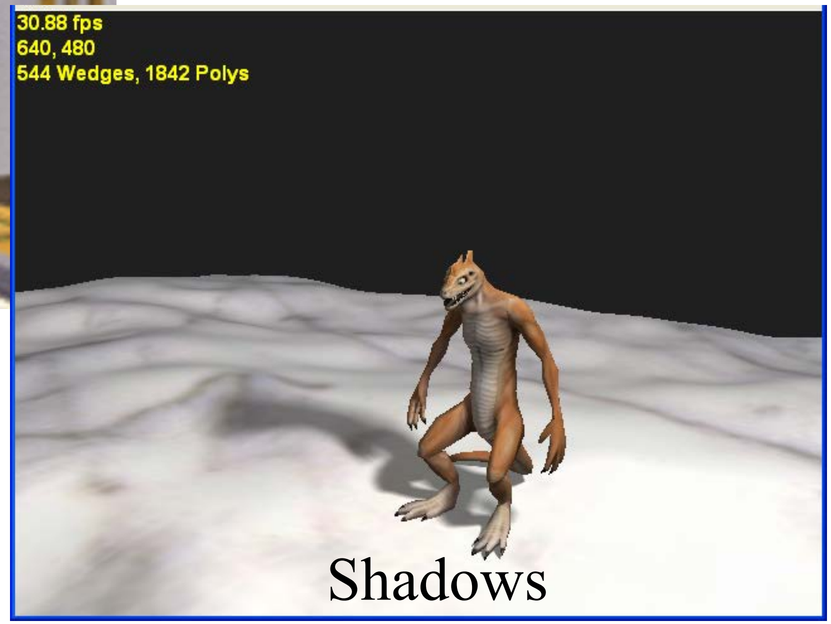
Kontext: datorgrafik, för t ex spel, film.
3D-objekt: ytor modelleras med trianglar i 3D.



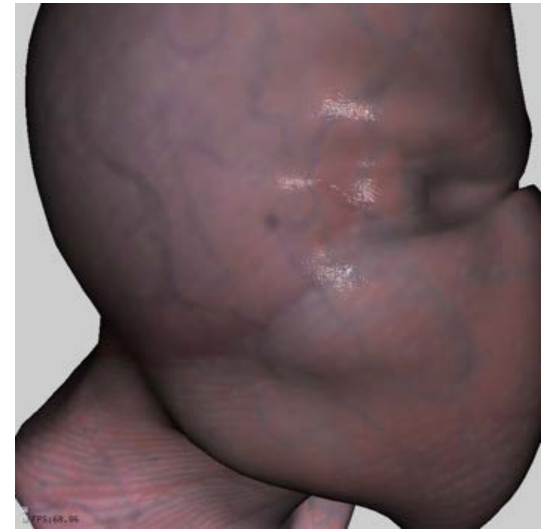
More



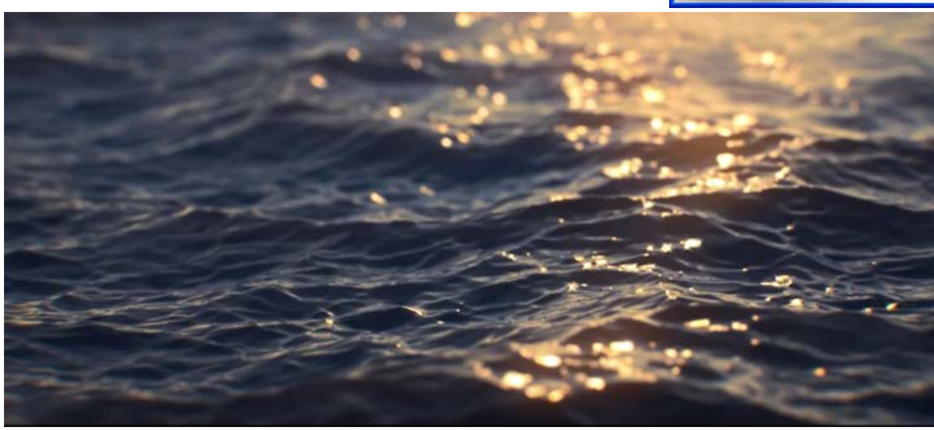
Reflections



Shadows



Materials



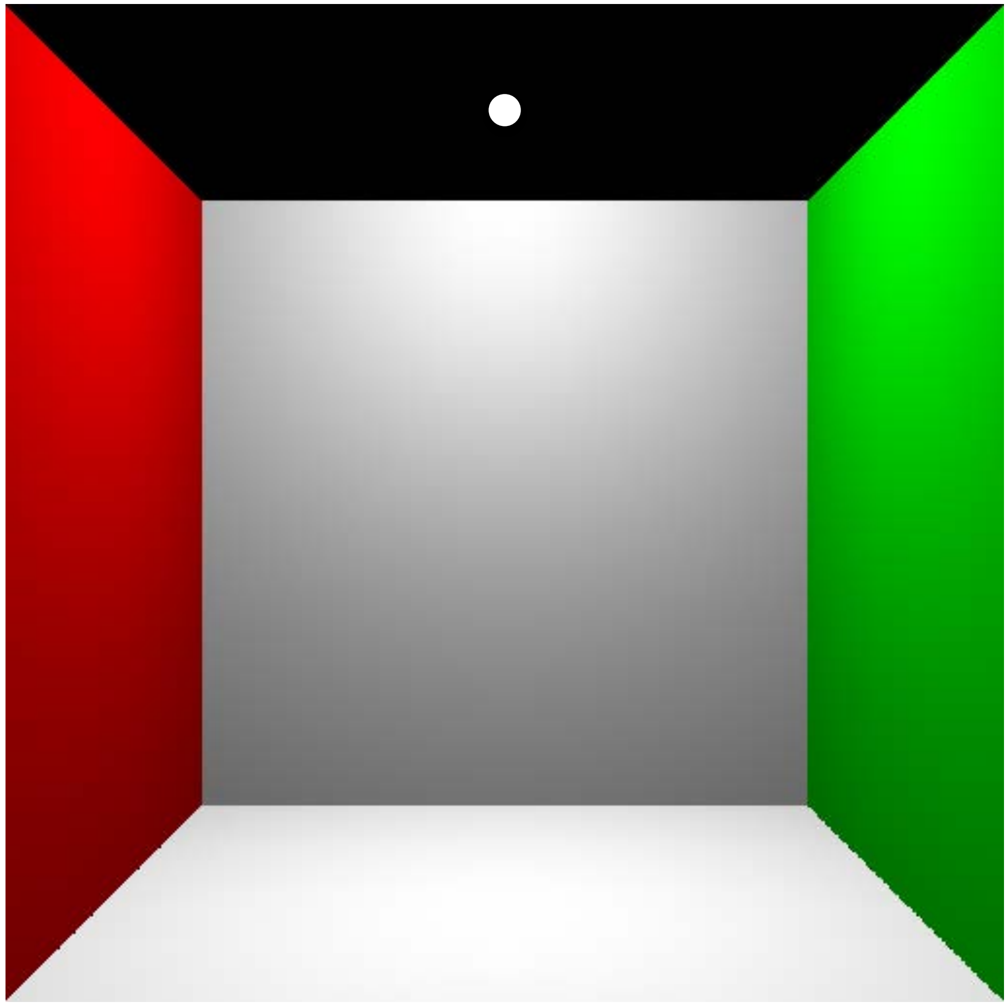
Water

Airlight

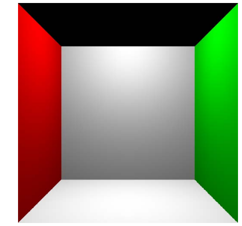
Fire

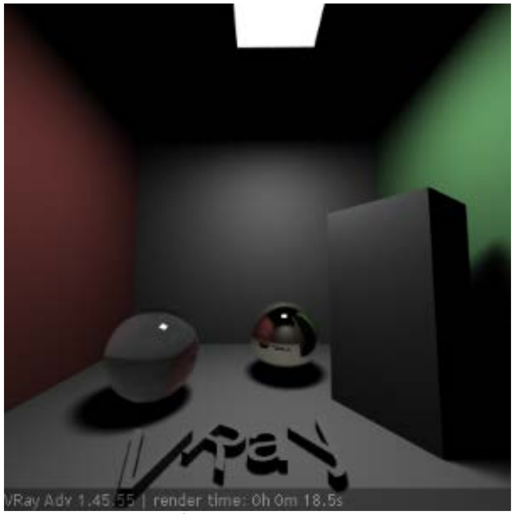
Light Bounces

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

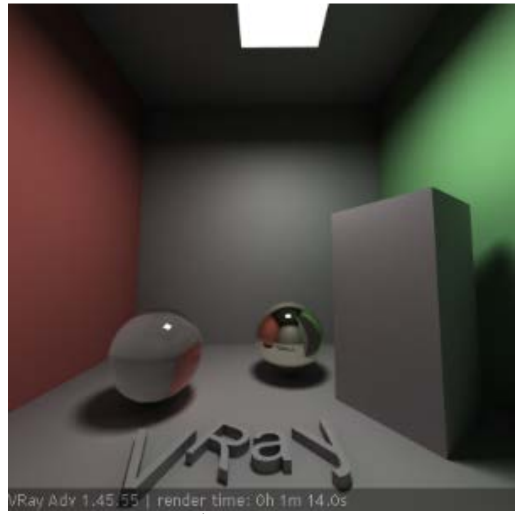


Light Bounces

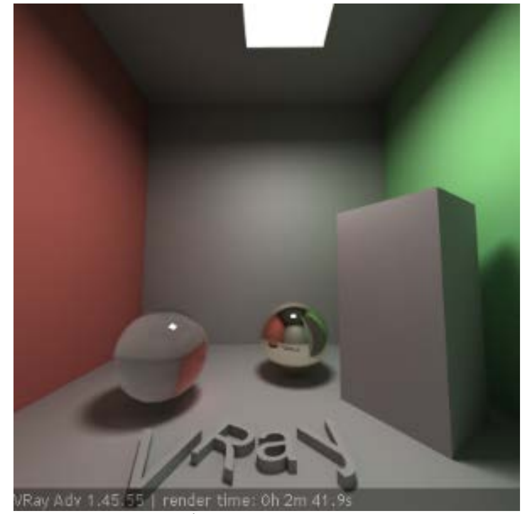




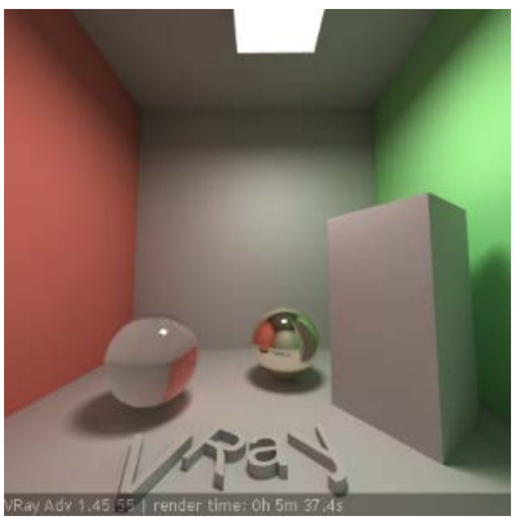
0 bounce



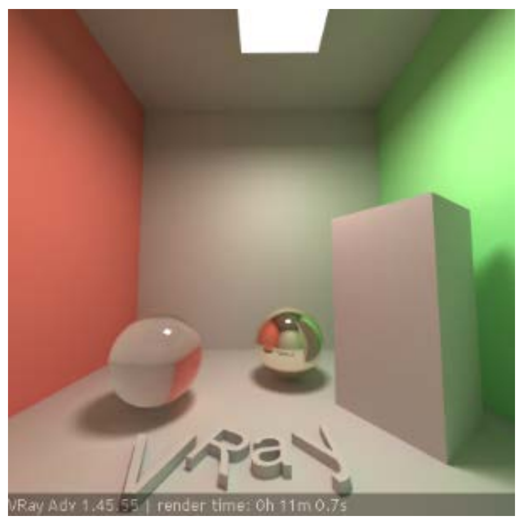
1 bounces



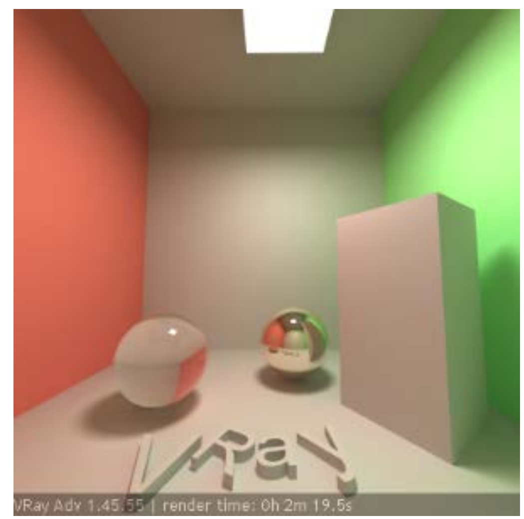
2 bounces



4 bounces



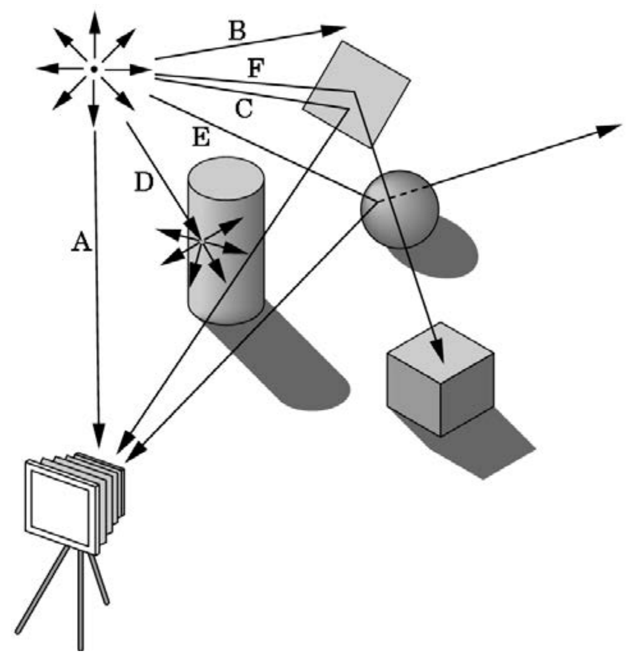
8 bounces



infinite bounces

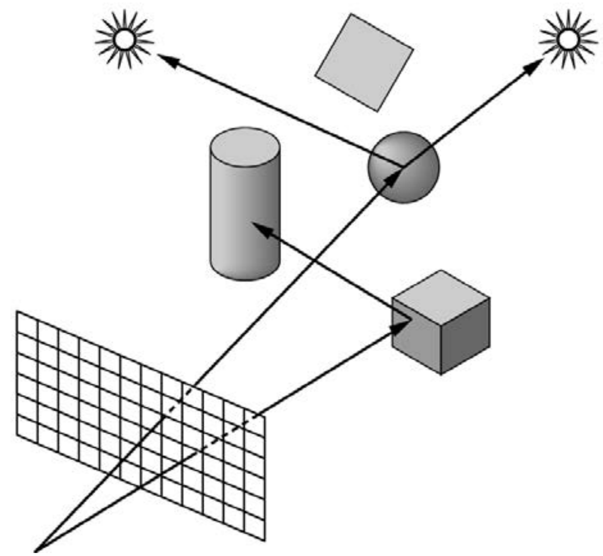
Tracing Photons

One way to form an image is to follow rays of light from a point source finding which rays enter the lens of the camera. However, each ray of light may have multiple interactions with objects before being absorbed or going to infinity.



Other Physical Approaches

- **Ray tracing**: follow rays of light backwards, from camera through each pixel, until they either are absorbed by objects or go off to infinity
  - Can handle global effects
    - Multiple reflections
    - Translucent objects
  - Faster but still rather slow
    - Can easily miss important directions e.g., caustics.



The Problem of Computer Graphics

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

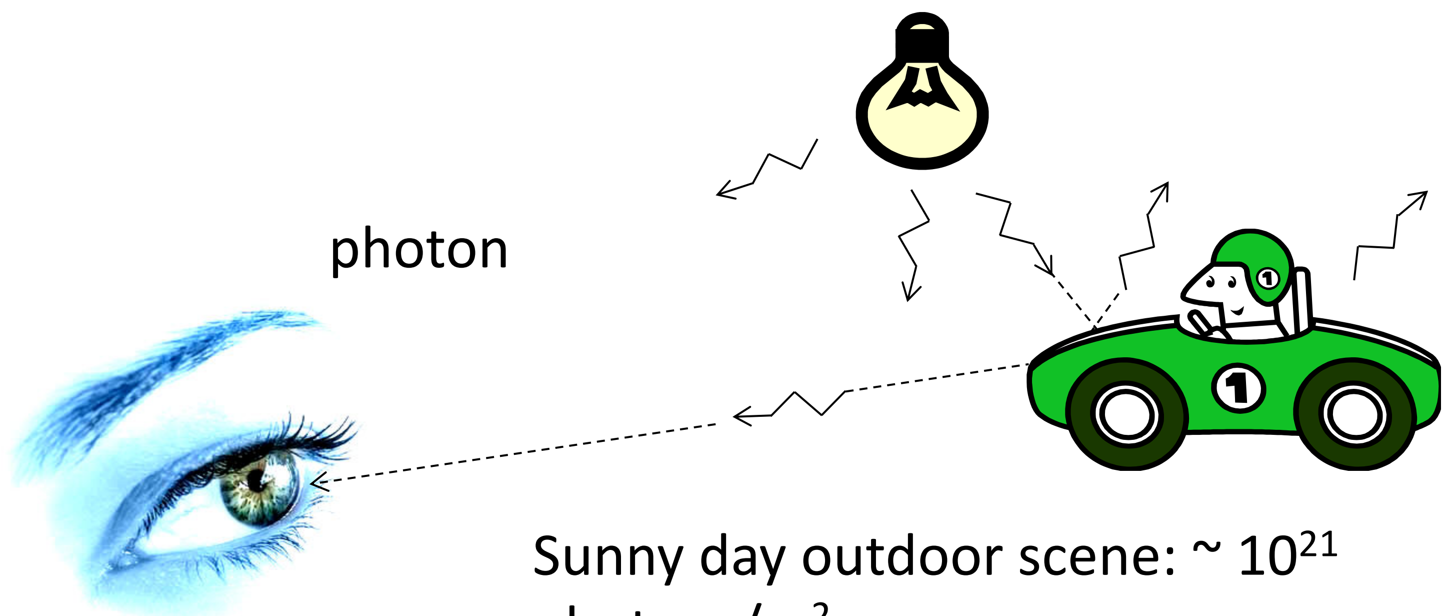
Probably not...

~20 to ~10^{15} 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: ~20 photons/s to ~ 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.



Sunny day outdoor scene: ~ 10^{21} photons/m^2s

Facts:

- Eye sensitivity: ~20 to ~10^{15} photons/s
- Sensitivity is logarithmic
  - i.e., difference between 100 or 200 photons is as noticable as for 10^{10} or $2*10^{10}$ photons
- ~ 10^{21} photons/m^2s
- 1 photon costs ~10.000 cycles

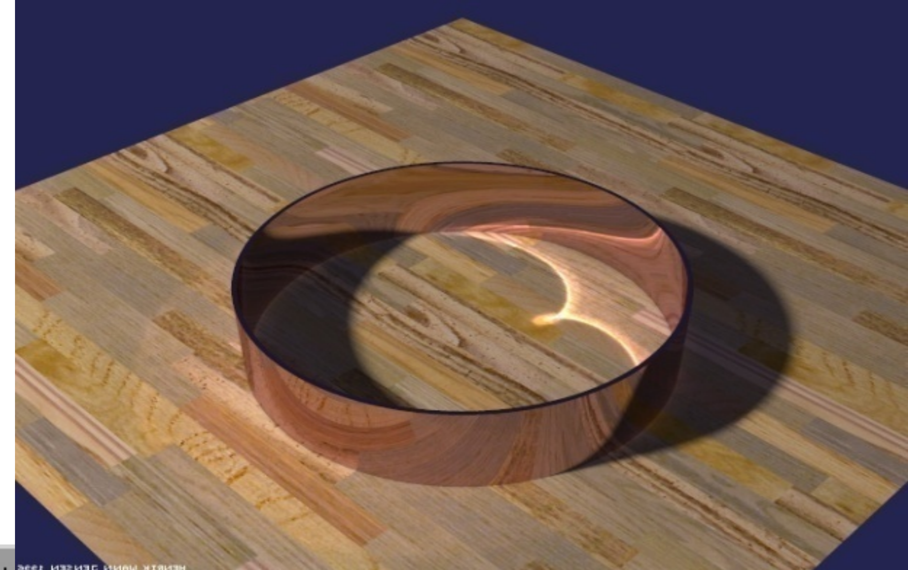
10 billion years per square meter for 1 computer

Solutions

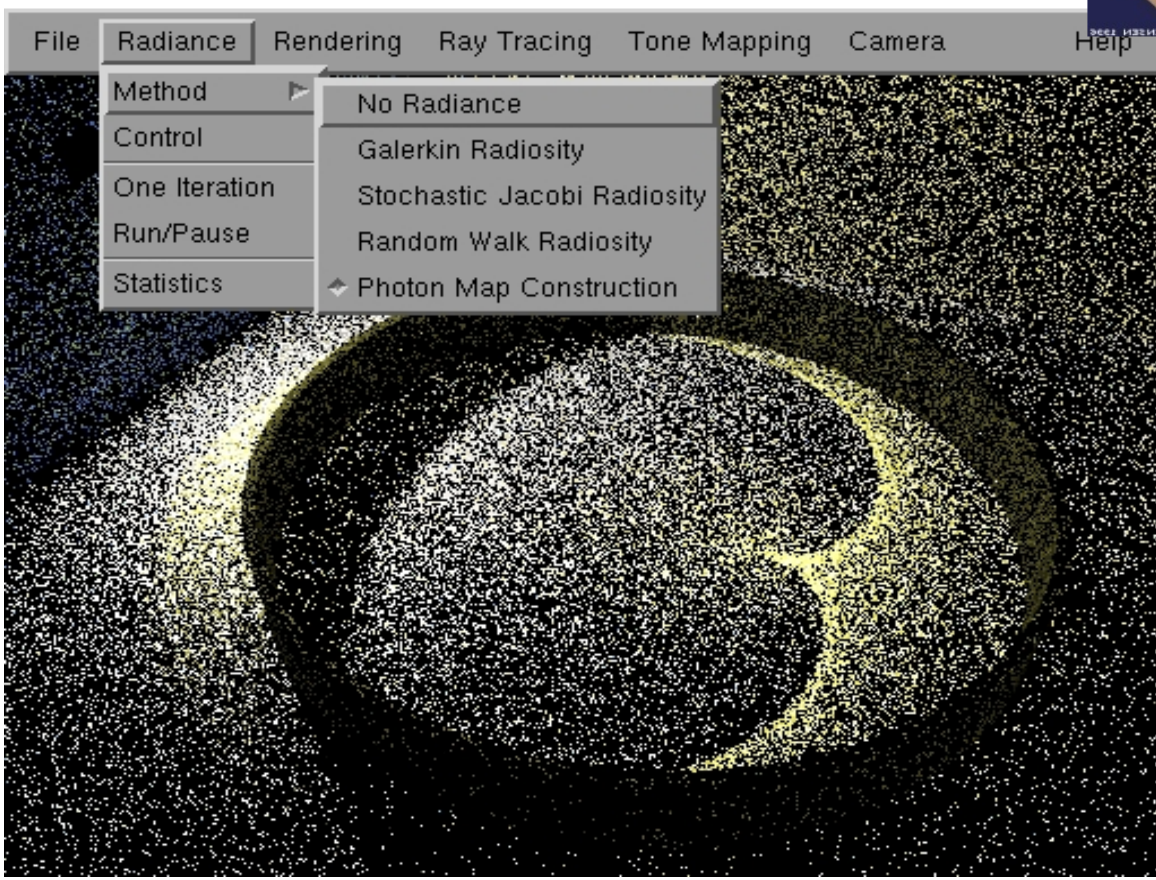
For games: Smart specialized algorithms. And cheat, cheat, cheat…. as long as it is not too noticable

For movies: we typically trace ~100M – 10B photon bounces (and also cheat).

Tracing Photons



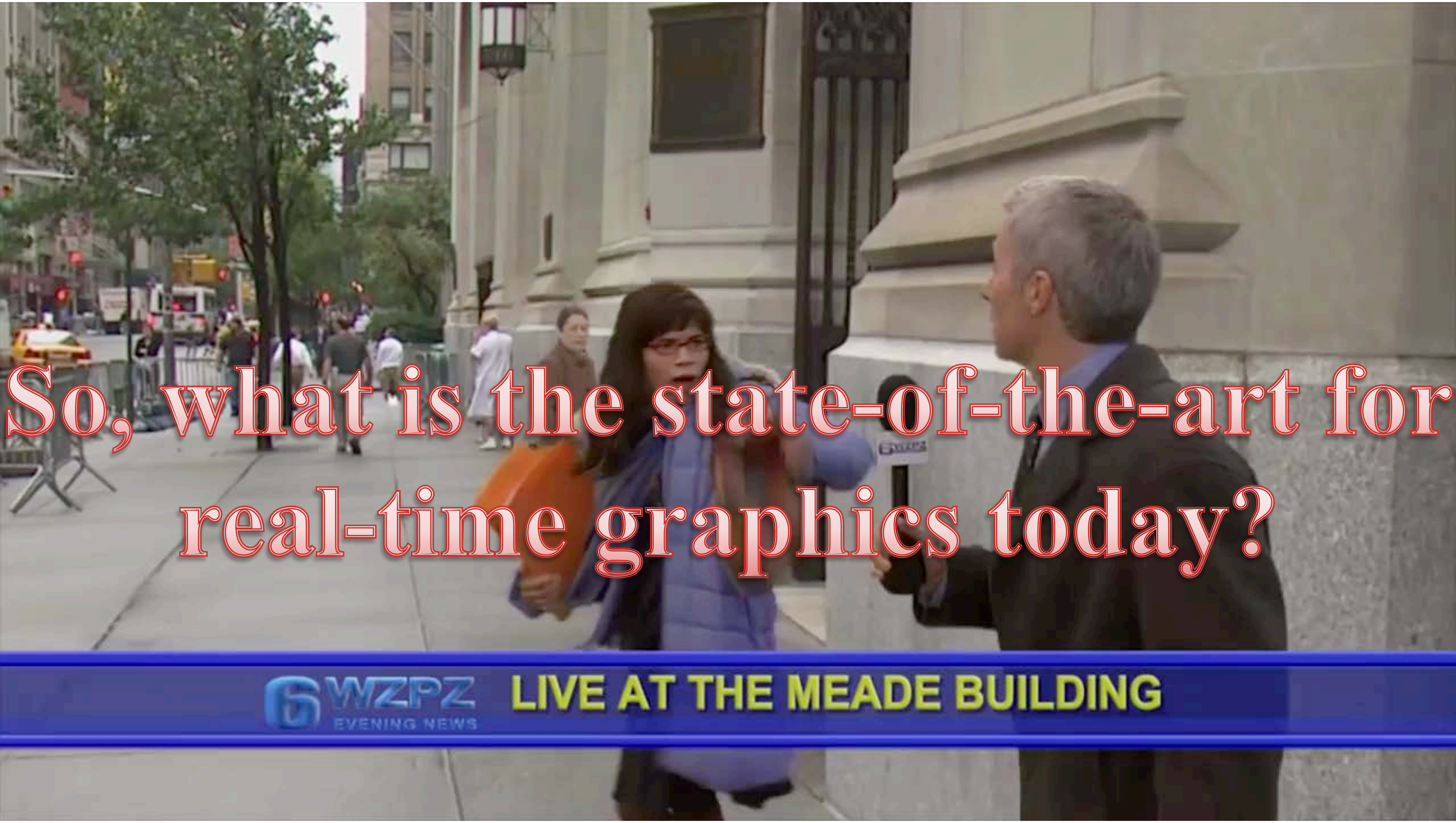
- Stored photons displayed:



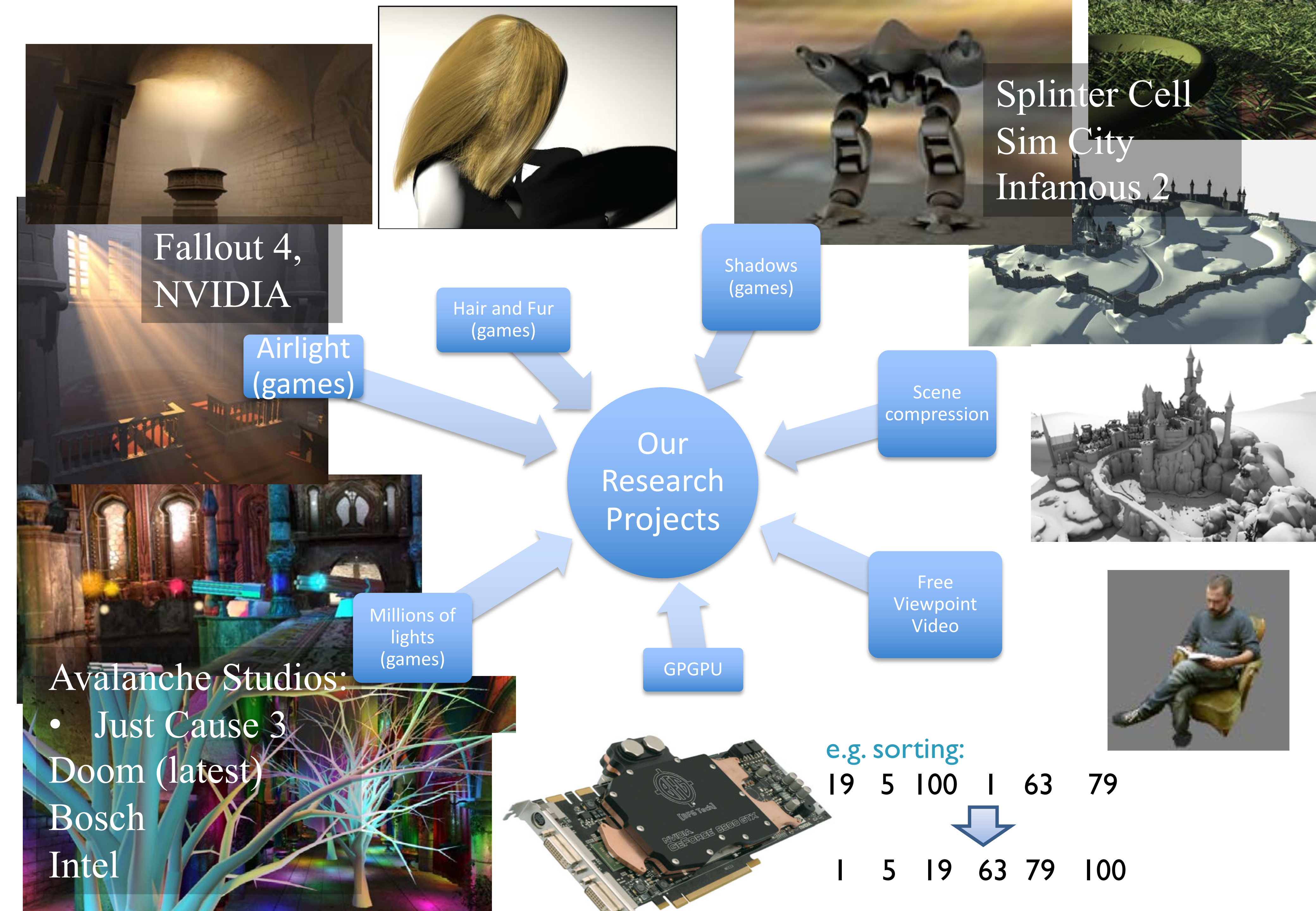
Typically: 100M – 10B photon bounces

- ~1M pixels
- ~100 rays per pixel
- ~10 bounces per ray.

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



Beäst + Unreal Engine

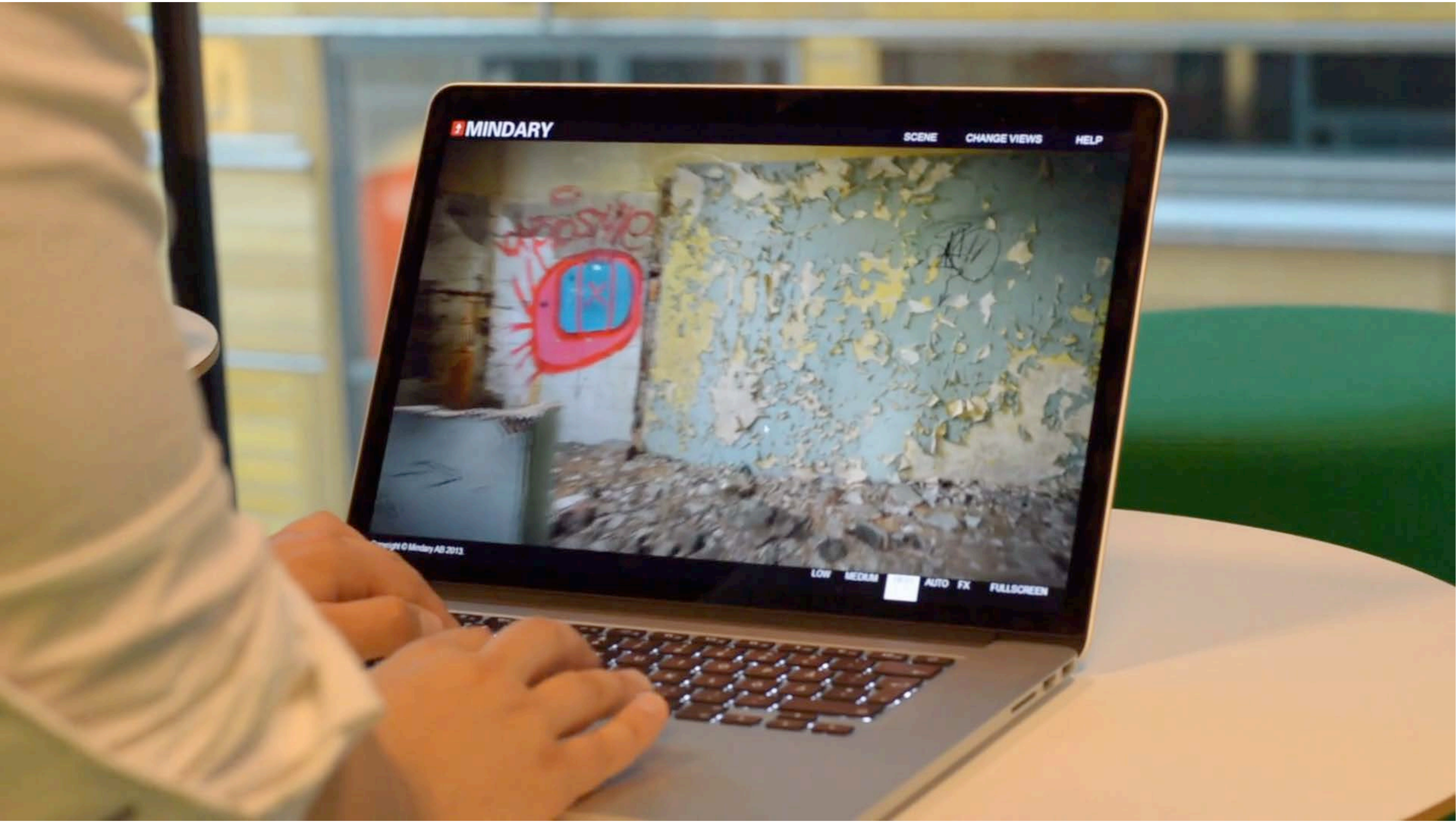

Fallout 4,
NVIDIA
Airlight (games)
Hair and Fur (games)
Shadows (games)
Splinter Cell
Sim City
Infamous 2
Scene compression
Our Research Projects
Free Viewpoint Video
GPGPU
Millions of lights (games)
Avalanche Studios:
• Just Cause 3
Doom (latest)
Bosch
Intel
e.g. sorting:
19 5 100 1 63 79
1 5 19 63 79 100

Mix of graphics and photographing

Textures from photographs

Mix of graphics and photographing

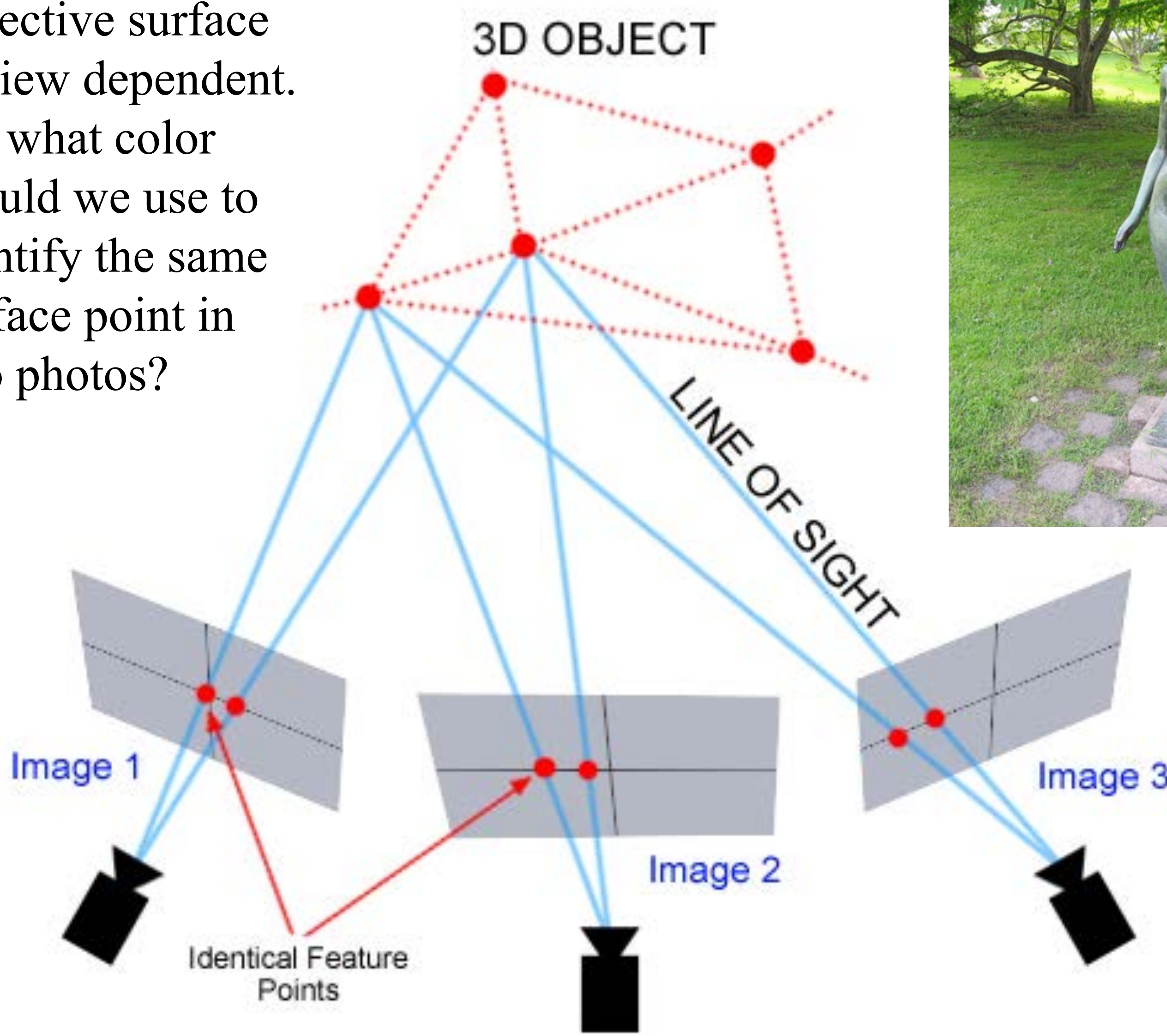
- Man kan använda fotografier som texturer.
- Men… Istället för att modellera 3D-grafik och datorberäkna realistiskt utseende:
  - Fotografera/filma verkligheten och konvertera den till 3D-grafik.
    - Enklare, billigare, snabbare (men ger för många triangular)
    - Ger automatiskt
      - texturer från fotografier
      - 3D-koordinaterna för triangelhörnen
  - Kallas “photogrammetry”:
    - The art, science and technology of obtaining reliable information about physical objects and the environment through the process of recording, measuring and interpreting photographic images

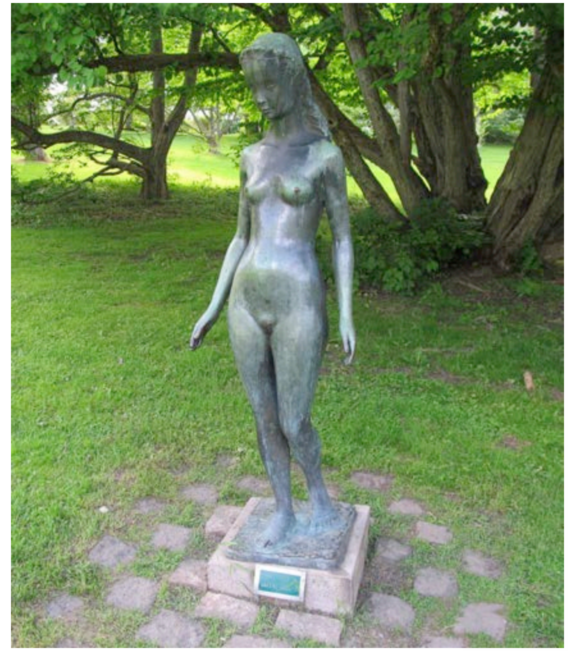


http://www.cse.chalmers.se/~uffe/mindary/demo/v2.html

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?





Scanned result

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 - highly reflective surfaces (e.g. mirror) not at all.

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

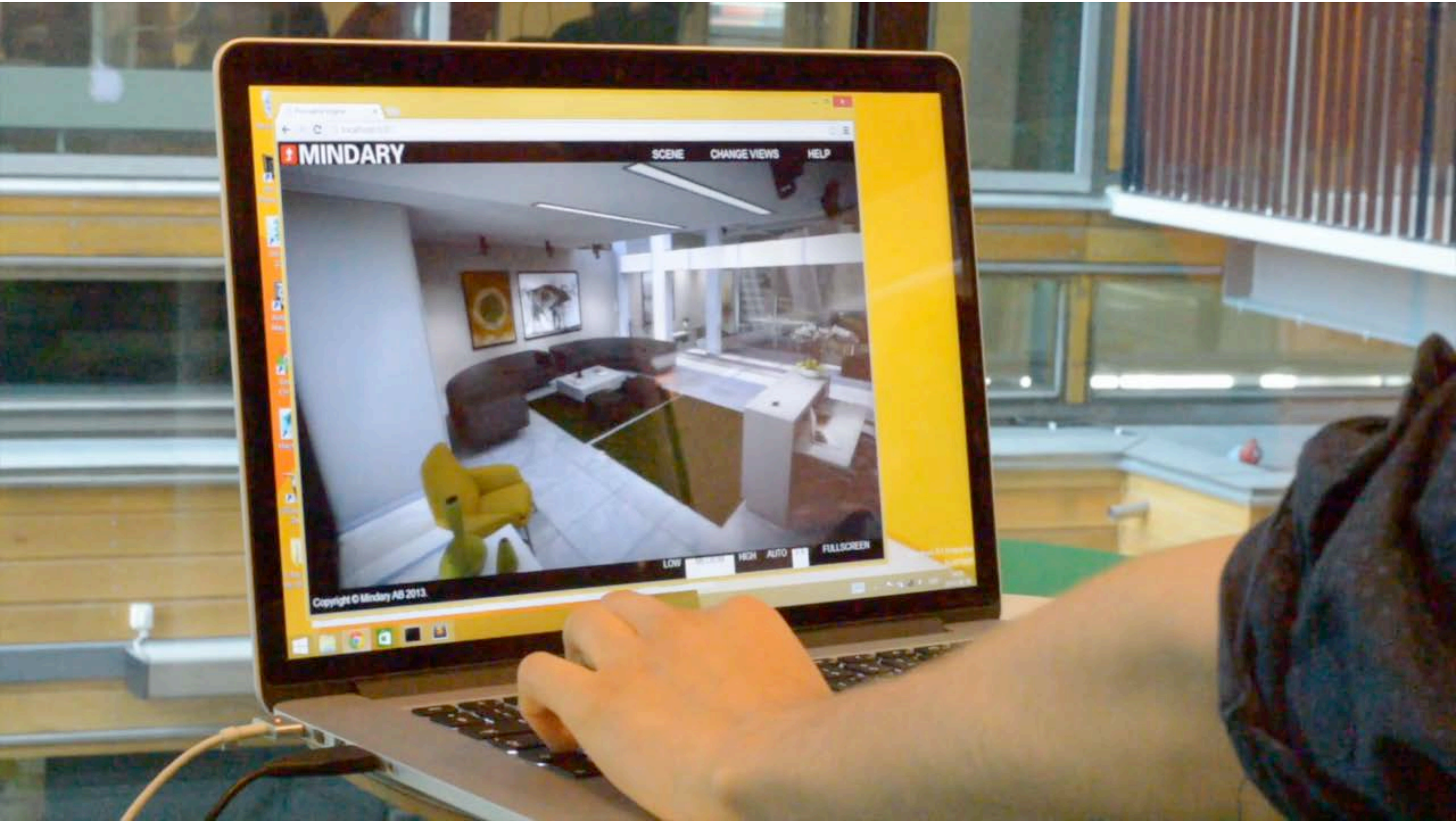
Unreal Engine 4

Combining photo textures and computer-generated view-dependent reflections

MINDARY
TECHNOLOGY AND VISUALISATION

Anyways,

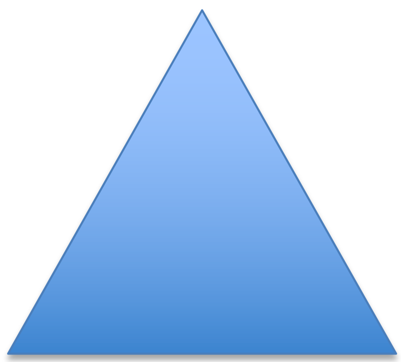
1. Generate static (non view-dependent) color textures
2. Add computer-generated reflections (most important view-dependent effect)



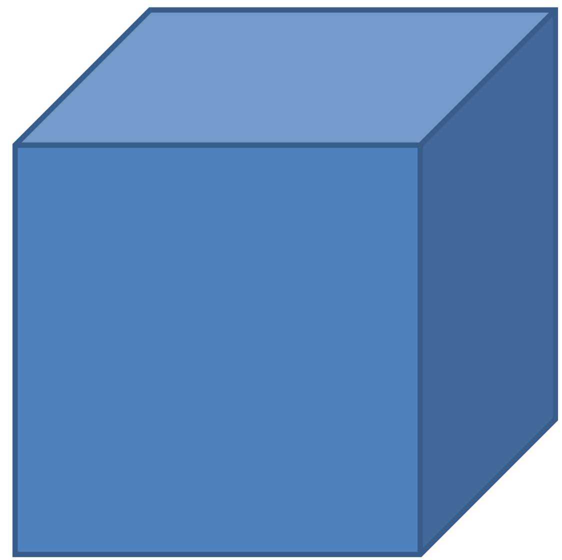
Increasing the amount of geometric details

Triangles

Voxels



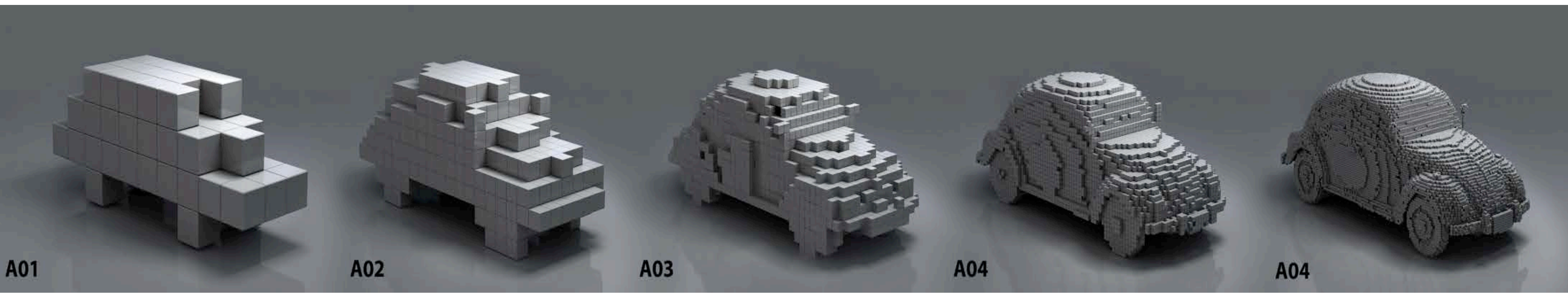
Triangle
36 bytes



Voxel
Volume – element
1 bit

Voxels

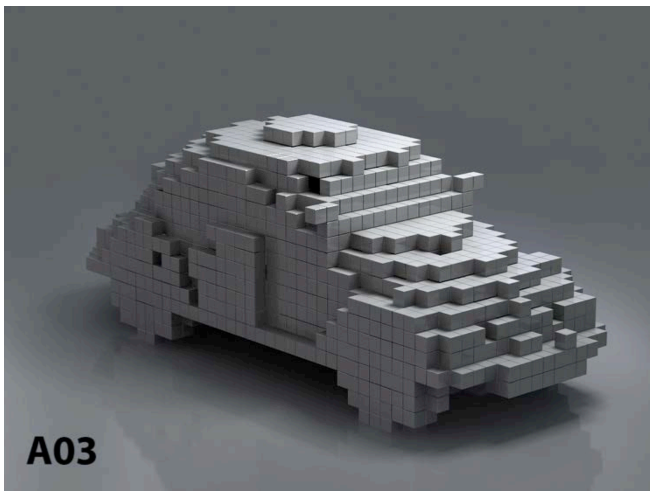
- Desirable to be able to use very high resolutions

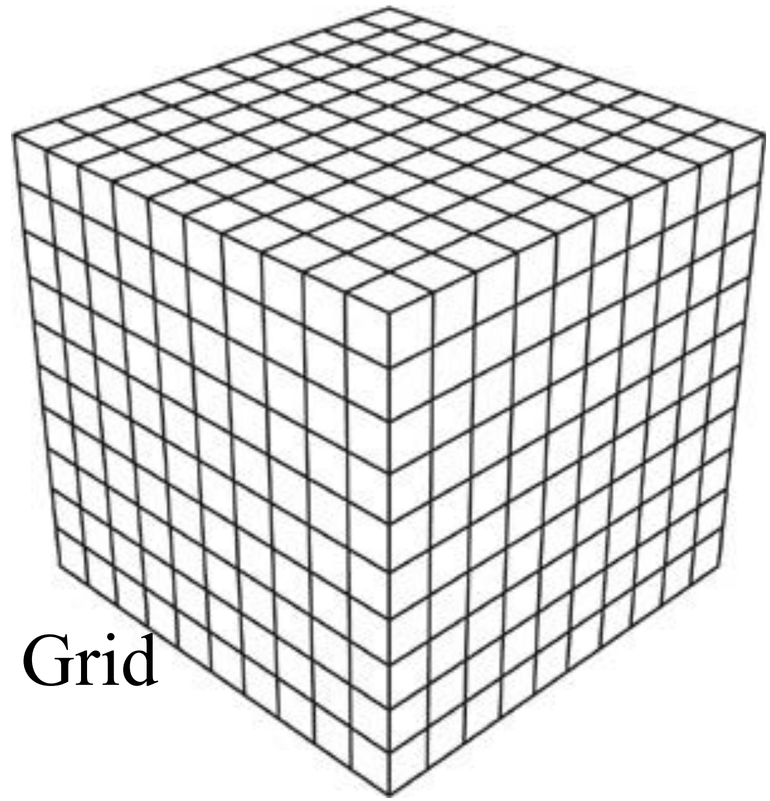


Voxels

One possible data structure:

- Grids – 3D array of 0:s and 1:s

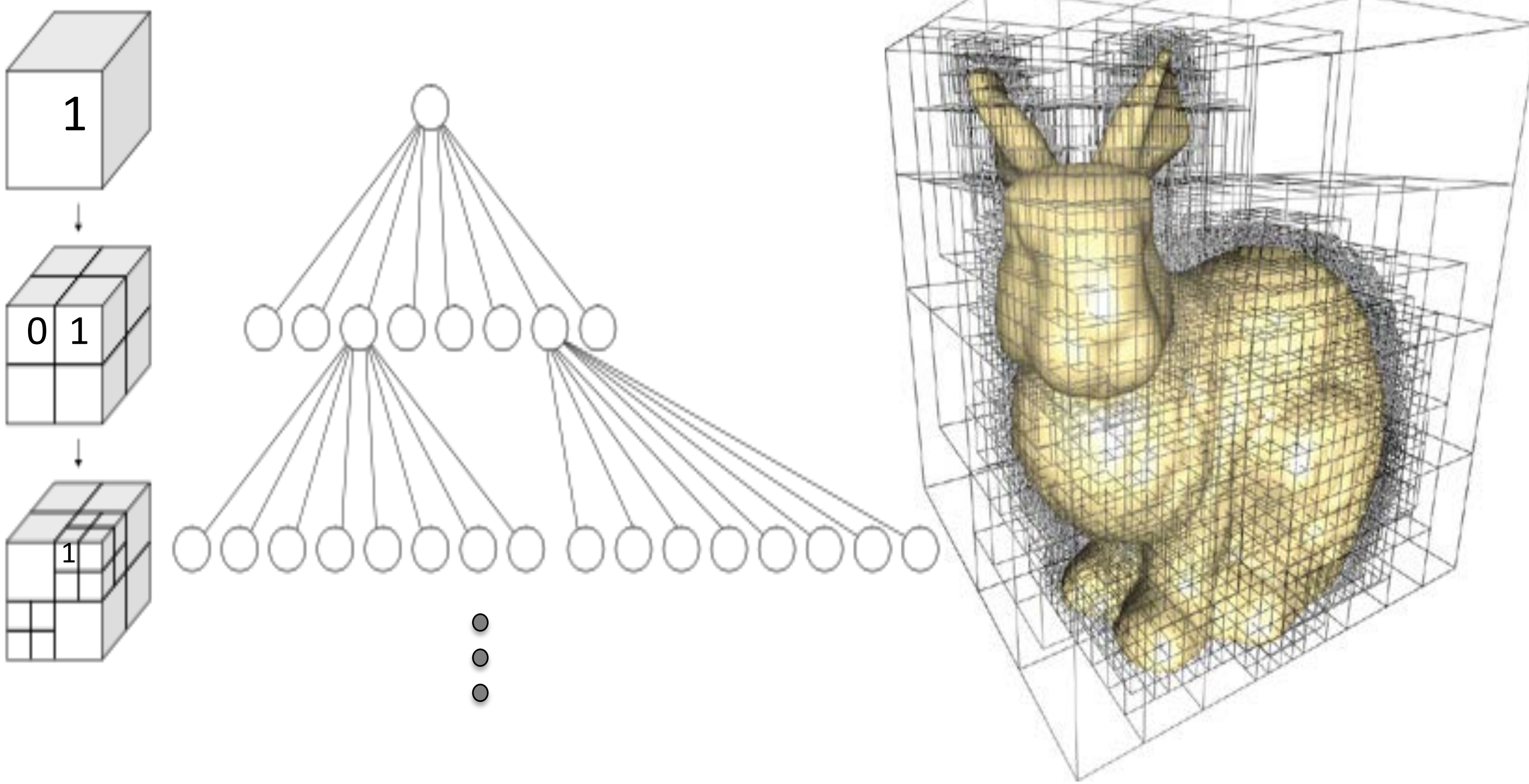




Grid

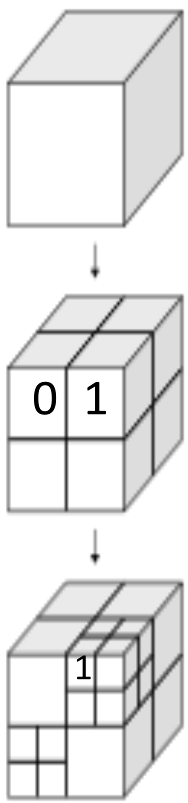
Sparse Voxel Octree

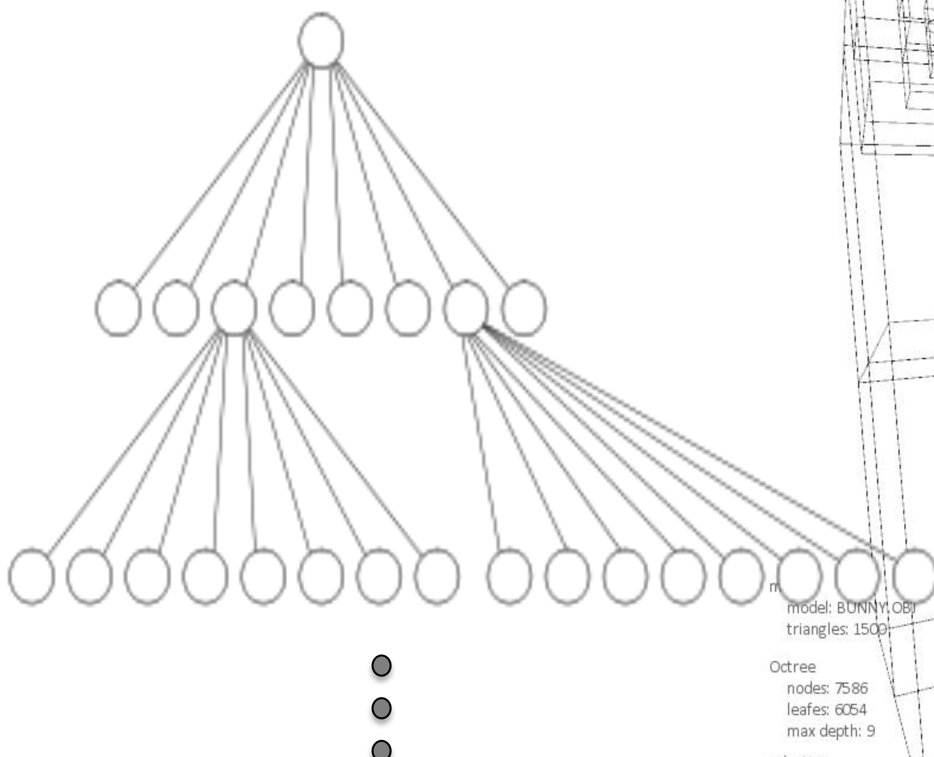
Each node has eight children, representing an octant of the parent node's volume.

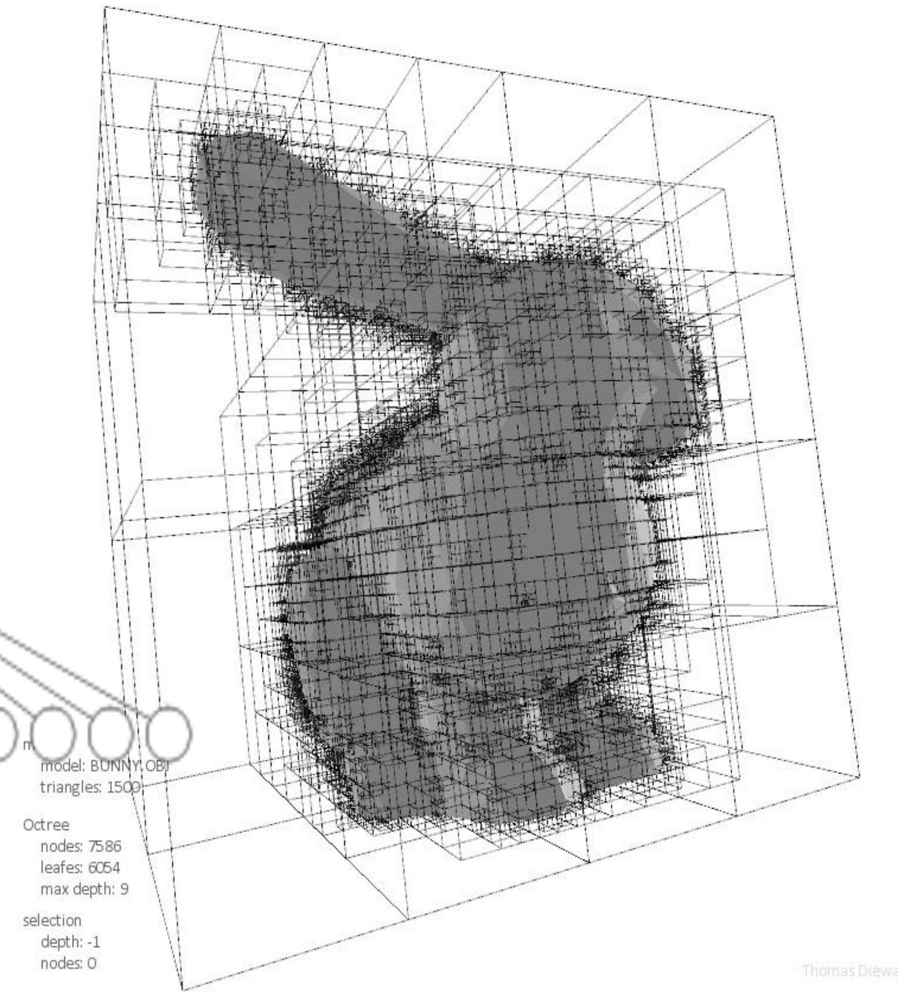


Sparse Voxel Octree

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





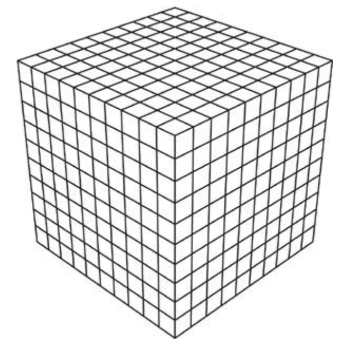


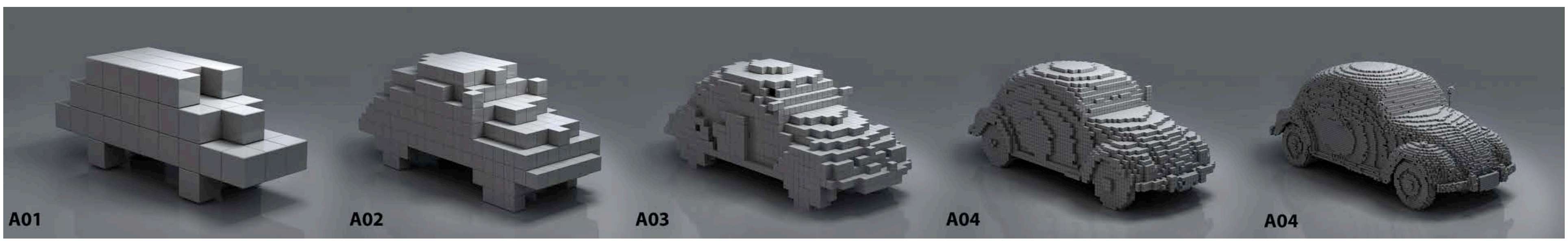
Sparse Voxel Octree

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

Voxels

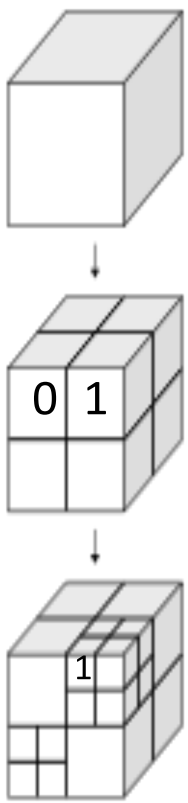
- Voxel = 1 bit.
- We currently handle scene of res = 128.000^3
  - Naively: 262 TB
  - DAGs => < 1GB

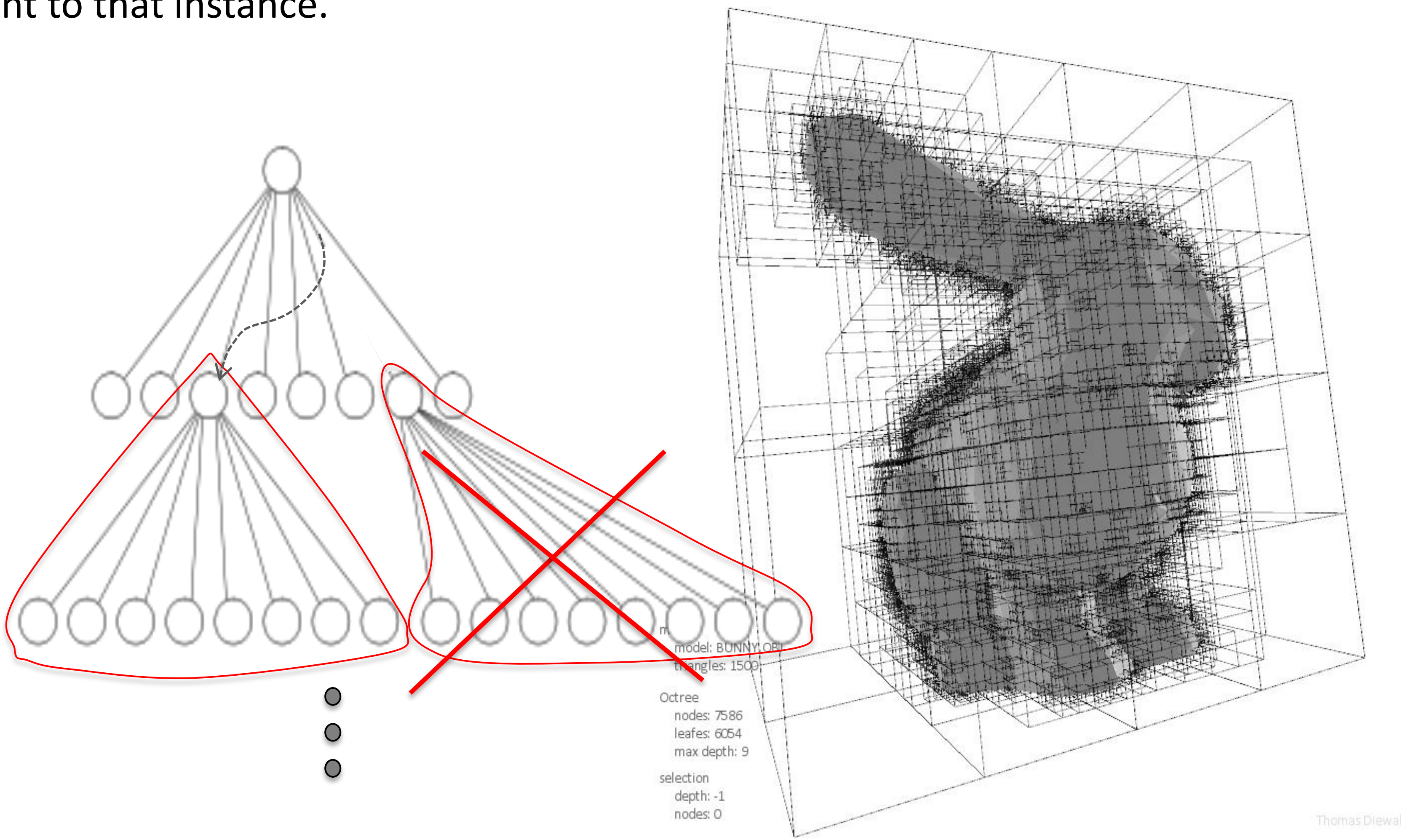




Our Voxel DAGs

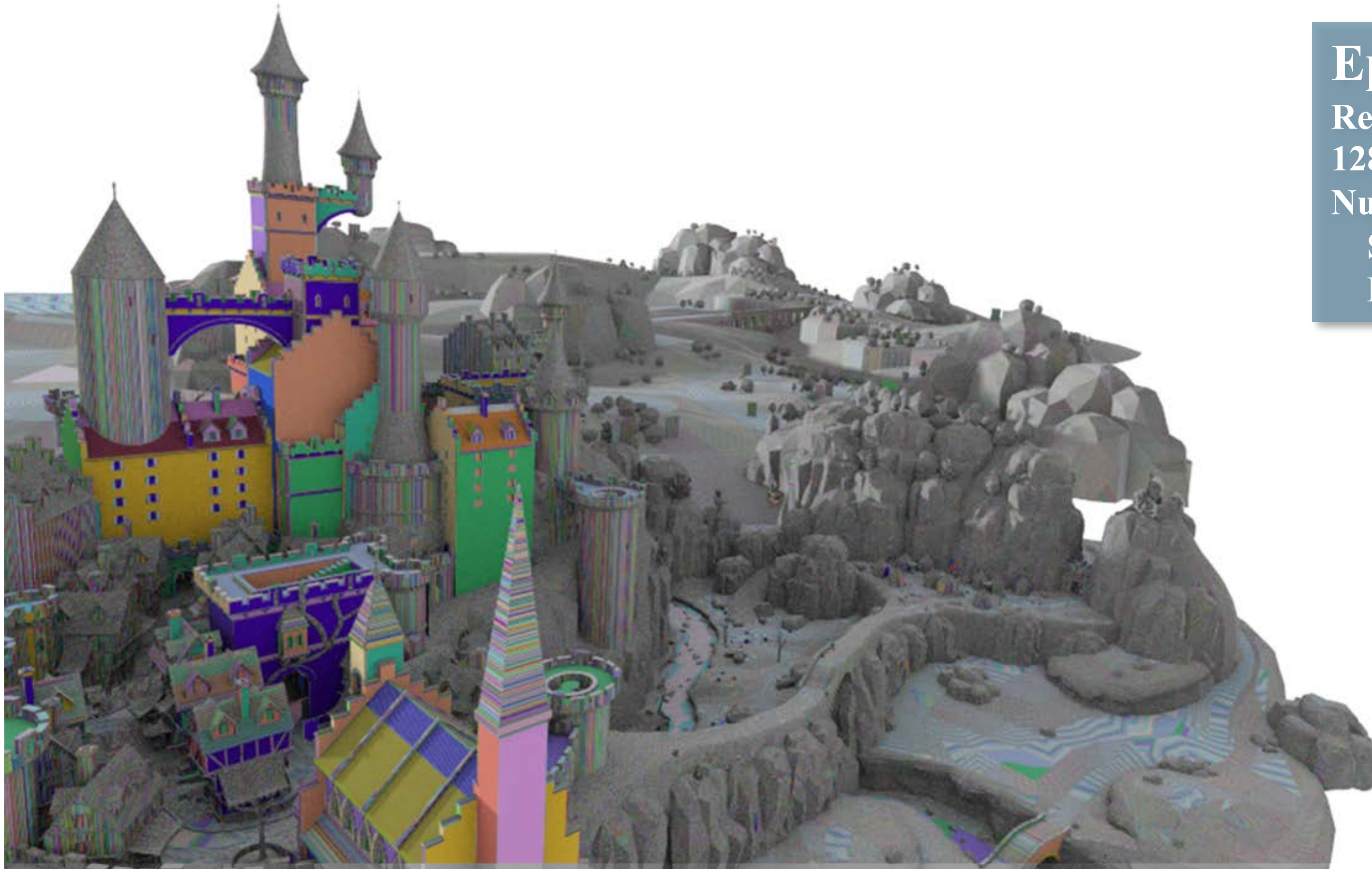
For identical subgraphs, only store one instance, and point to that instance.



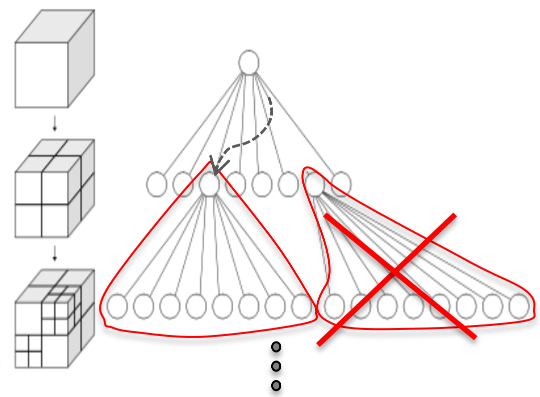



Resolution:
131072³
(~900MB)

Visualizing Identical Subtrees



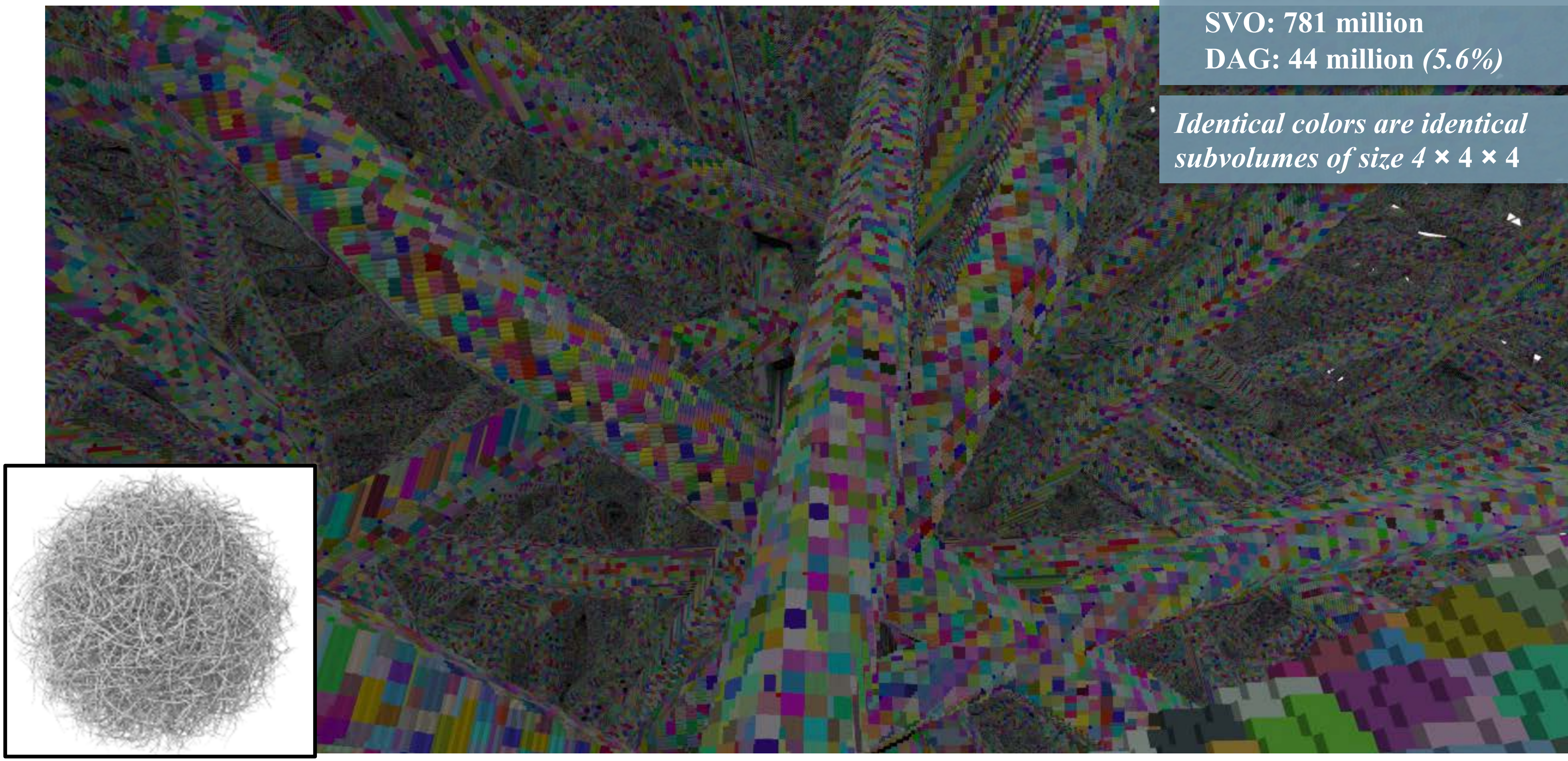
Epic Citadel
Resolution: 128K × 128K × 128K
Number of nodes
SVO: 5.5 billion
DAG: 45 million *(0.8%)*



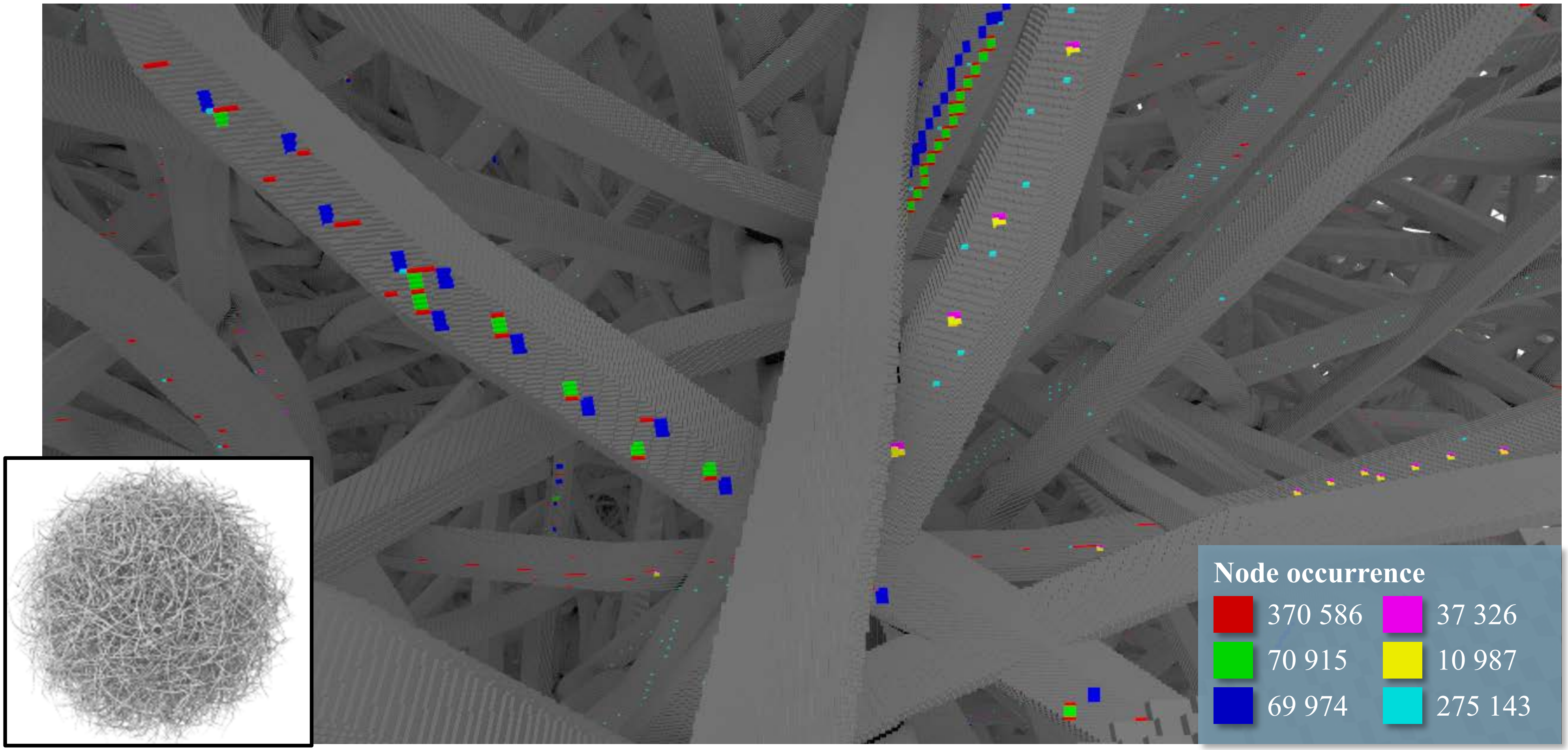
Visualizing Identical Subtrees

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., <u>Free Viewpoint Video</u>

Volumetric Video

Want:

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

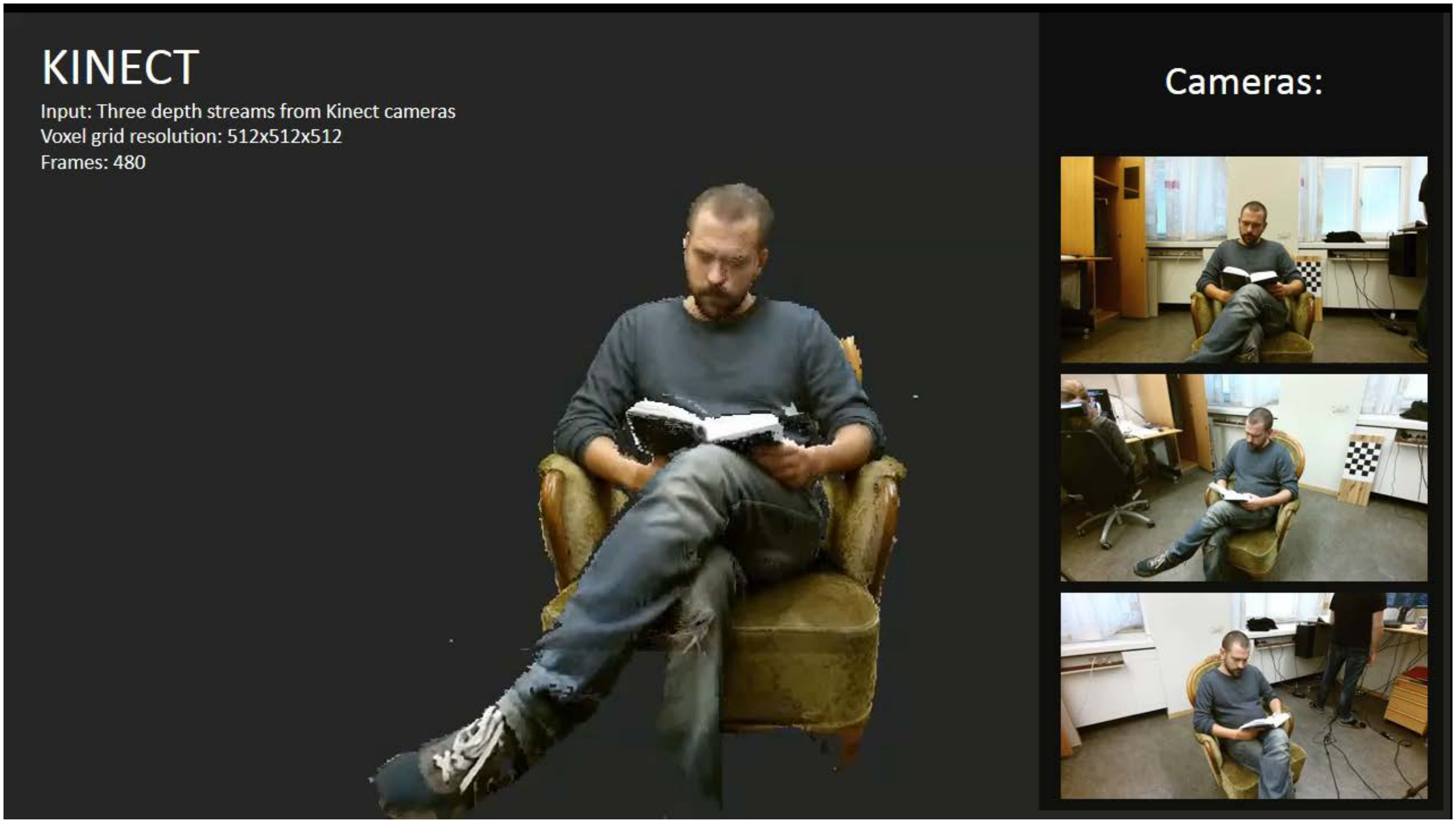
Med 2 eller fler kameror kan man beräkna djup – precis som våra ögon.

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 ~$1cm^3$. Önskar ~$1mm^3$

Volumetric Video



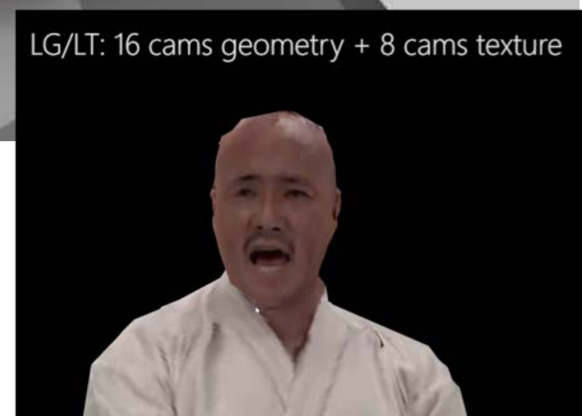
480 frames
512^3 grid
20 sec
@24Hz

5.2 MiB
0.9
GiB/hour
2.1 Mbits/s

Volumetric Video

- Professional studios:

E.g.:
- ~100 8K rgb cameras à $1.5K
- + IR cameras.
- Green screen
- Special light sources + reflectors



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

Volumetric Video for the Masses, not the Classes.

(aka Free Viewpoint Video, or 3D scanning+viewing of dynamic scenes)

”For the Masses” - reason:

1. 100 high-end cameras, rgb + IR + controlled lighting + green screen:
   - very very expensive.
   - Lots of inefficient computations => hard to make real time.
2. We want accessibility for anyone:
   - teenagers, influencers, youtube, …
3. Volumetric face-to-face communication.

Volumetric-video film makers

Handcraft recording

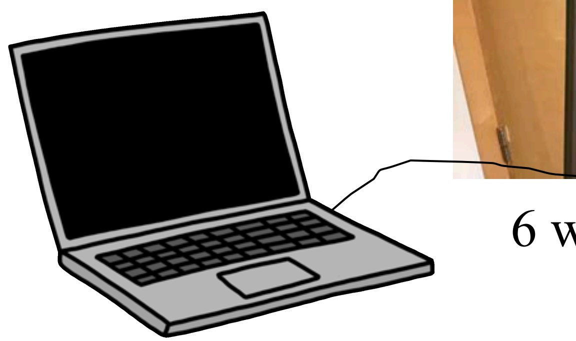
Influencer

Volumetric Video for the Masses, not the Classes.

Subproblems:

1. cheap real-time 3D scanning (acquisition) using only a handful of cheap web cameras without requiring complex setup.
2. Highly compressed 3D-video-transmission format over internet
3. High-quality Real-time rendering
   - view dependence

6 webcams à €60, HD-res, 30fps

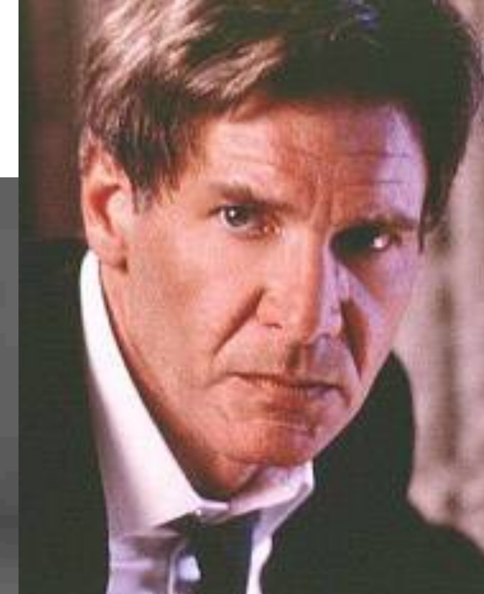


1:

All in real-time 30 fps (recording + viewing)

- using unsynchronized cheap web cams,
- USB to **one** computer. Running on **one** CPU/GPU
- inherently not restricted to faces only.

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

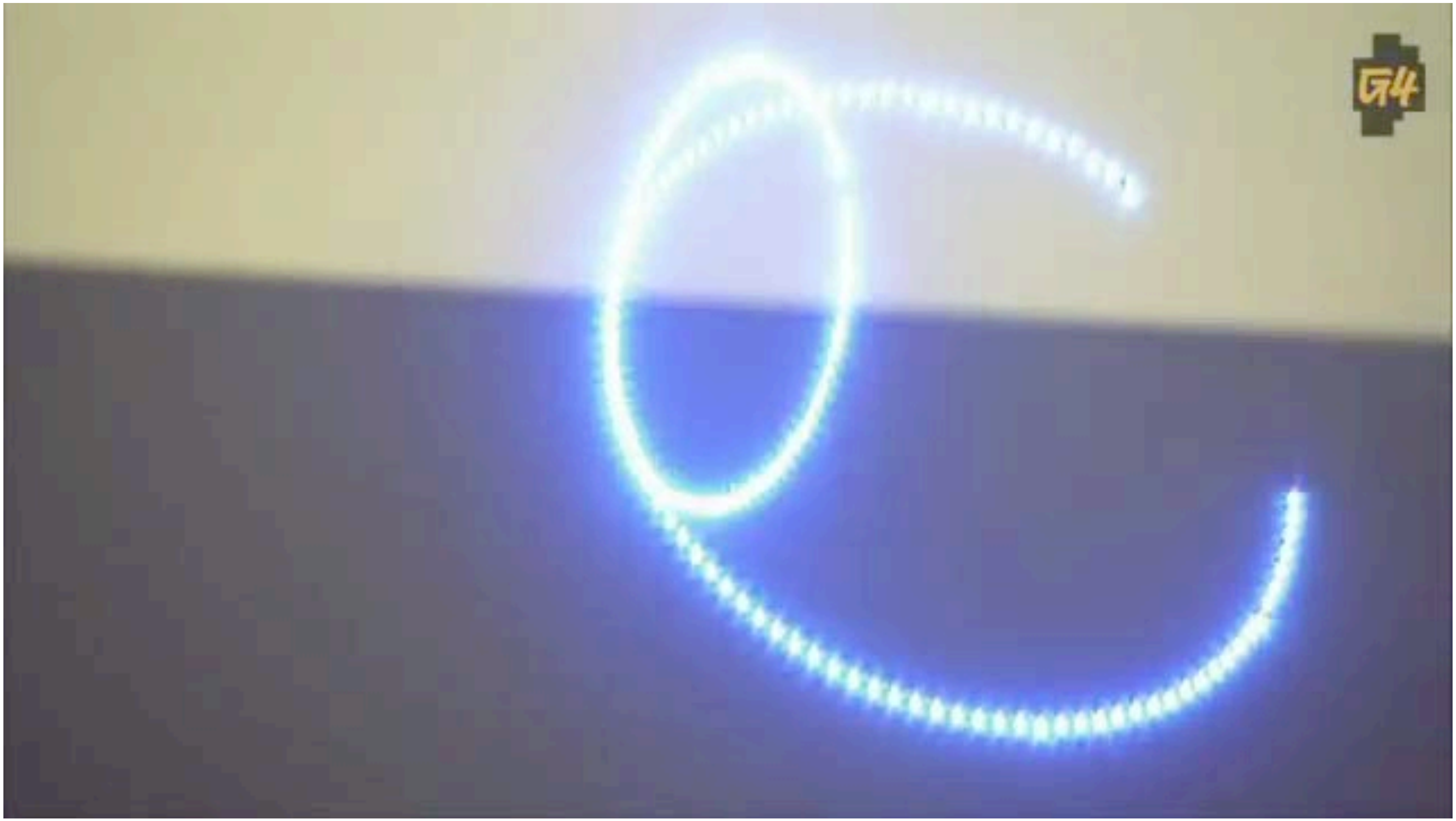
Star Trek - Tablets

80:ies

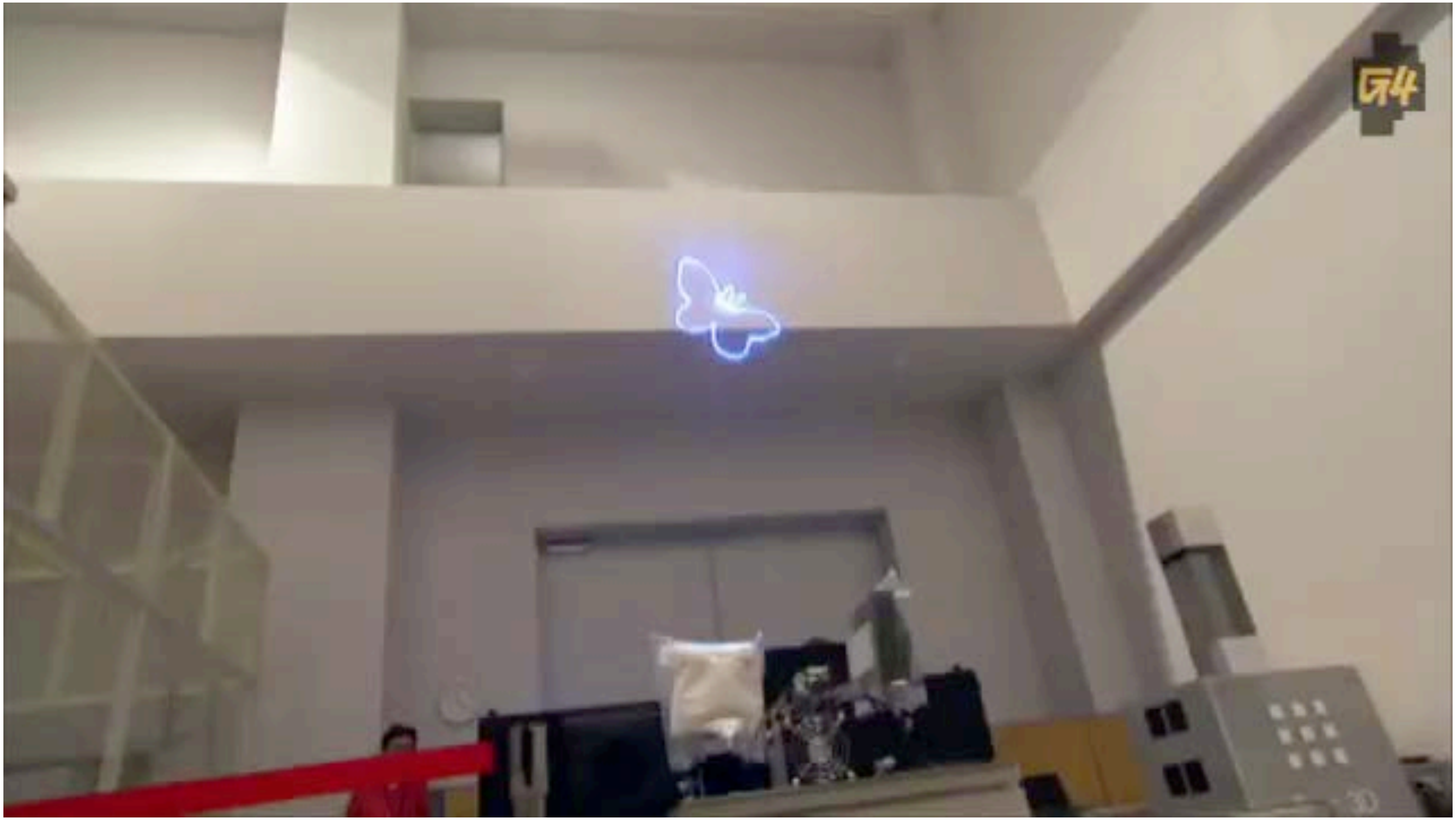
2010

Star Trek - Holodeck

Mid air display

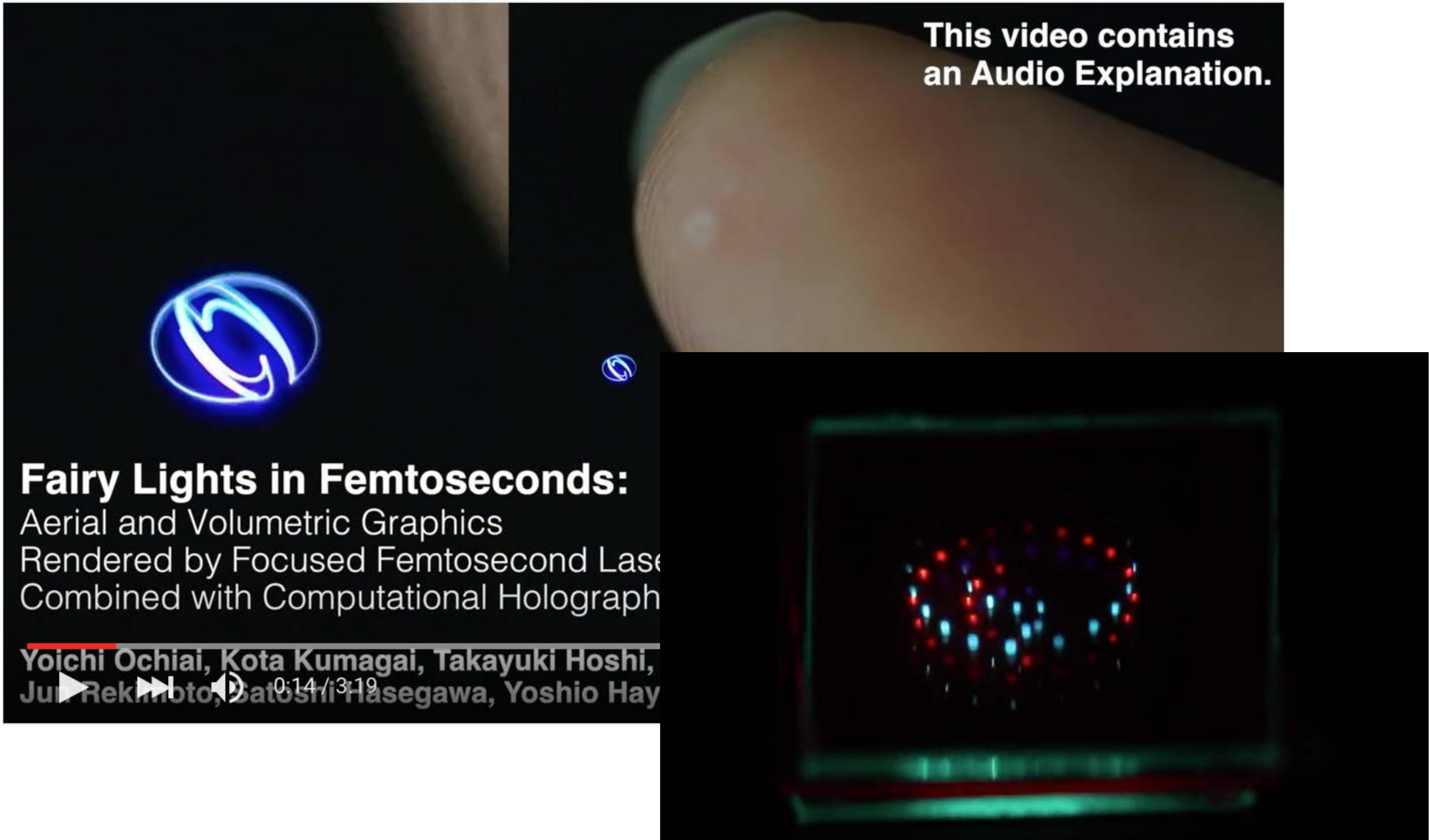


Mid air display



Mid air displays 2015

Mid air displays 2015



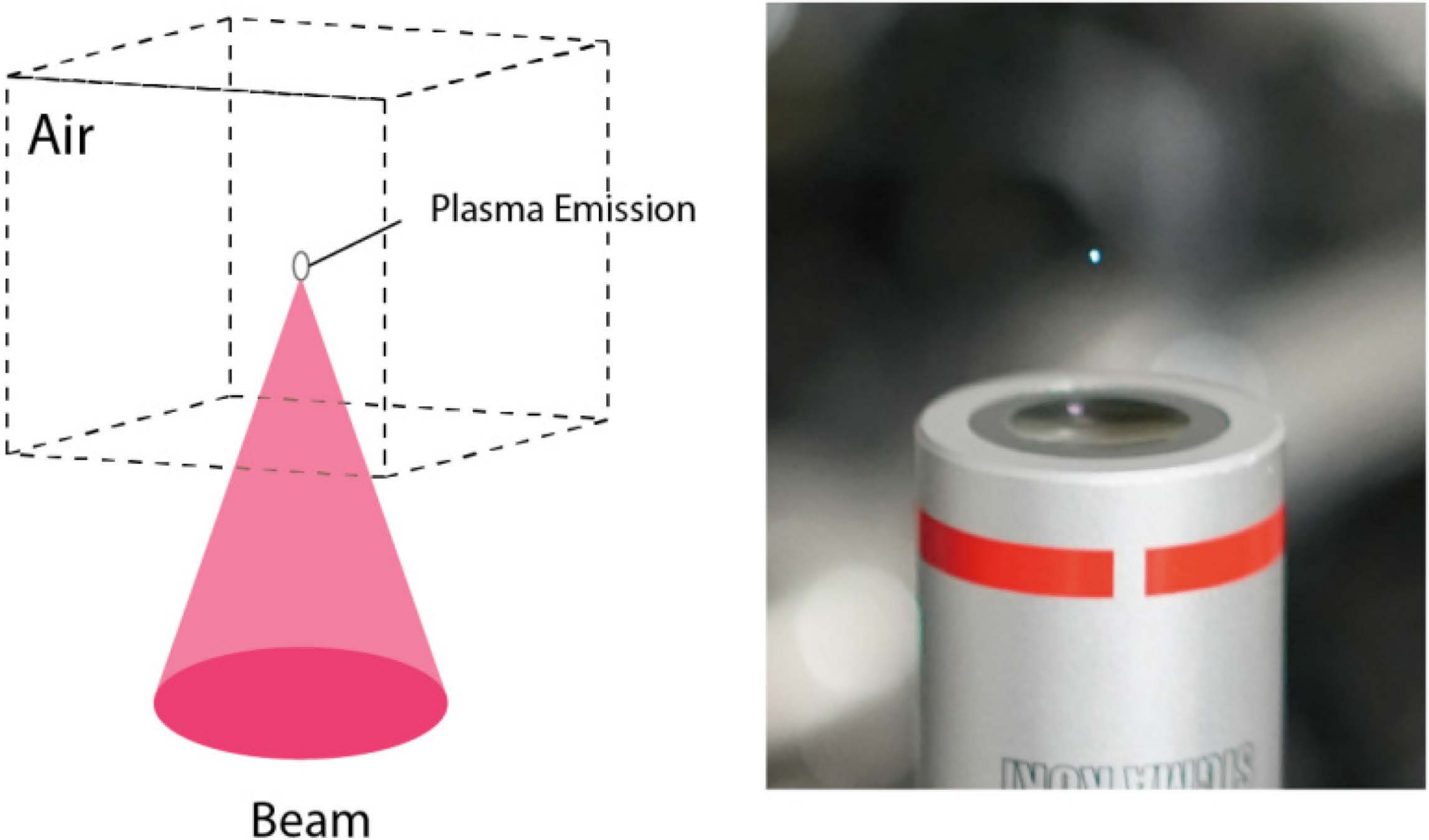


Figure 4: *Laser plasma induced by focused femtosecond laser.*

Rendering Volumetric Haptic Shapes in Mid-Air using Ultrasound

Inget nytt under solen

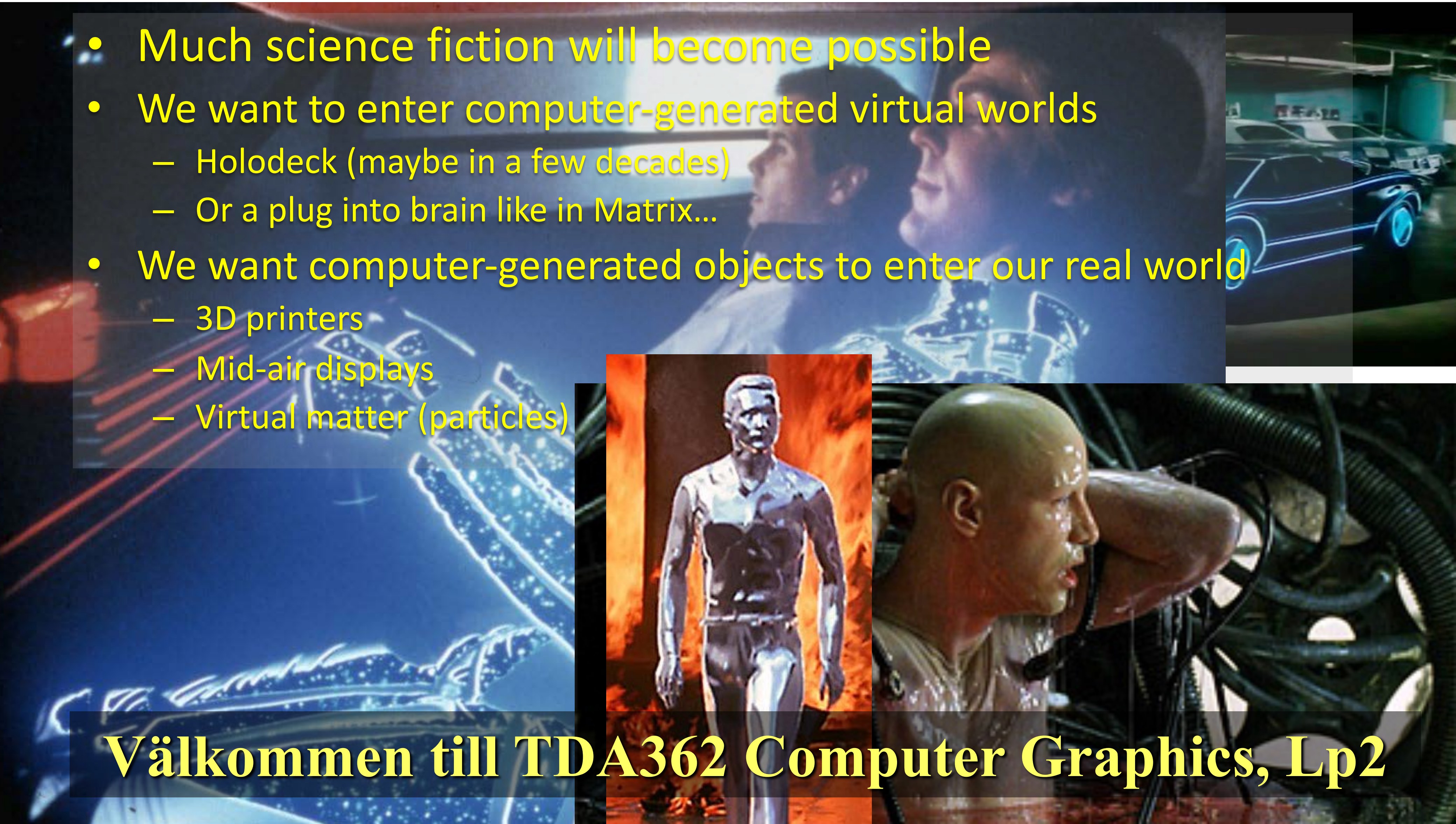
- Titta på Automan från 1983-1984.

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