3D Digitization for Cultural Heritage

3D FROM 10.000 FEET

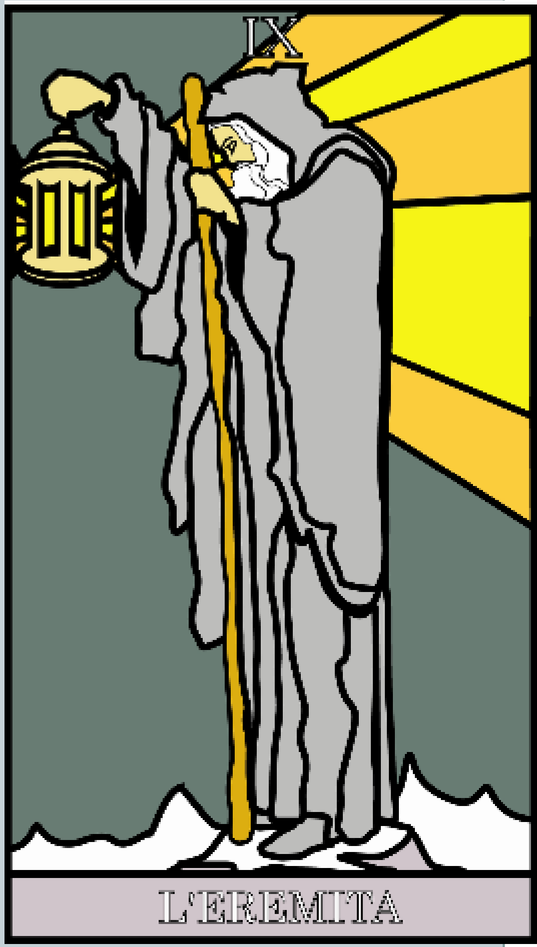
Who am I?

Marco Callieri

- Master degree & PhD in computer science
- Researcher at the Visual Computing Lab, ISTI-CNR, in Pisa
- I work on 3D data manipulation and rendering... lot of experience in 3D scanning and data processing
- Most of my activities are in the field of cultural heritage

http://vcg.isti.cnr.it/~callieri

callieri@isti.cnr.it



Beside this:

an eclectic artisan, an avid gamer, a former biker, a good cook, an incorrigible geek... and much more

Visual Computing Lab

Research group working on **3D computer graphics**

part of:

Istitute of Science and Technologies of Information (**ISTI**)

part of:

National Research Council (**CNR**)

http://vcg.isti.cnr.it

3D from 10.000 feets

3D has become quite a buzzword in the last few years

Seems that, with 3D, everything is better...

We aim at providing you a focused knowledge of devices, tools and techniques that employ 3D in the field of cultural heritage...

However, we do believe an introduction to basic, general 3D concepts may help... even a simple (and very incomplete) one as this :) :)

Dual nature

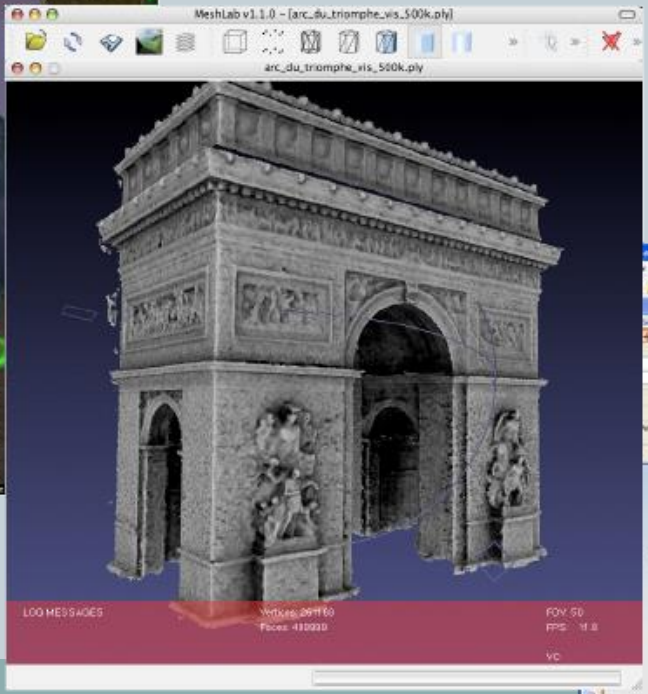
3D is used a lot because it has two natures:

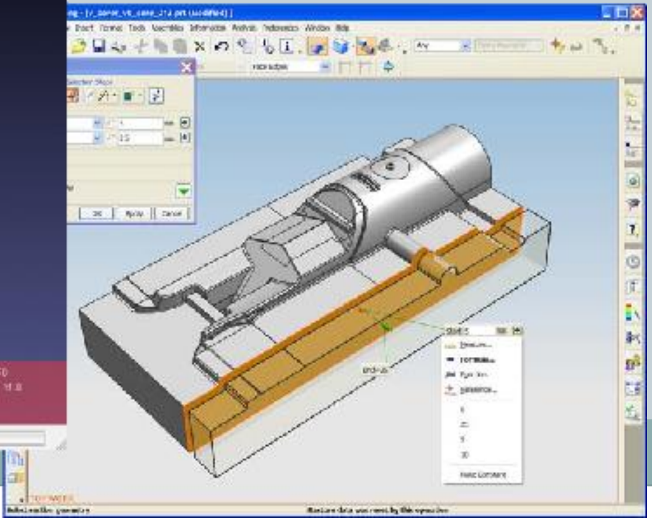
- **PURE DATA:** 3D is metric, geometrical information…. In a word: numbers. It is possible to do calculation over it, and work on it from a mathematical point of view.
- **VISUAL:** 3D data can be visualized such that is perceived in the same way we perceive reality. In this way we can access the data in a more natural way.

The combination of these two properties is what makes 3D valuable…

Interactive 3D

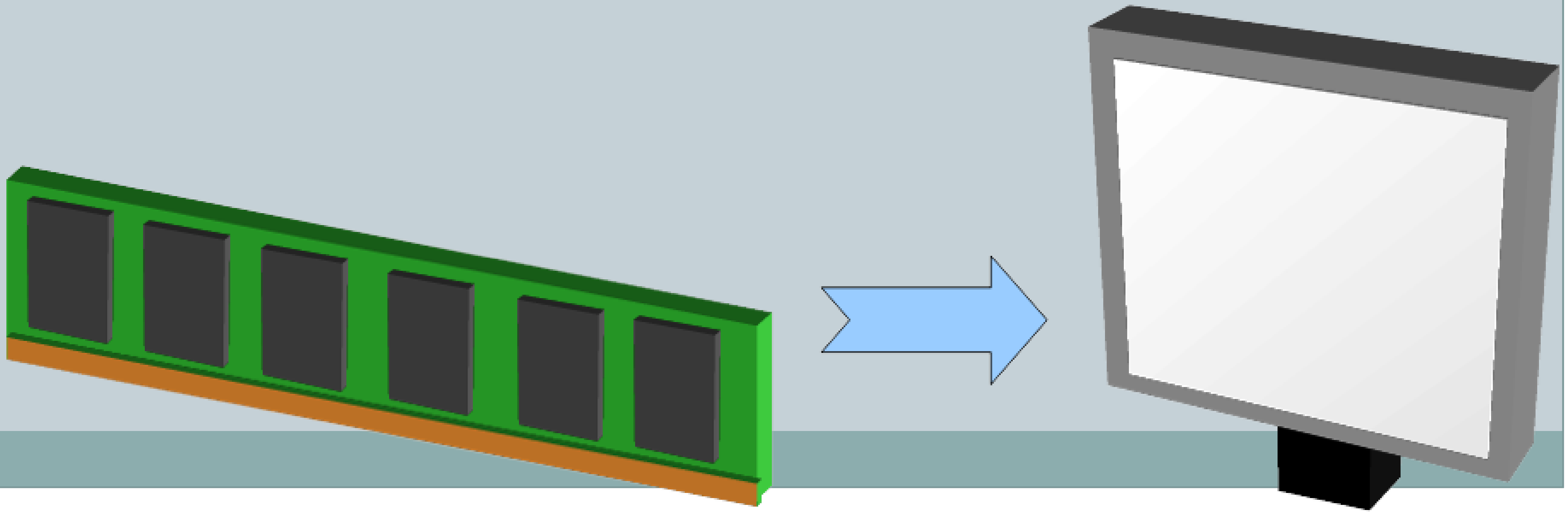
You may have encountered applications where a digital 3D world is *somehow* displayed and, *somehow* the user may change the view over this world or the world content...


MeshLab v1.1.0 – [arc_du_triomphe_vis_500k.ply]
arc_du_triomphe_vis_500k.ply
LOG MESSAGES
FOV: 50



Behind the curtain

Given this behaviour, you may speculate that there is a representation of the a 3D environment in the computer memory, AND a way to produce (in real-time) an image that represent a view over this world, PLUS a way to interact with this environment...

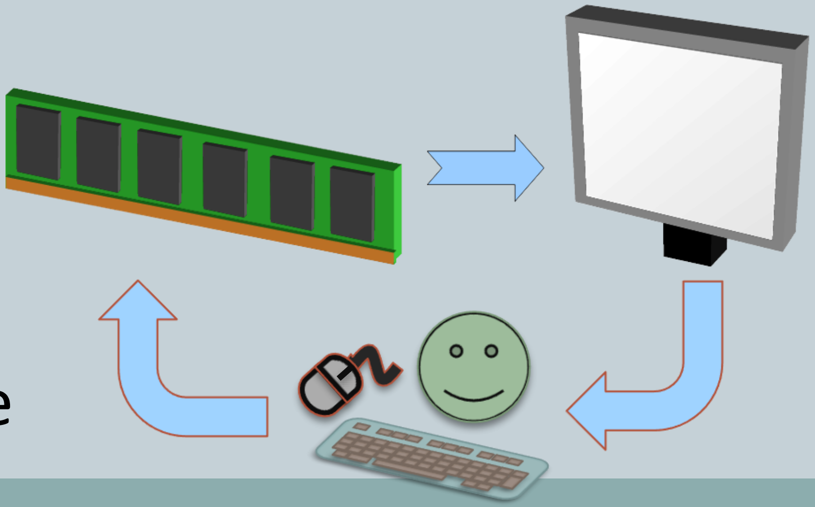


Realtime Cycle

Actually, what happens is a CYCLE.

1. 3D scene state is rendered into an image
2. User looks at the image, and issues commands via some interface (keyboard, mouse, …)
3. The 3D scene state is updated accordingly
4. Rinse and repeat

This process is repeated MANY times per second, to give the impression of a continuous update



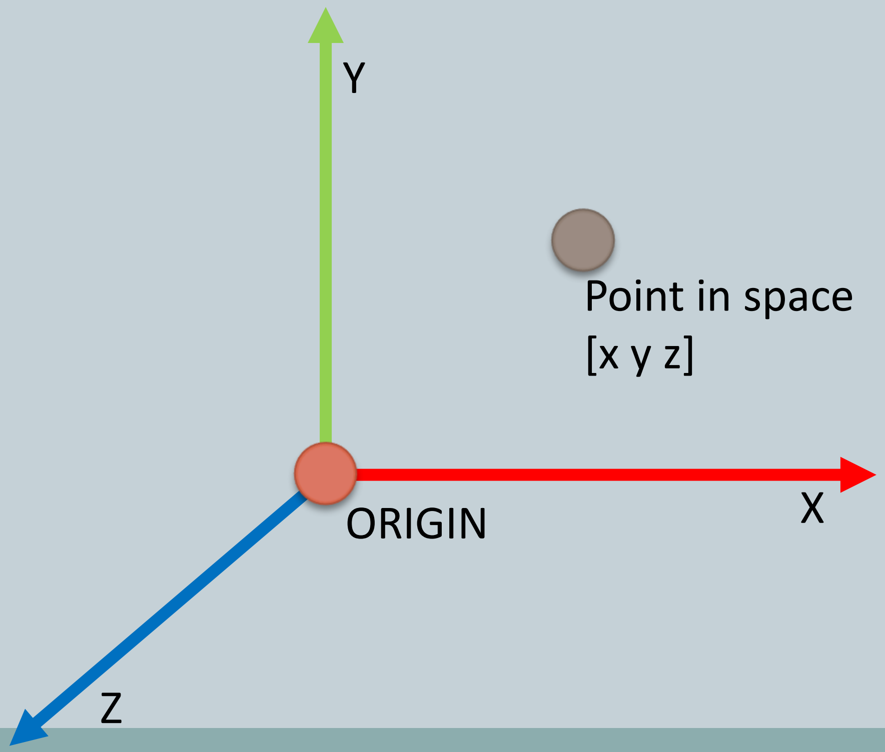
RENDERING

A common word in the 3D field, “rendering” is used in different ways, but has a general meaning

RENDERING: the process by which some data in memory (in some abstract/structured way), is converted into a representation that can be directly perceived by the user

3D Space

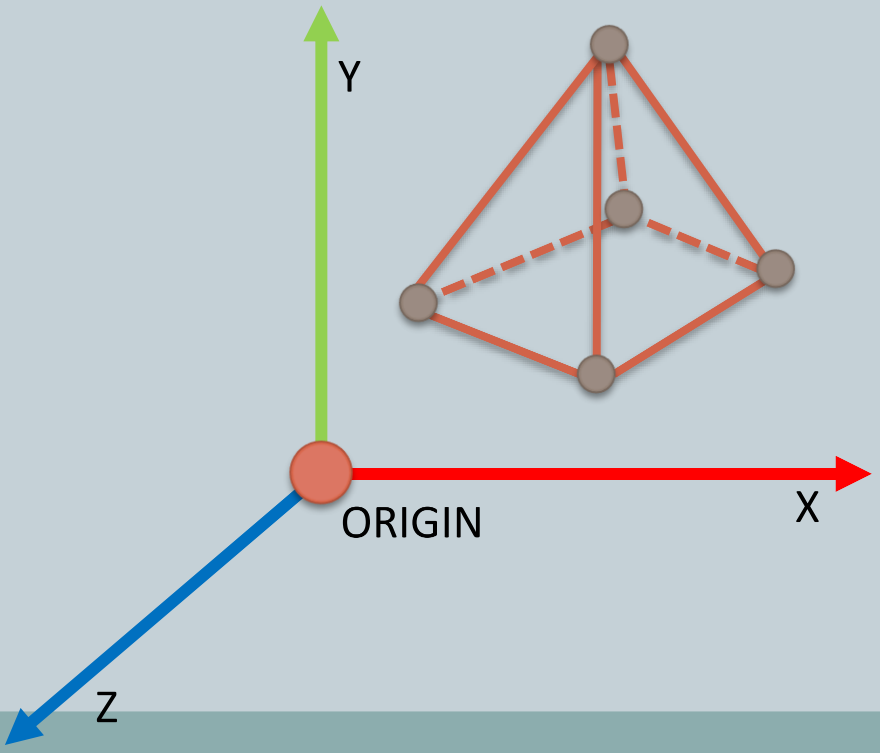
Our 3D world is a **space** defined by an **origin** (a reference point arbitrarily chosen) and three orthogonal axis. Points in this space are defined by their three coordinates [X Y Z].



3D Space

VERTICES are just point in space, FACES (triangles) are built on top of vertices, a 3D MODEL is a MESH of triangles.

Vertices are the 3D spatial info, triangles are TOPOLOGY.



3D models

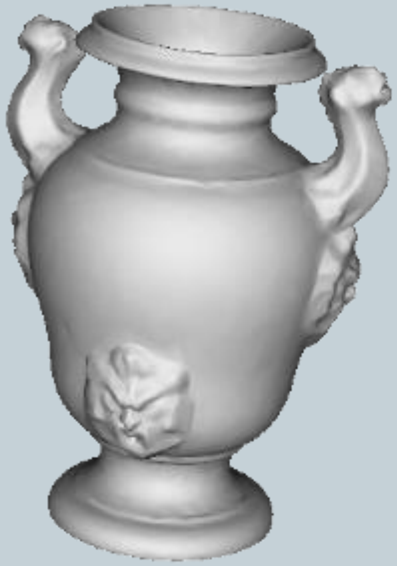
Most 3D models (especially the ones you'll see in our course) are made just by VERTICES (points in the 3D space) connected in TRIANGLES.

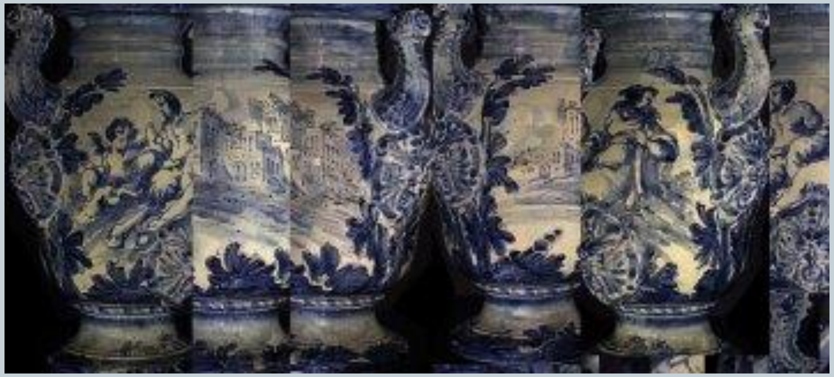
Triangles are always planar, with enough triangles you may represent any shape, the mathematics and algebra used to manipulate triangles is simple, fast, and easy to be managed by computers...

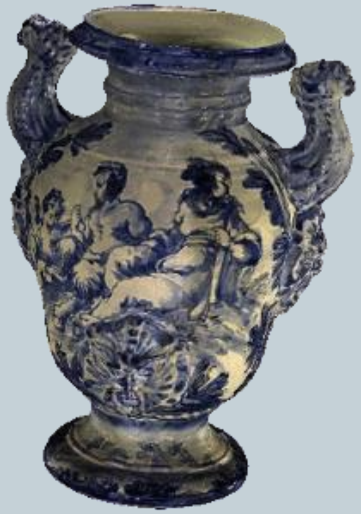
Dual nature

Triangulated models does only have geometrical information of a zero-thickness surface (only the **external shell**).

Other info, like color, are generally mapped on this surface.


3D geometry




Texture mapped rendering

The SCENE

Models are arranged inside a space, trough rotations translations and scale operations, defining a **SCENE**.

What we see in the screen is a **VIEW** of the scene.

Every tool defines a scene with its own data structure…

Beside 3D models, in a scene, there are often many other elements

- One or more ***camera***s: defining view over the scene
- **Lights**: used to simulate illumination
- ...

Interaction

In a realtime application it is possible to “interact“ with the 3D scene in some way, and instantly see the effect of our interaction.

At minimum, it is possible to change the point of view over the 3D space.

On a different level, it may be possible to interact with scene elements (moving models around the scene).

Editing tools (3D modeling, CAD and similar) let you also modify models.

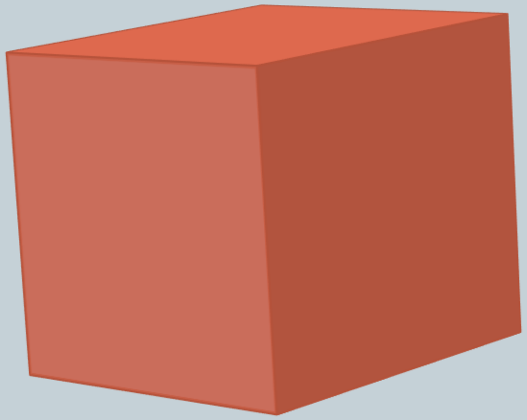
How this interaction is done, does change from one tool to another...

Realtime Rendering

Bringing these kind of data to the screen in an interactive application is called **realtime rendering**.

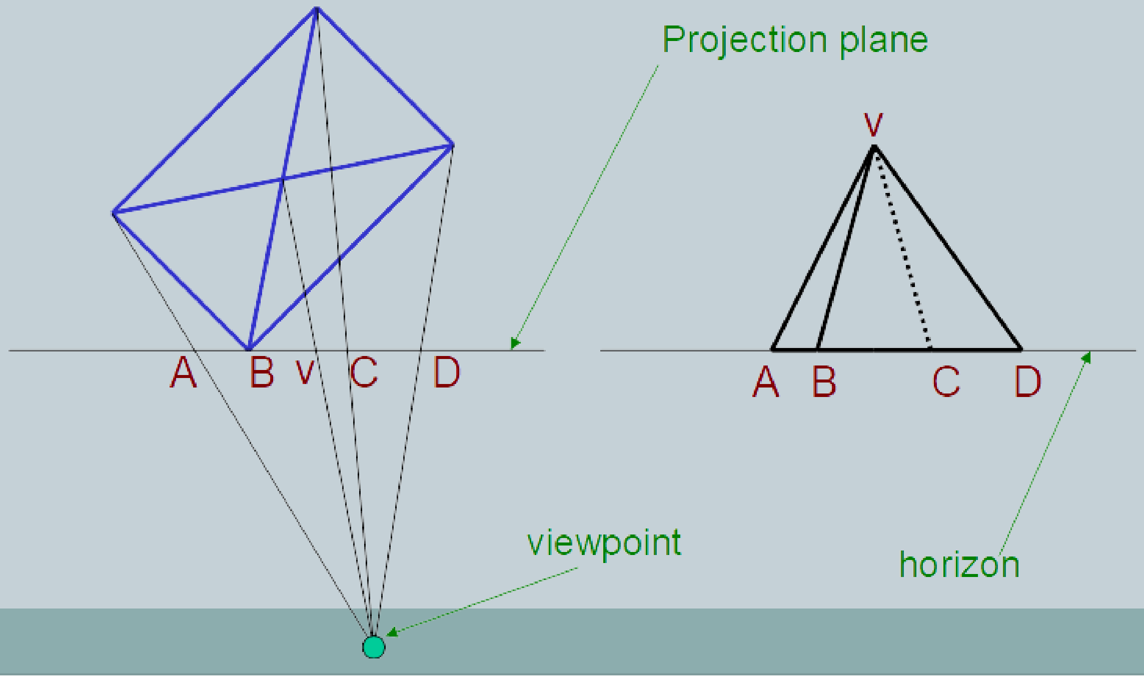
Basically, is a way to simulate how the image is formed in our eyes (perspective projection) and how light interacts with objects (lighting)...

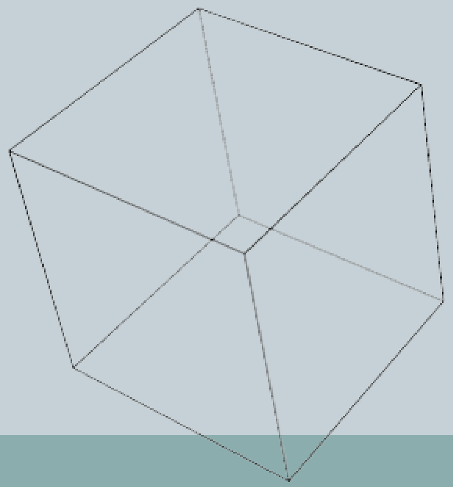
Of course this simulation is simple and crude, to be able to do it many times per second



Dual nature

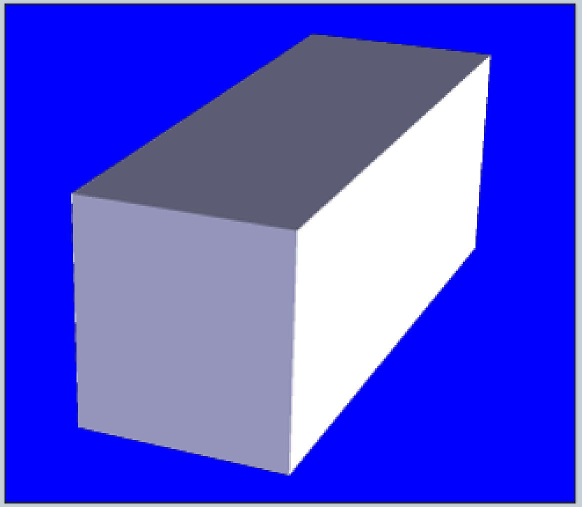
Given a 3D space, with 3D models and a **CAMERA**, the vertices are perspective-projected like we did at school in technical drawings... Only vertices are projected, triangles are drawn filling the space between points...

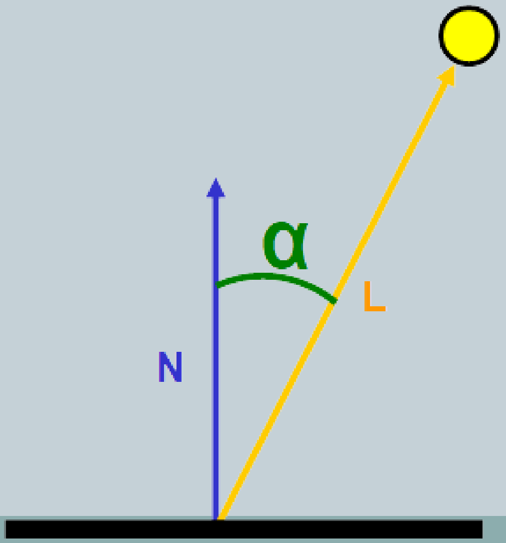


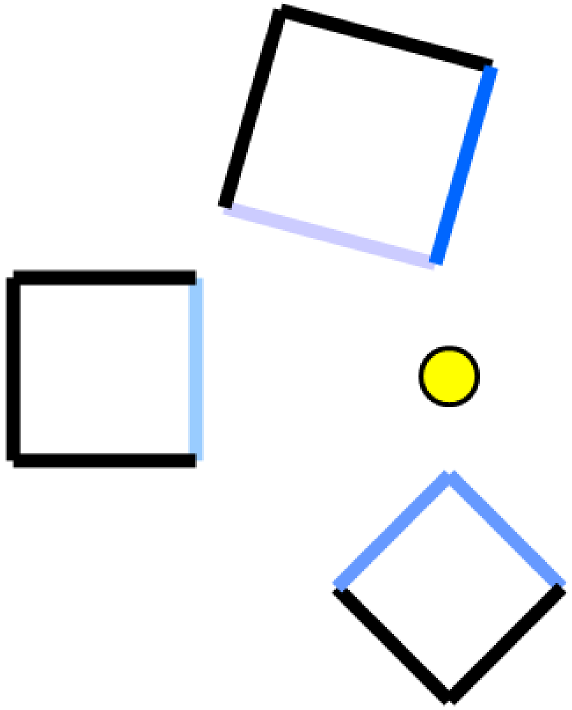


Dual nature

The projected model will appear correctly from a geometrical point of view, but completely flat. To simulate lighting, the simplest method is to calculate the angle between light and each triangle. The more orthogonal the light is, the brighter the surface.





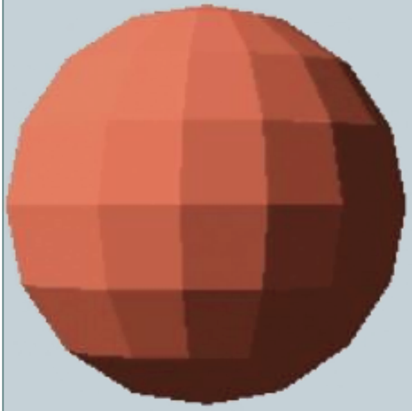


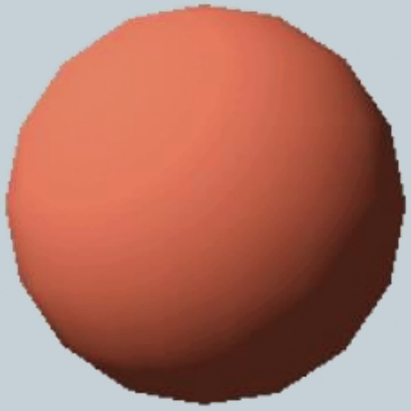
Dual nature

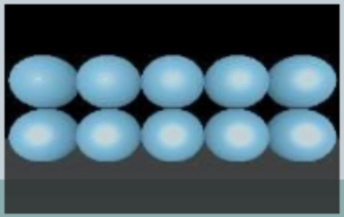
Things may get much more complex than this... Adding more lighting effects, more complex lighting models, and other real-time tricks.

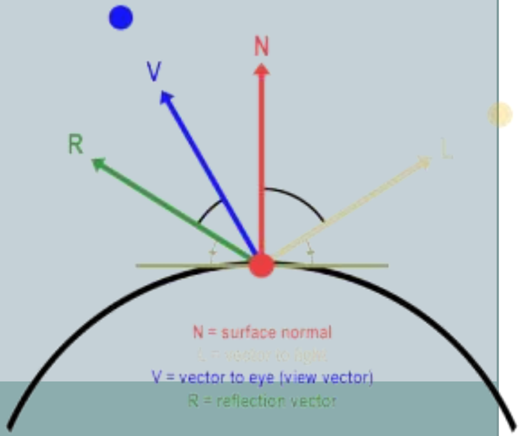
TRICKS, because almost nothing that is done in real-time rendering is really physically correct...

We will return to realtime soon, but first...

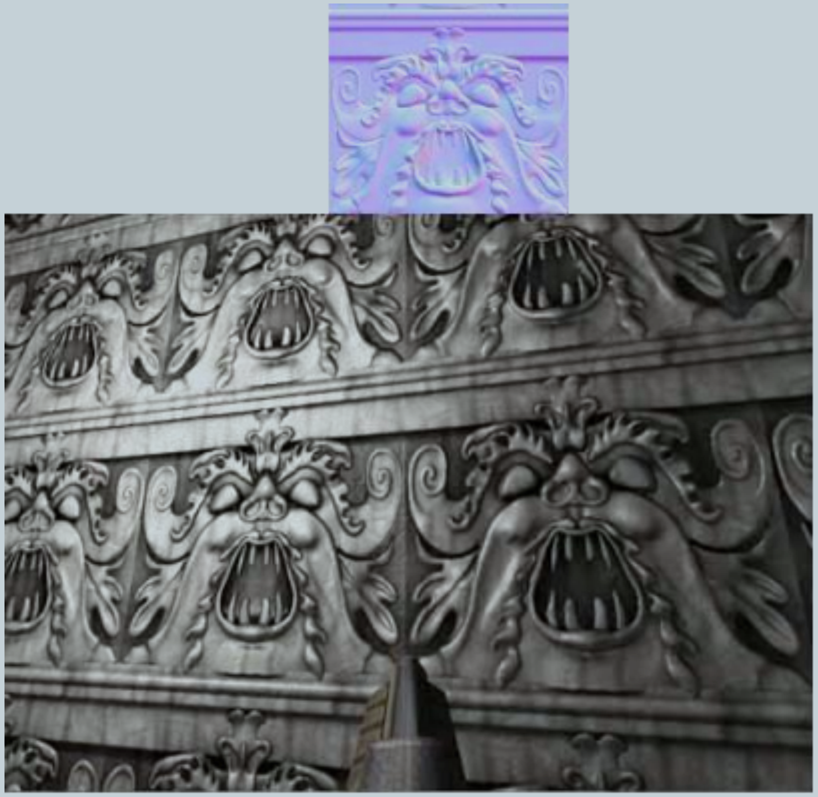

Flat


Gouraud

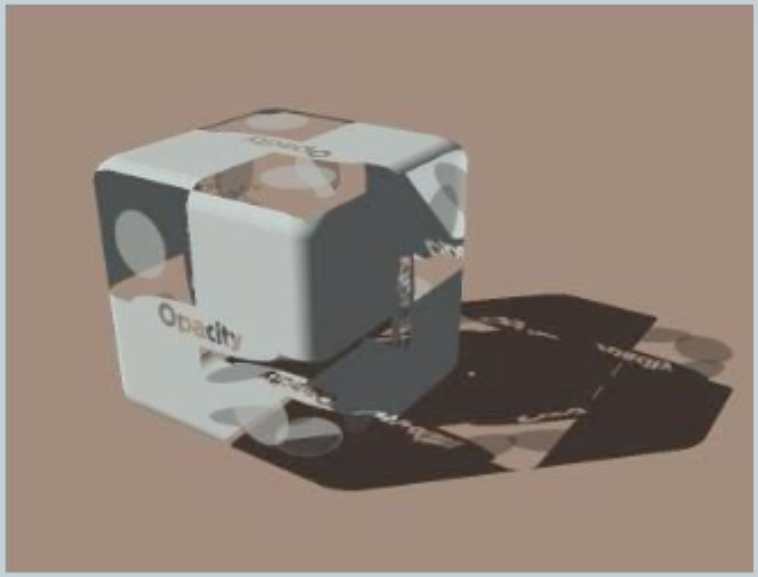




Tricks



Tricks



Non-Interactive 3D

But you may have also seen images and videos generated from 3D...

Offline Rendering

Again, you may speculate there should be a way to go from a 3D data representation (which may be *similar* to the 3D representation used in realtime) to a photorealistic representation.

Yes, this method is much more complex and cannot be achieved in realtime (we will call it OFFLINE rendering)..

No interaction cycle. Result is a passive media.

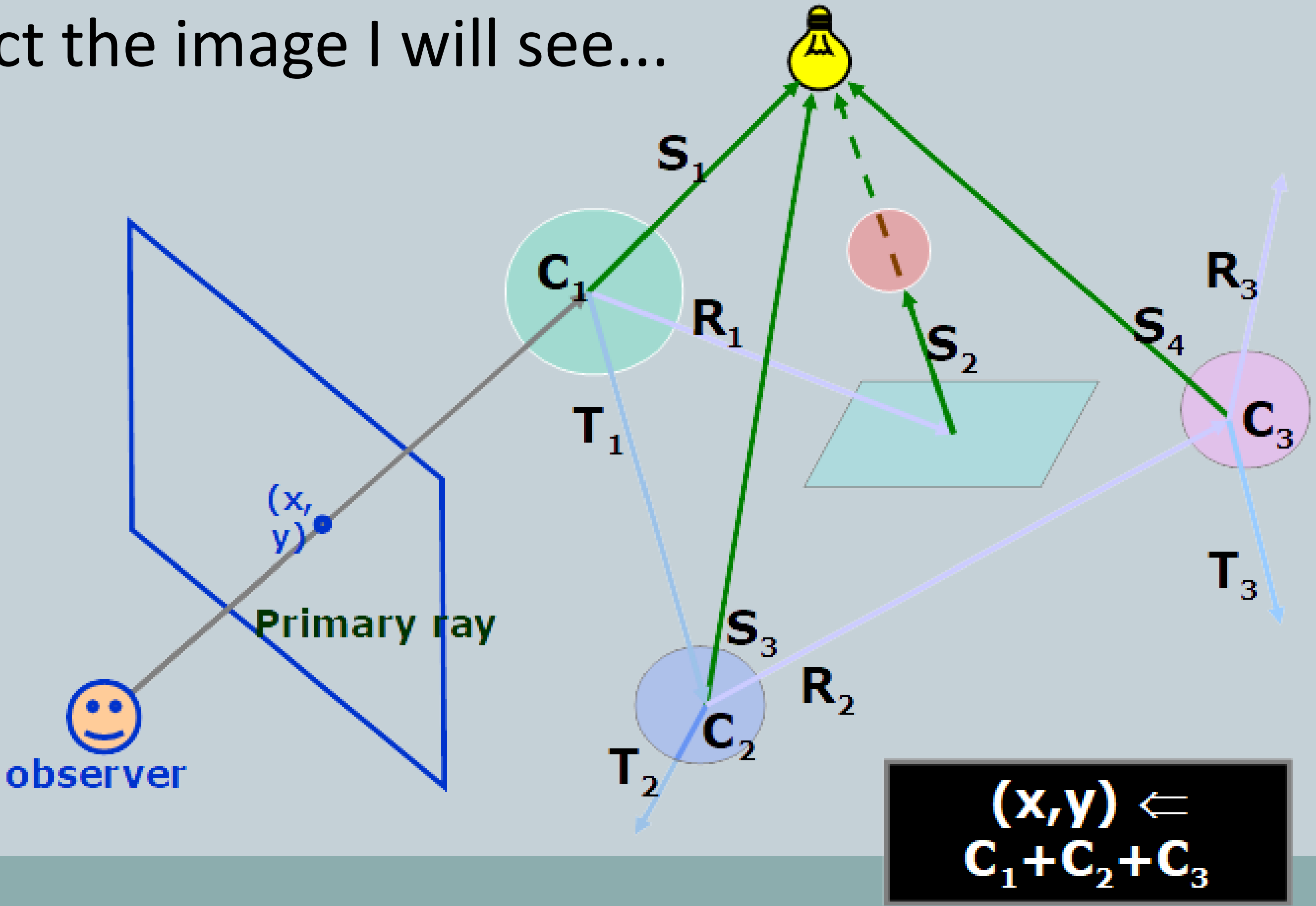
Offline Rendering

Light is what make us see, a better simulation of light behaviour is the key to photorealism...

The three main rendering methods try to simulate how light travels in the scene, follow the physical nature of light.

Ray Tracing / Path Tracing

Light travel in straight rays, following strict geometrical (optical) rules: if I follow back each ray entering my eyes, I can reconstruct the image I will see...



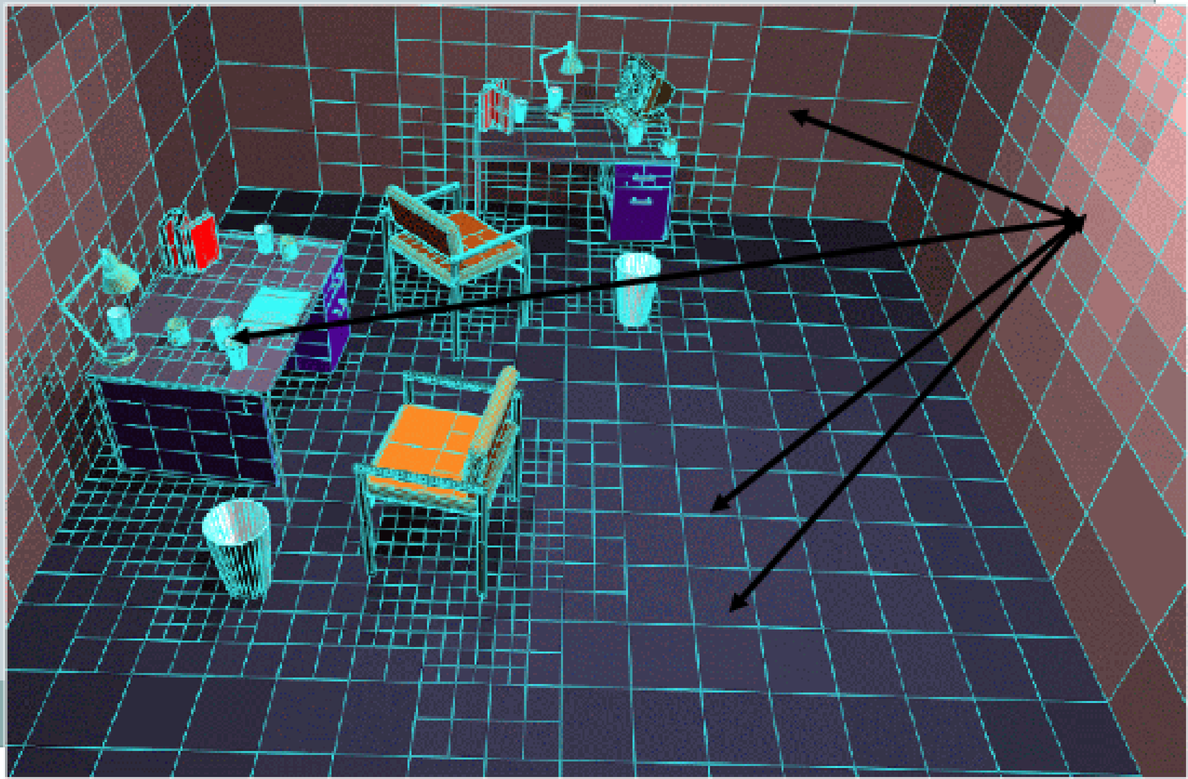
$(x,y) \Leftarrow C_1+C_2+C_3$

Ray Tracing / Path Tracing

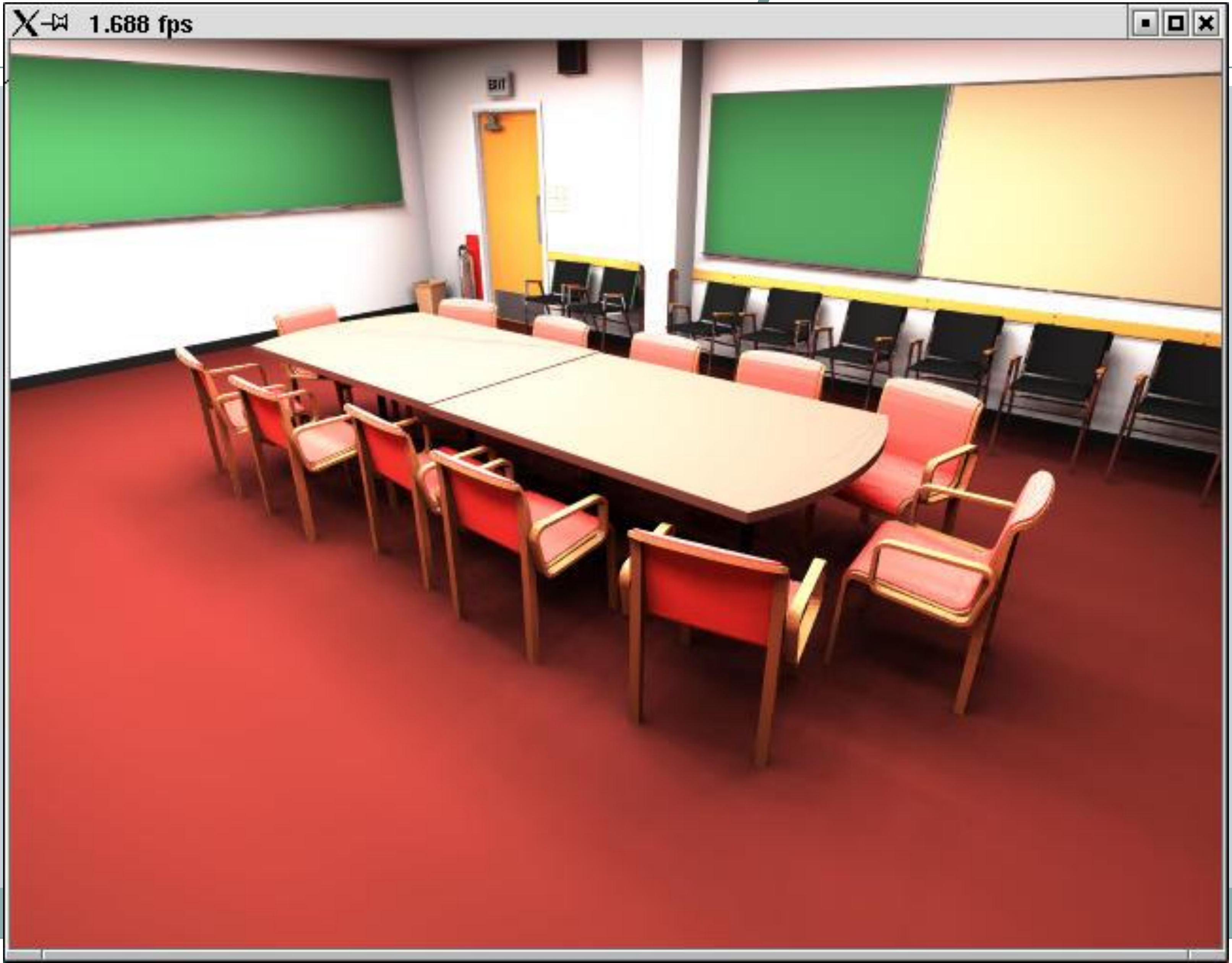
Linux
Modélisation/Rendu
André Pascual

Radiosity

Light is energy and, when emitted from a light source is just bounced/absorbed/transmitted around the scene. Idea: divide the scene in patches, calculate how energy moves around the scene until no more energy remains...



Radiosity



Photon Mapping

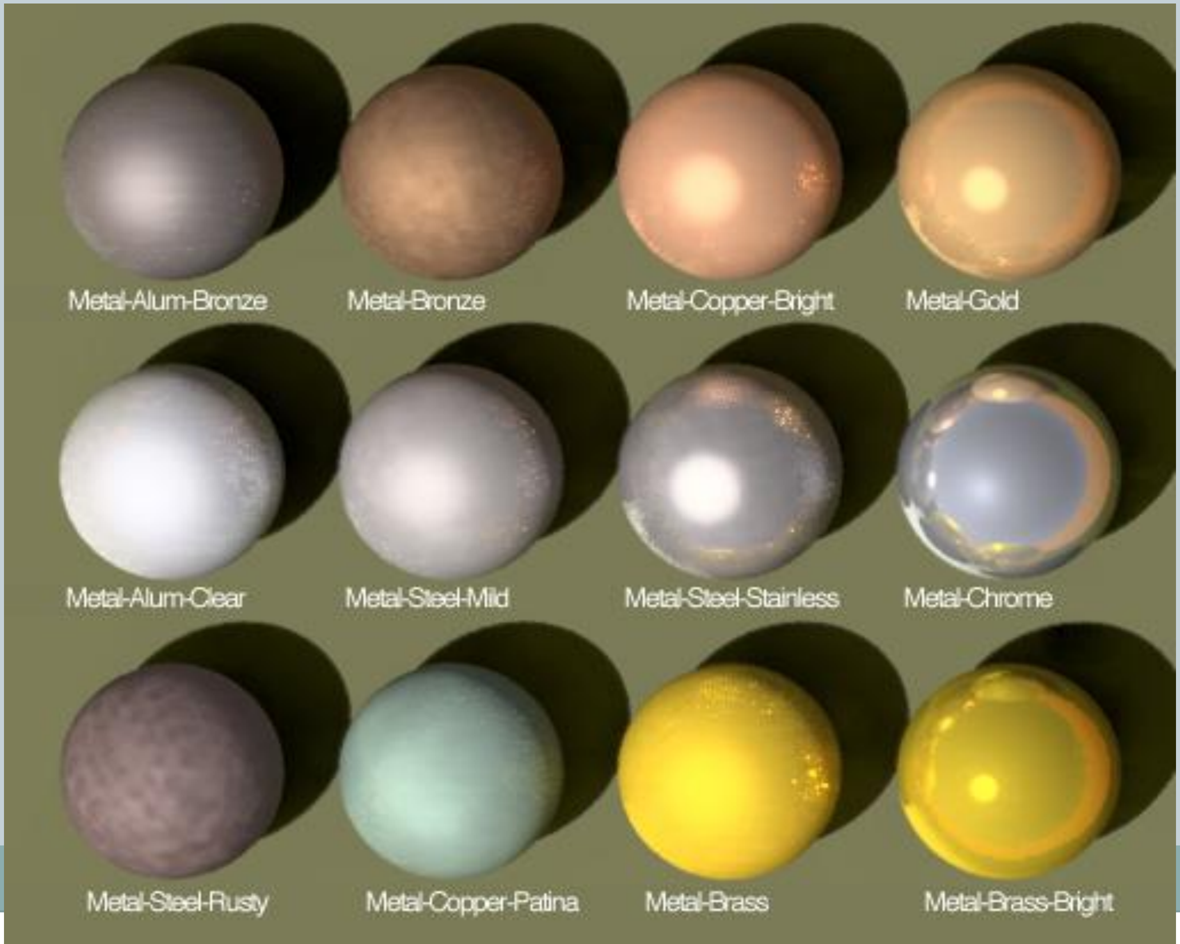
Light is energy carried by photons. let us scatter some (millions) photons in the scene and follow each one, then scatter ray from the camera, and see what goes where…

By your powers combined....

The various techniques started separate, but are converging. Most existing tool mix and match different strategies and algorithms, to combine the strengths of the various approaches...

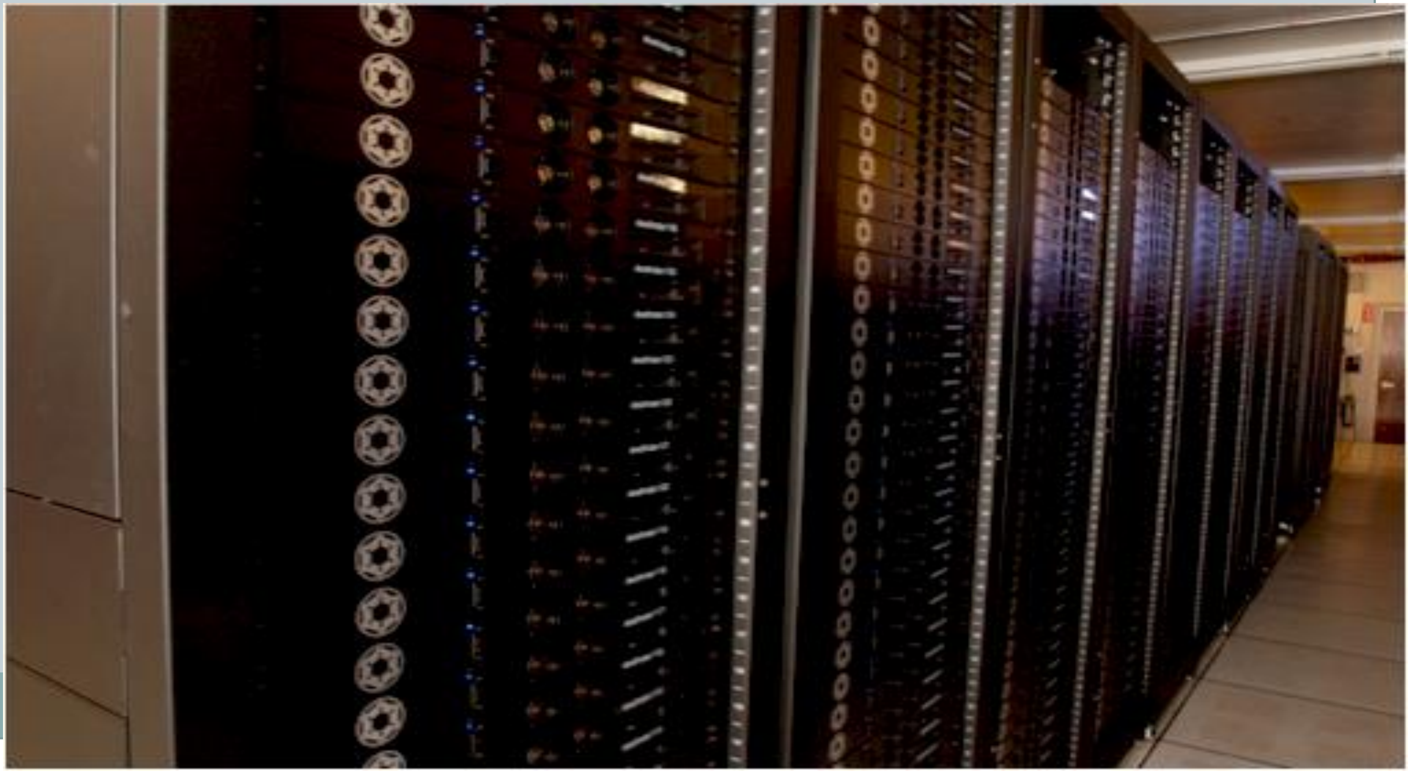
Not just Light(s)

I lied (again), the added realism is also due to much more complicated calculation on light-matter interaction...



Back to Realtime

The computational power required to follow these methods is too high for realtime, but many of the more accurate lighting and material calculations have been transferred from offline to realtime...



3D Video Cards

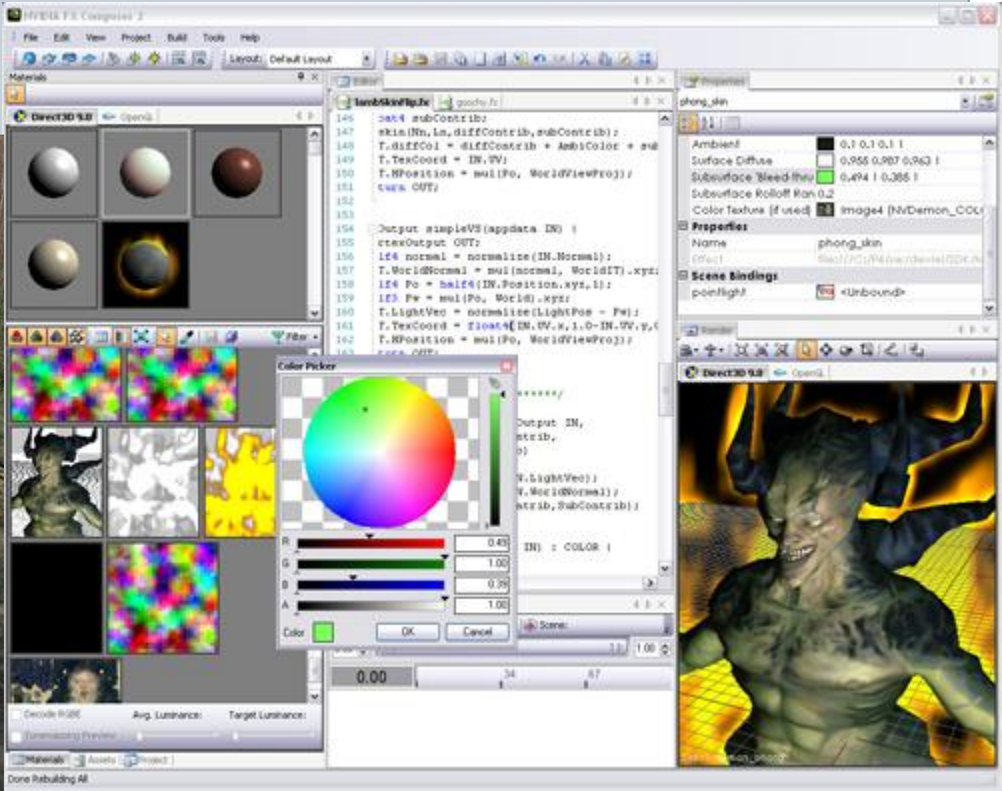
The first cause for the 3D explosion has been the availability of “accelerated“ 3D video cards.

First of all, all the mathematical calculation for scene setup and perspective projection are managed by the video card that, on this specific task, is much faster than the CPU...

But this does not explain why realtime 3D is nowadays so nice to see...

(modern) 3D Video Cards

Modern videocards are "programmable", that means it is possible to employ some of the lighting calculation and material simulation from the offline rendering methods, obtaining stunning results...

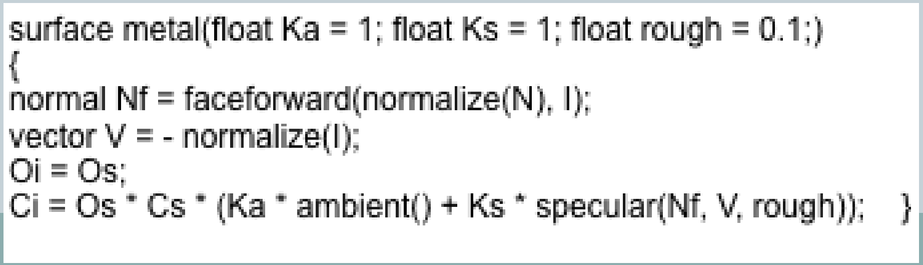


3D Video Cards - shaders

SHADER is another very common word in the modern 3D field.

A shader is a piece of code which is used to specify some aspect of the rendering, from the geometry, to the screen projection, from the light-material interaction to the final image filtering.

Most hardware, today, is shader-based (especially mobile hardware).



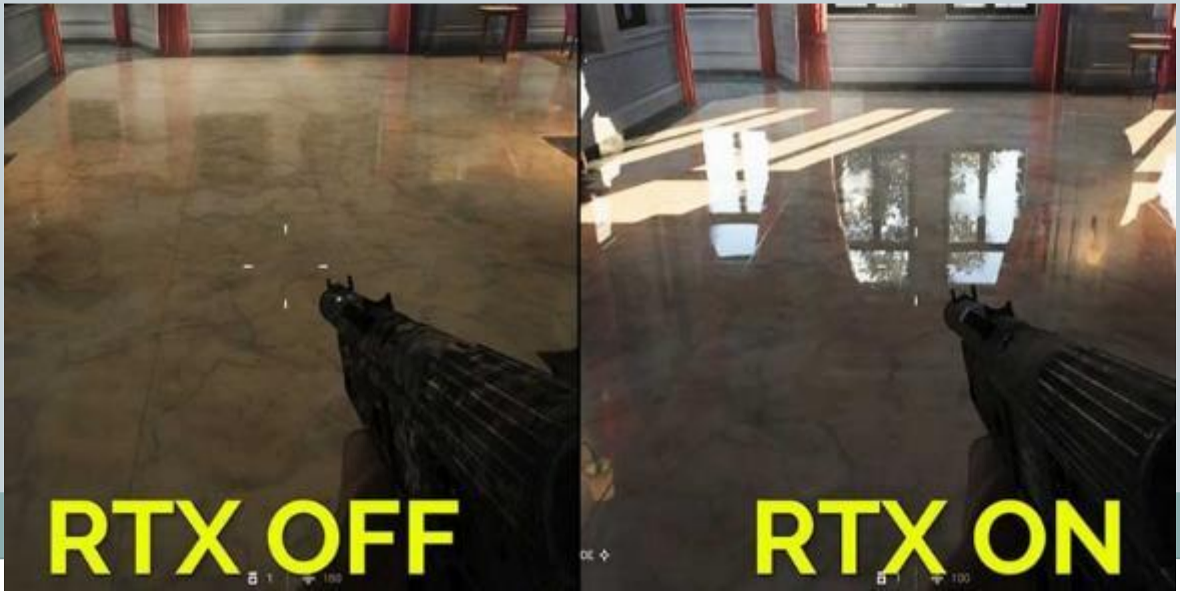
```
surface metal(float Ka = 1; float Ks = 1; float rough = 0.1;)
{
normal Nf = faceforward(normalize(N), I);
vector V = - normalize(I);
Oi = Os;
Ci = Os * Cs * (Ka * ambient() + Ks * specular(Nf, V, rough));    }
```

3D Video Cards – ray tracing

The current craze is ACCELERATED ray/path tracing.

Recurring operations in ray tracing are calculated in parallel by the GPU hardware: realtime interaction with a CG movie –like quality in reflections/refractions and shadows.

A lot of limitations, and most of the illumination effects are still tricks… nevertheless, impressive



Question Time

INTERRUPT ME ANYTIME FOR QUESTIONS

callieri@isti.cnr.it