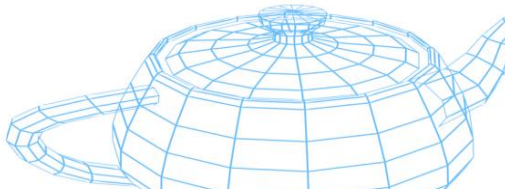
SUPSI

Computer Graphics

Ray tracing VS rasterization

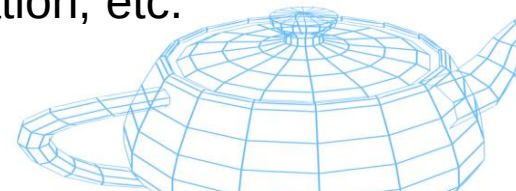
Achille Peternier, adjunct professor

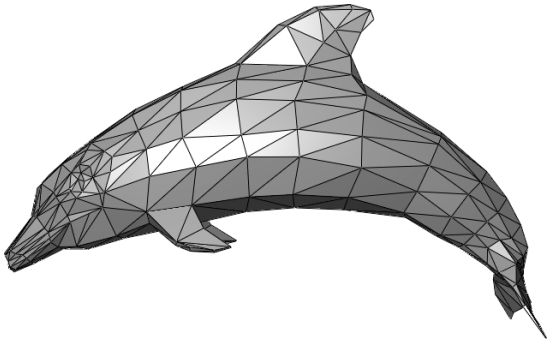


Offline rendering VS real-time

- Offline rendering:
  - Used to generate images with the highest achievable quality:
    - *"Take all the time you need to do a good job"*-philosophy.
  - Time required is not critical.
  - Typical scenarios: movies, photorealism, architectural rendering, etc.

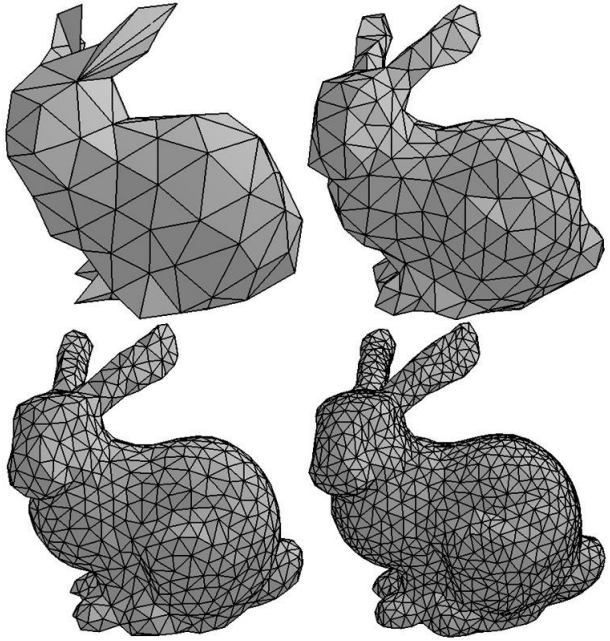
- Real-time rendering:
  - Several images generated per second:
    - Frames per second (**fps**), or **framerate**.
  - Real-time requires at least 24 fps:
    - Interactive rendering is about 1-5 fps.
  - Typical scenarios: simulation, Computer Aided Design (CAD), games, Augmented Reality (AR), any human-interactive application, etc.

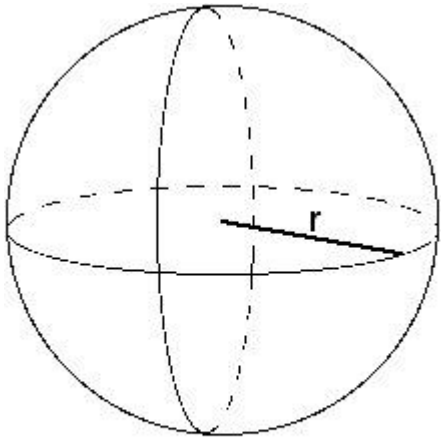




3D modeling

- Mathematical representation of the surface of an object:
  - Using formulae for well-defined objects, e.g., lines, planes, spheres, etc.
  - Using a series of polygons (mainly triangles or quadrilaterals) to reconstruct the original shape.





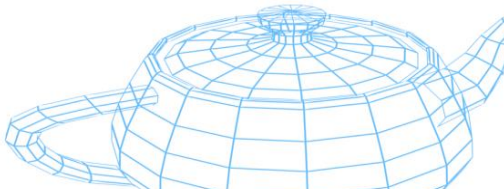
$$\text{Volume} = \frac{4}{3}\pi r^3$$

Polygon mesh

"A **polygon mesh** is a collection of vertices, edges and faces that defines the shape of a polyhedral object in 3D computer graphics and solid modeling. The faces usually consist of triangles (triangle mesh), quadrilaterals, or other simple convex polygons, since this simplifies rendering, but may also be composed of more general concave polygons, or polygons with holes."

(Wikipedia)

- Synonyms (or equivalent, depending on the context) are:
  - Mesh.
  - Object.
  - Model.
  - Primitive.



Ray casting (1968)

- Arthur Appel, IBM.
- Precursor to ray tracing.

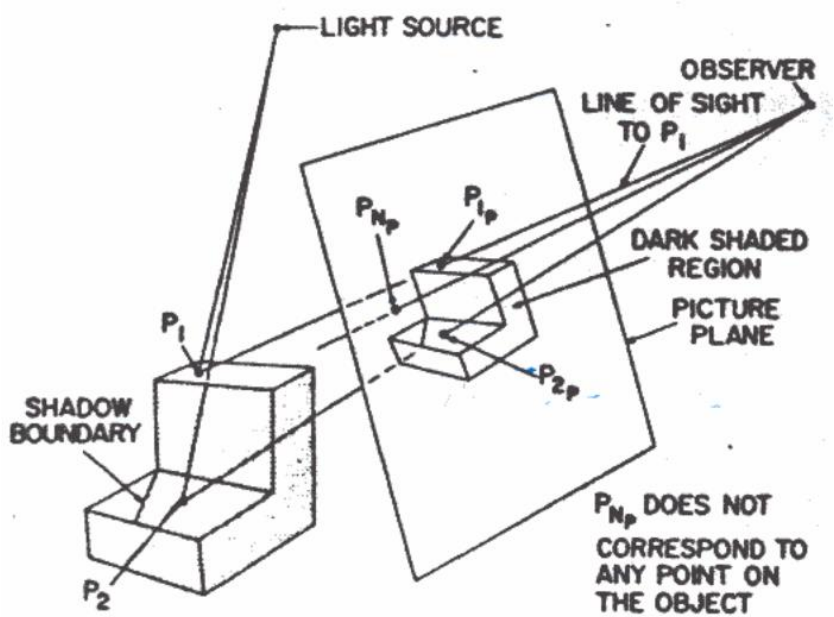


Figure 6 – Point by point shading

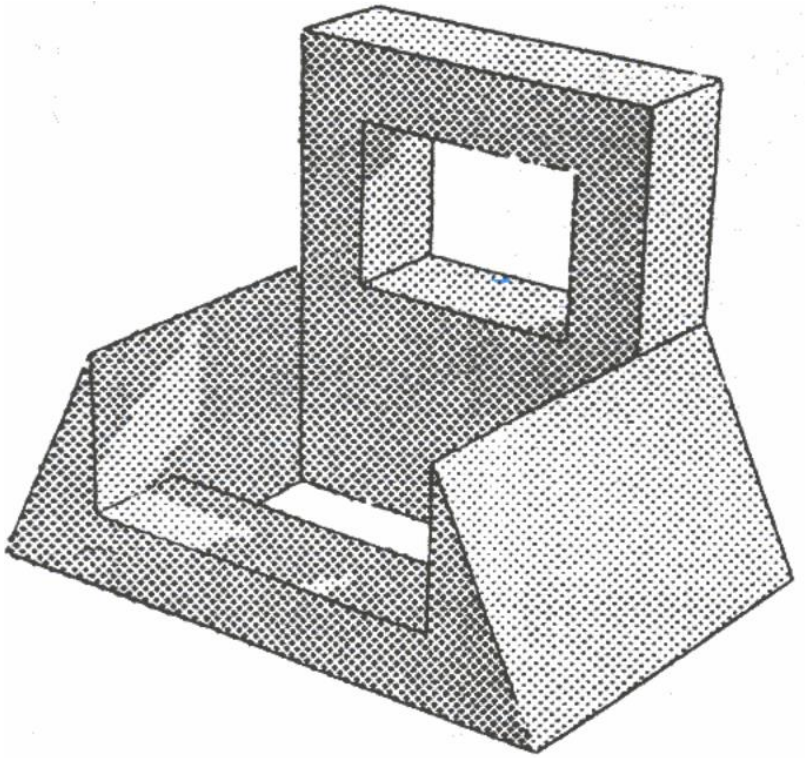
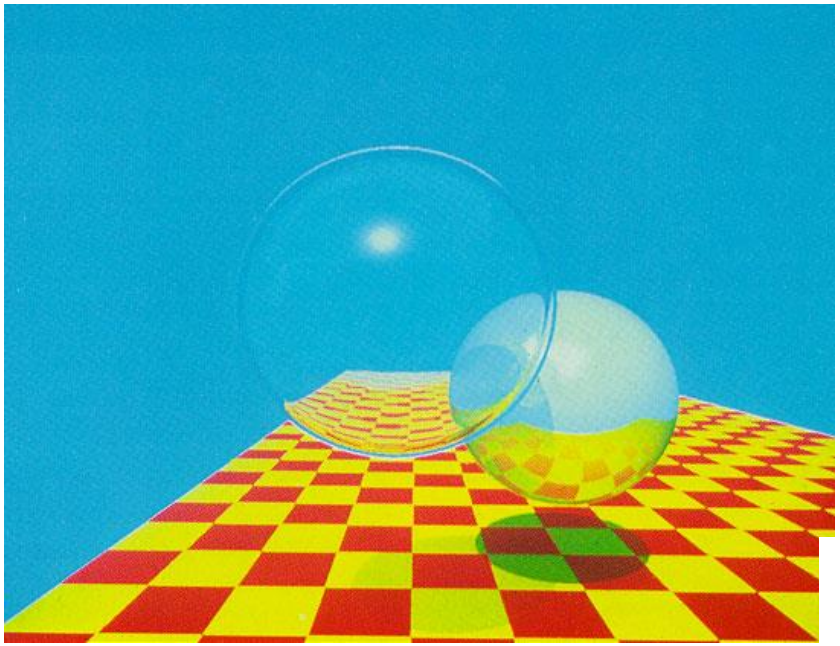


Figure 14 – Another view of the machine part shown in the previous figure. The light source has been moved relative to the object. Notice the light passing through the opening in the object

[A. Appel, *Some techniques for shading machine renderings of solids*, In proc. of AFIPS, 37-45, 1968]

Ray tracing (1980)

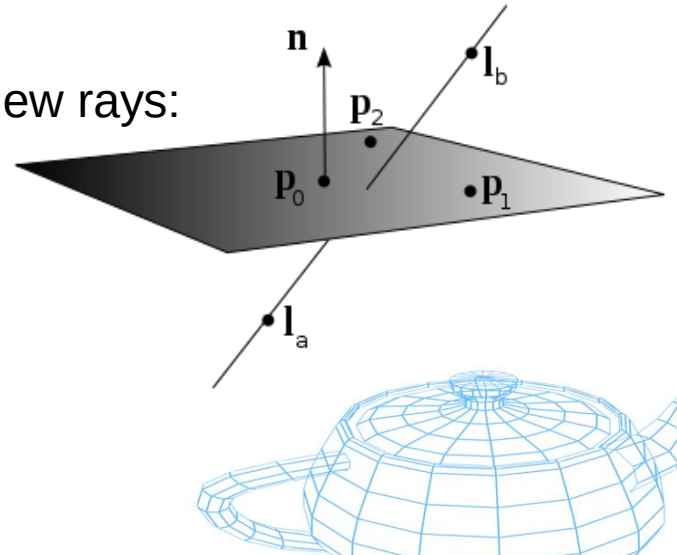
- Turner Whitted, Bell Labs.
- Unified approach integrating reflection, refraction, and shadows.



Ray tracing

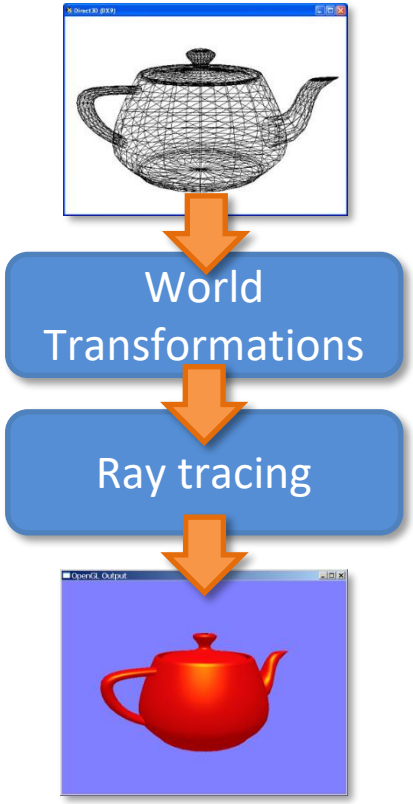
- Rays are casted from the camera into the scene:
  - Unlike reality, where photons hit the camera's sensor:
    - Light-tracing VS eye-tracing.
  - More efficient:
    - It reduces the number of wasted photons.

- For each ray, multiple intersection tests are performed:
  - Ray/plane, ray/sphere, ray/triangle, etc.
  - Each time you hit an object, cast one or more new rays:
    - Recursive approach.
    - Give-up after N iterations.

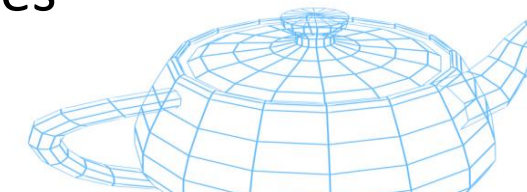


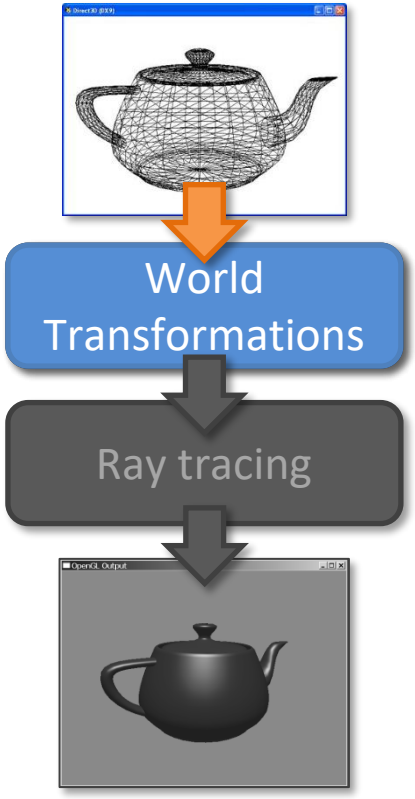
3D primitives

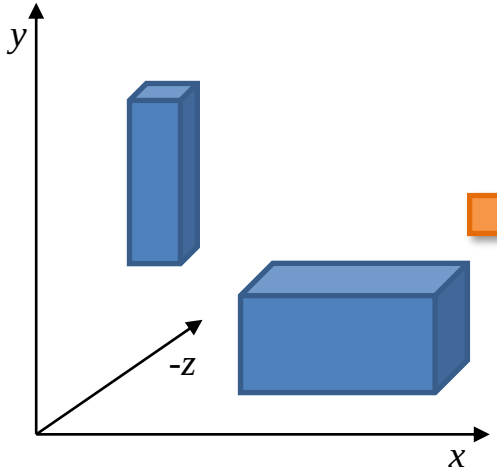
Rendering pipeline



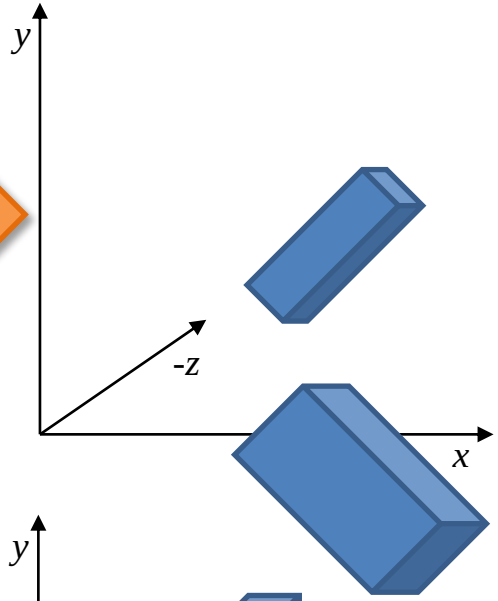
2D images

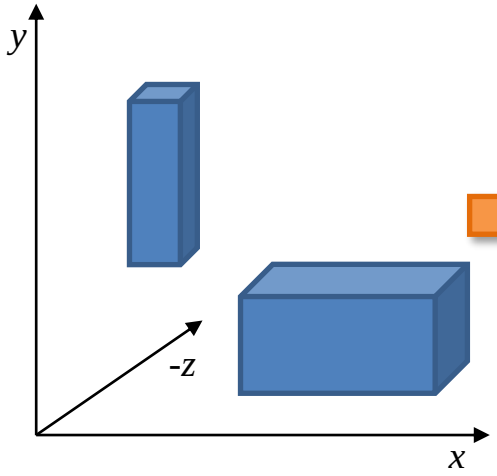




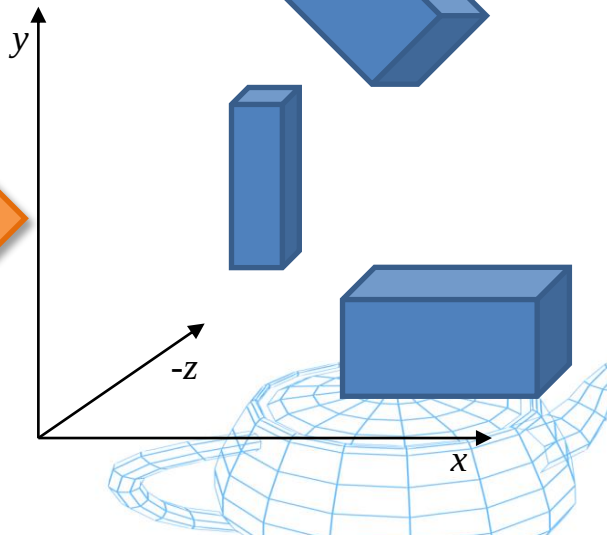


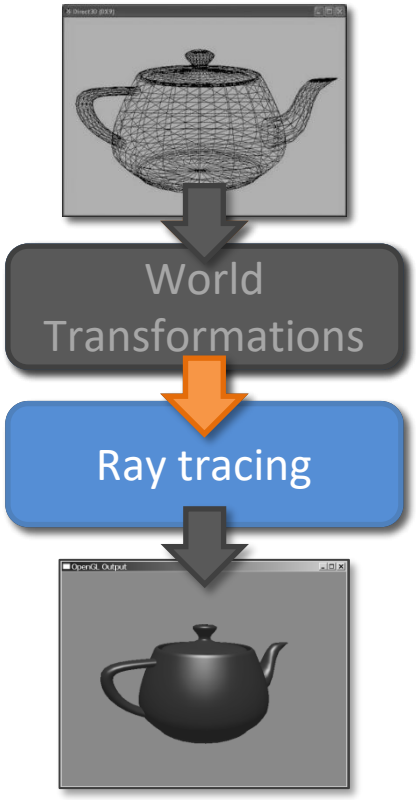
RotateZ -45°





TranslateX 5




World
Transformations
Ray tracing

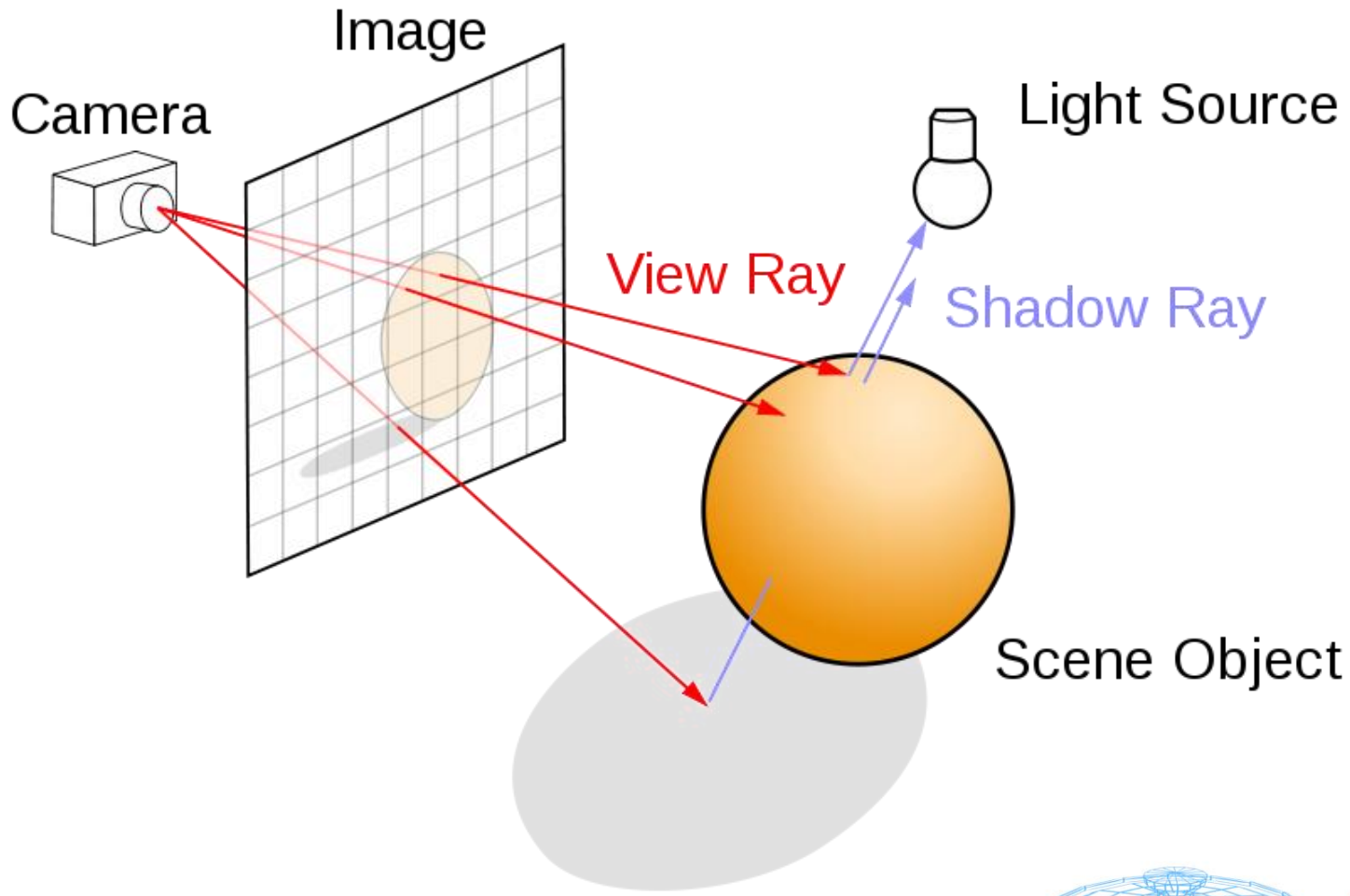
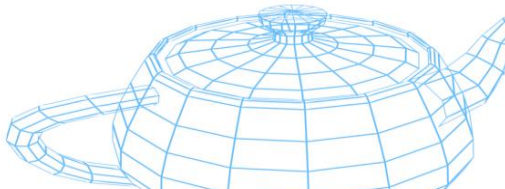
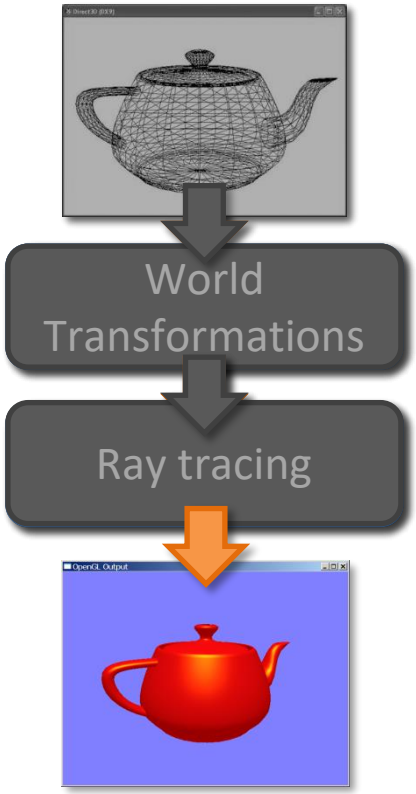
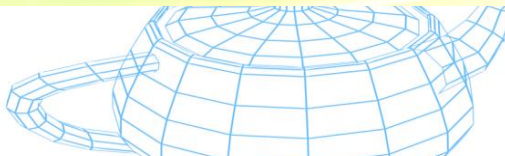


Image
Camera
Light Source
View Ray
Shadow Ray
Scene Object

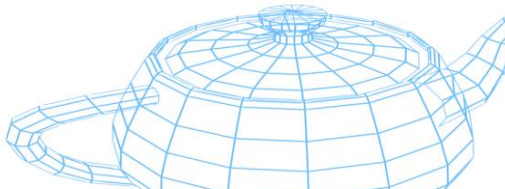



World
Transformations
Ray tracing



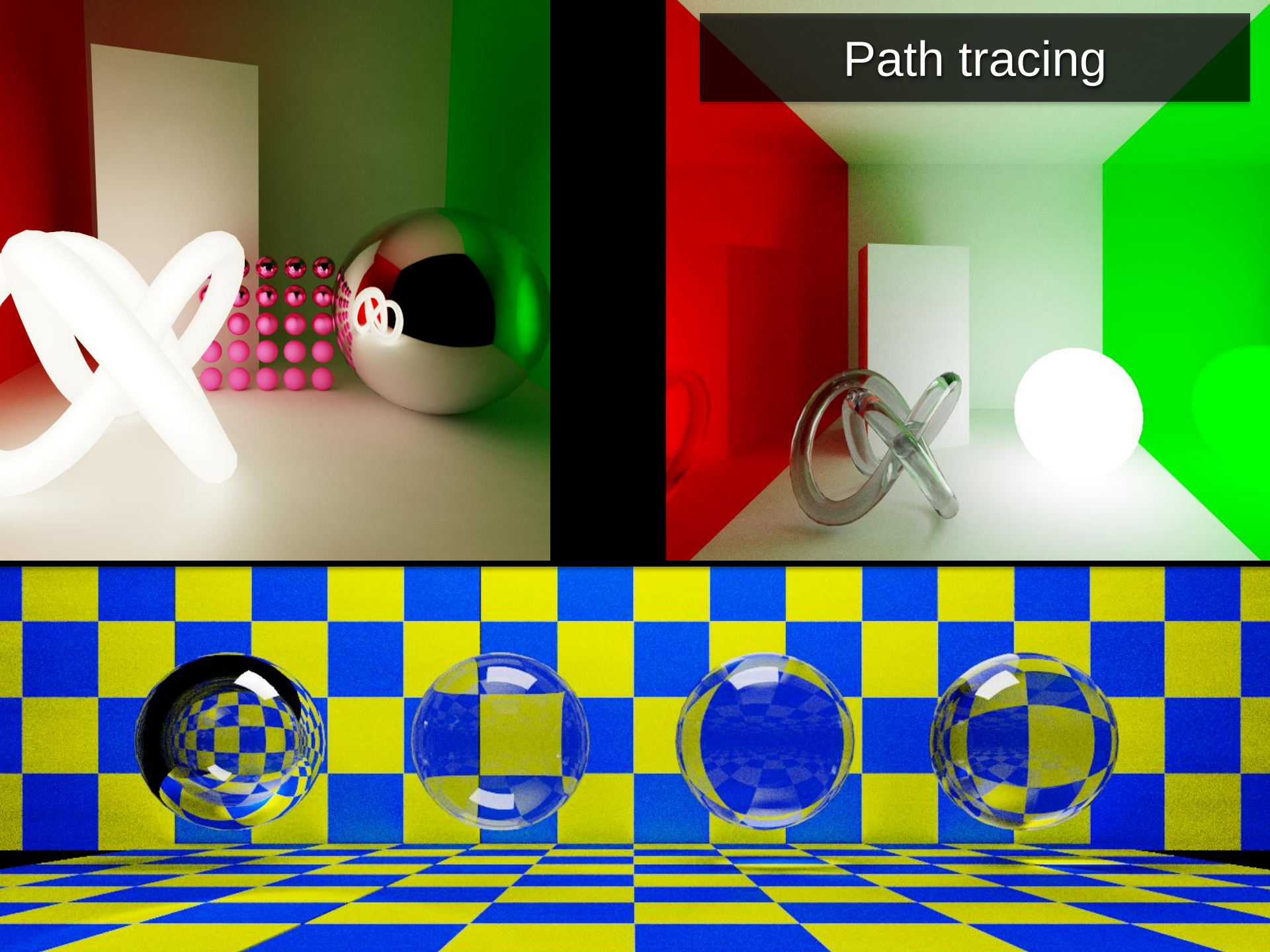
Ray tracing

- Ray tracing is a generic term covering several different techniques, such as:
  - Ray casting (Appel, 1968):
    - one ray from eye into the scene.
    - Stops at the first intersection.
  - Recursive ray tracing (Whitted, 1980):
    - For each collision, three new rays are recursively generated (refraction, reflection, shadow).
  - Path tracing (Kajiya, 1986):
    - Multiple pseudo-random (Monte Carlo) simulations of the way photons propagate into the scene to approximate the rendering equation.
  - …



Recursive ray casting

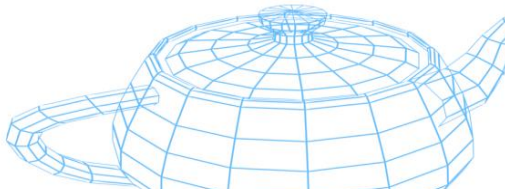
Path tracing


Path tracing

Ray tracing

- Advantages:
  - High level of realism.
  - Realistic simulation of the way light works.
  - Correct transparency, reflections, and shadows are automatically generated.
  - Easy to parallelize.
  - Intuitive and easy to implement.

- Online example: https://www.shadertoy.com/view/ttfyzN

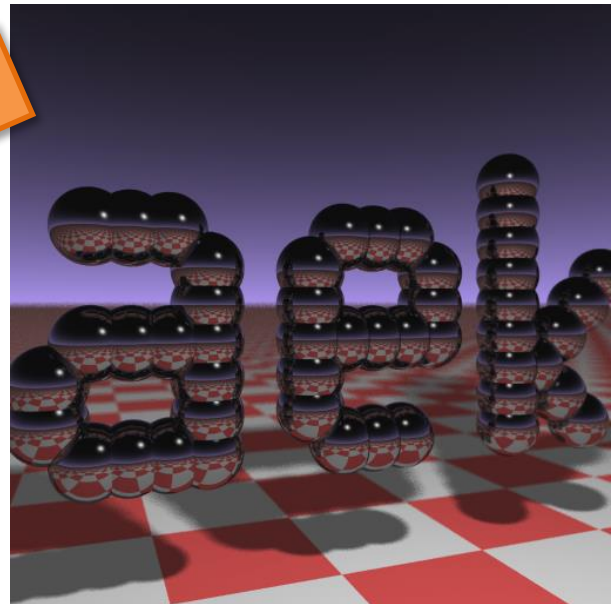


Ultra-compact ray tracer example

- See Andrew Kensler's card ray tracer: https://fabiensanglard.net/rayTracing_back_of_business_card/

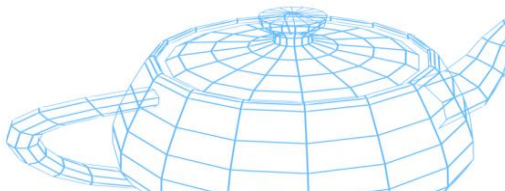
- To compile and run it (under Linux) copy/paste the code into card.cpp, then:

```
g++ -O3 -o card card.cpp
./card > card.ppm
```



Ray tracing

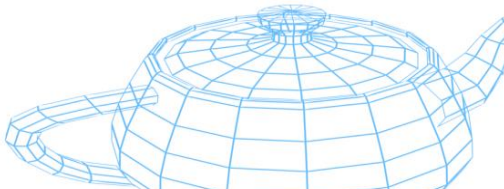
- Disadvantages:
  - Computationally expensive → slow!
  - Not GPU-friendly:
    - Not because of ray tracing *per se* but because GPUs have been optimized for a different technique.
  - Things are changing?
    - HW acceleration for ray tracing in modern GPUs.
    - Nvidia OptiX ray tracing engine:
      - GPGPU, programmable GPUs, OpenCL, etc.
    - OpenRL, DirectX Raytracing (DXR), Vulkan RT extensions:
      - API (kind of OpenGL for ray tracing).
    - Hybrid approaches (e.g., for global illumination).



Ray tracing

- Popular ray tracers:
  - POV-Ray.
  - Pixar's RenderMan (hybrid since "Cars").

- Real-time ray tracing examples:
  - REMRT/RT (1986).
  - Intel's Quake Wars: Ray Traced (2008):
    - 15-20 fps on server-level machines:
      - dual/quad-CPU models.
      - priced higher than 15.000 CHF.
  - Nvidia RTX (2018).




QUAKE WARS
RAYTRACED

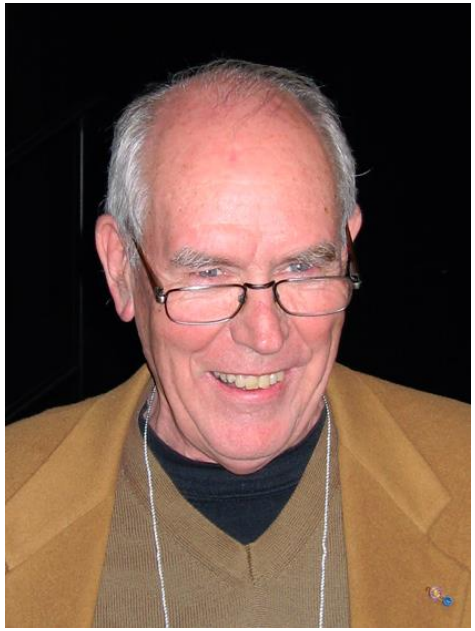
https://www.youtube.com/watch?v=J3ue35ago3Y

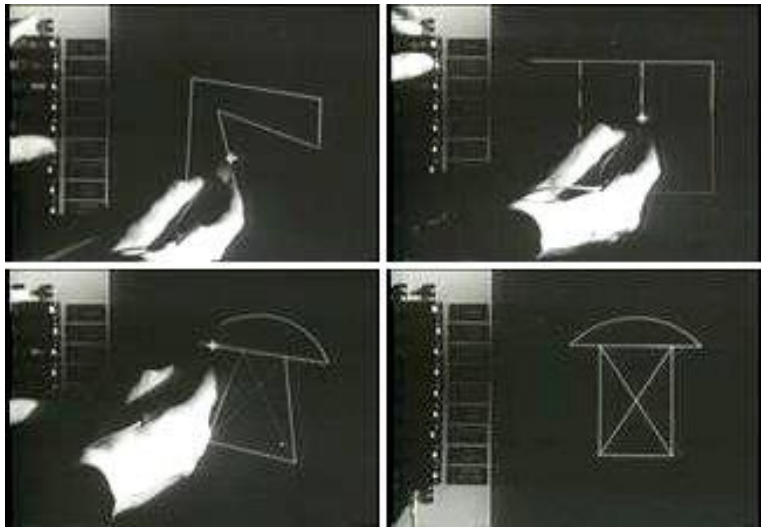
Rasterization rendering pipeline

Sketchpad (1962)

- Ivan Sutherland, MIT.
- Generally considered as the origins of Computer Graphics.

Ivan Edward Sutherland
1938

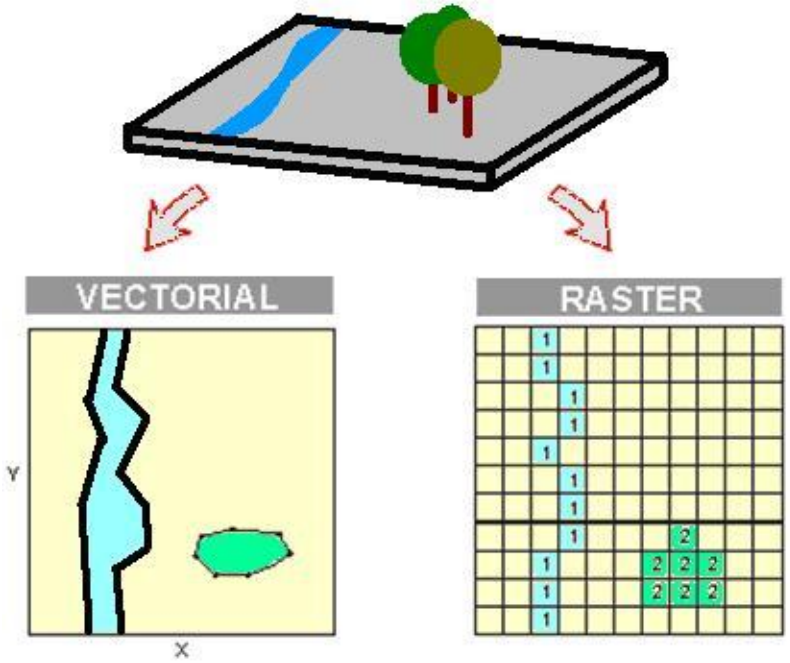




https://www.youtube.com/watch?v=57wj8diYpgY

The framebuffer (1969)

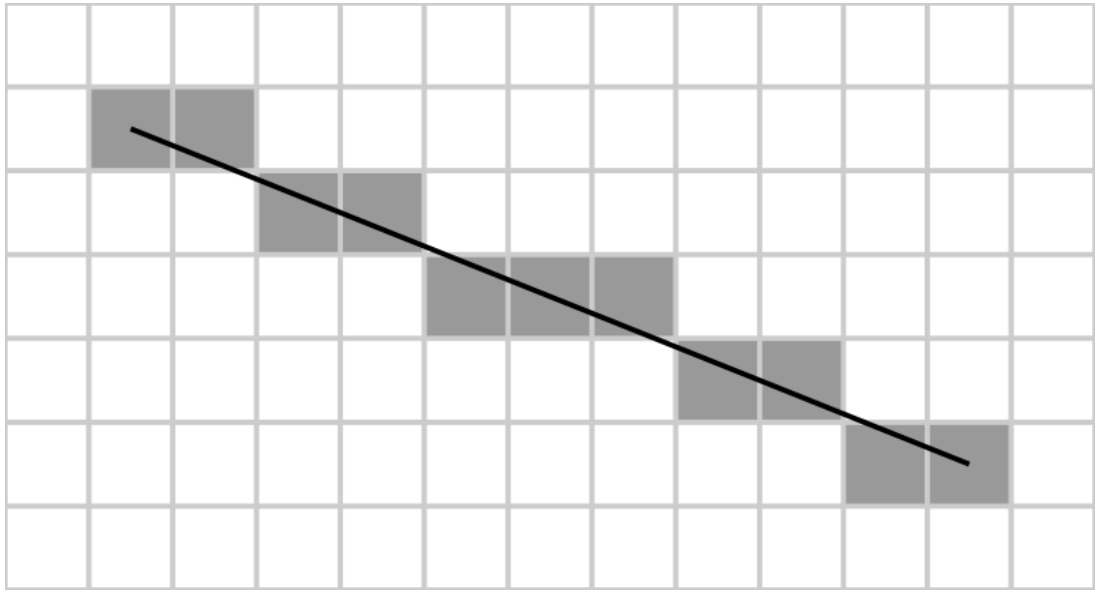
- Michael Noll, Bell Labs.
- Transition from vector to raster graphics (on Cathode-Ray Tube (CRT)).



(image by Dr. Pau Fonseca I Casa)

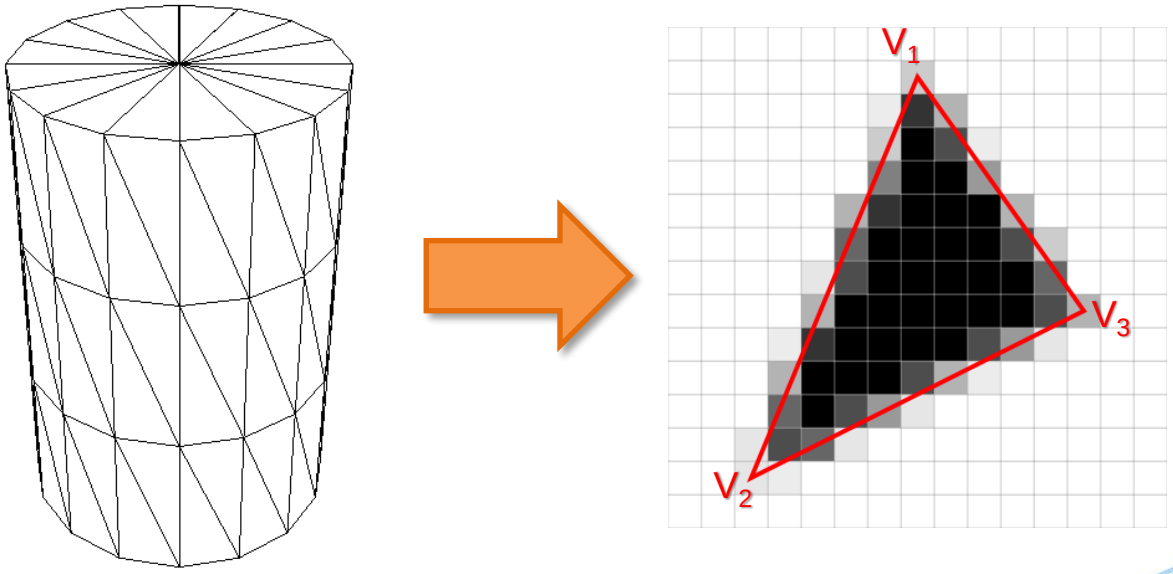
Line drawing algorithm (1965)

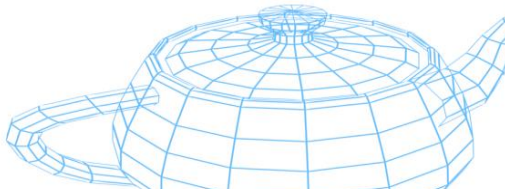
- Jack Bresenham, IBM.
- It only requires integer operations (addition, subtraction, and bit shifting).

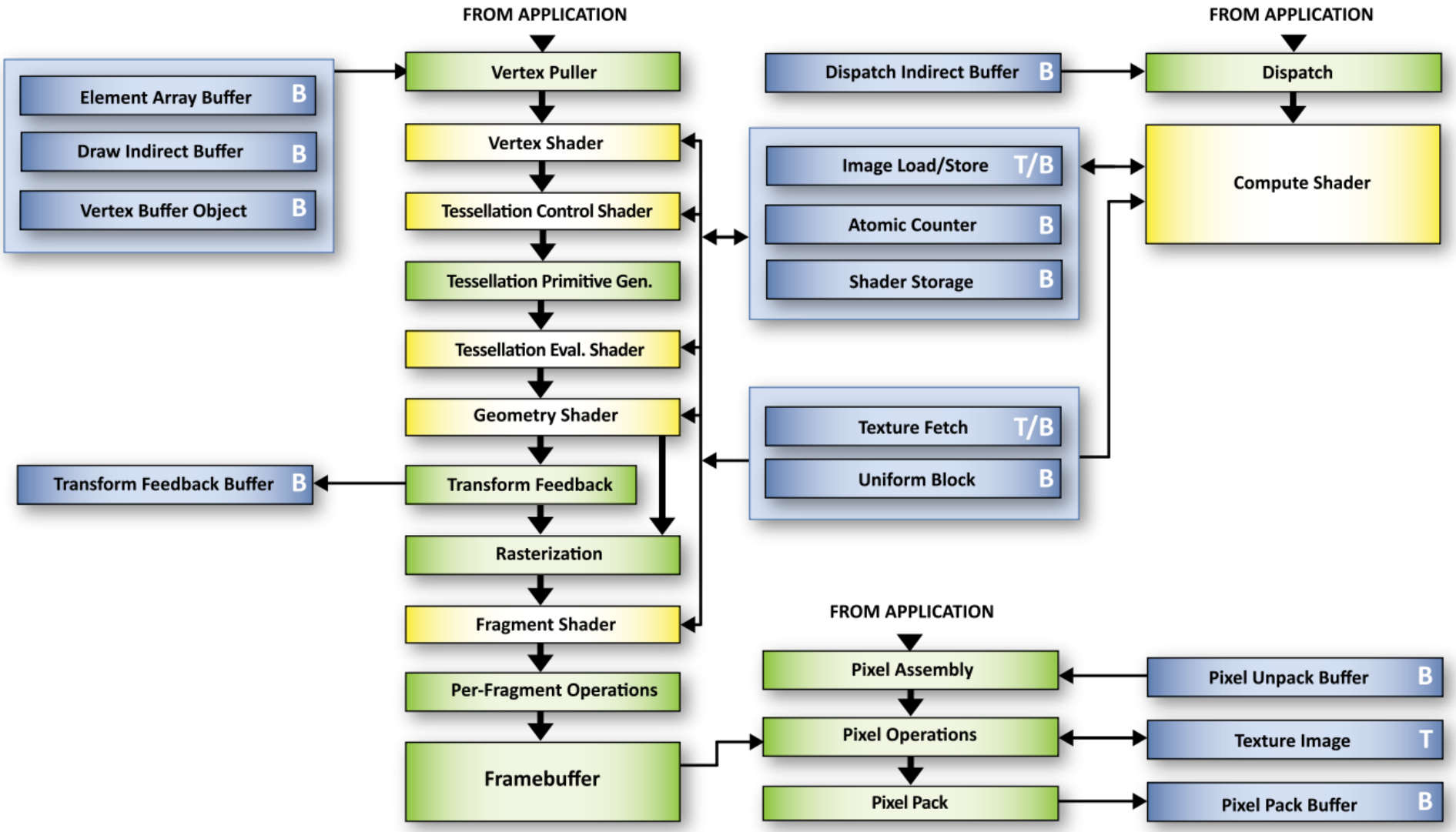


Rasterization

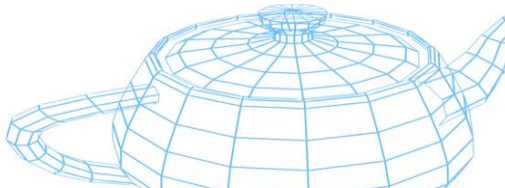
- 3D scenes are designed using points, lines, triangles, and other primitives:
  - Their vertices are processed independently.
  - Their vertices are projected into screen points and connected to draw lines and triangles.



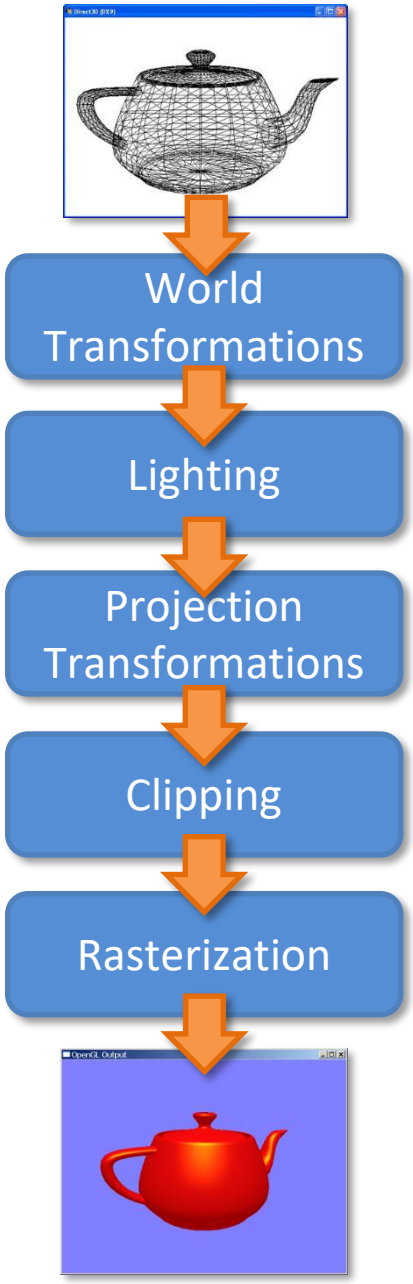




(OpenGL 4.5 pipeline)

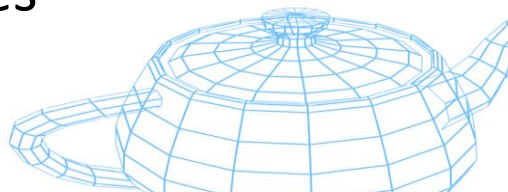


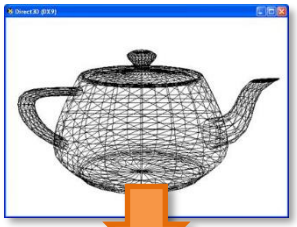
Rendering pipeline



3D primitives

2D images





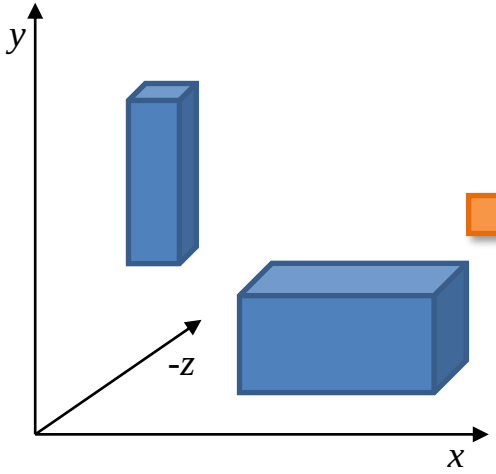
World Transformations

Lighting

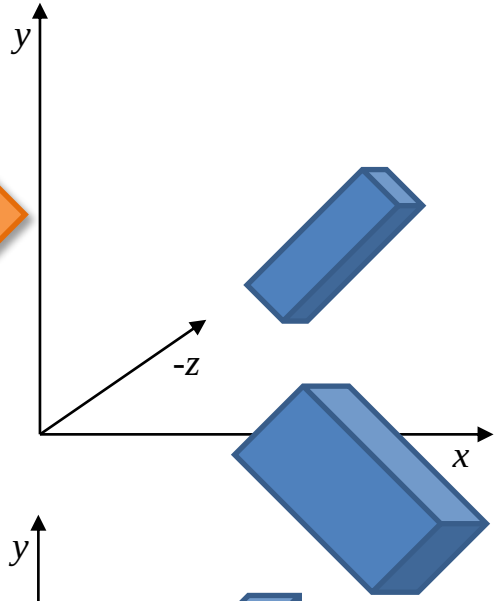
Projection Transformations

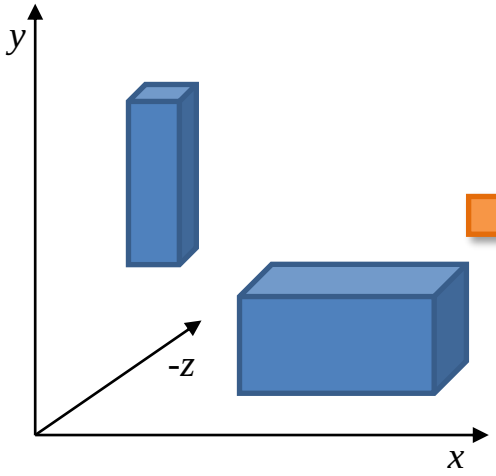
Clipping

Rasterization

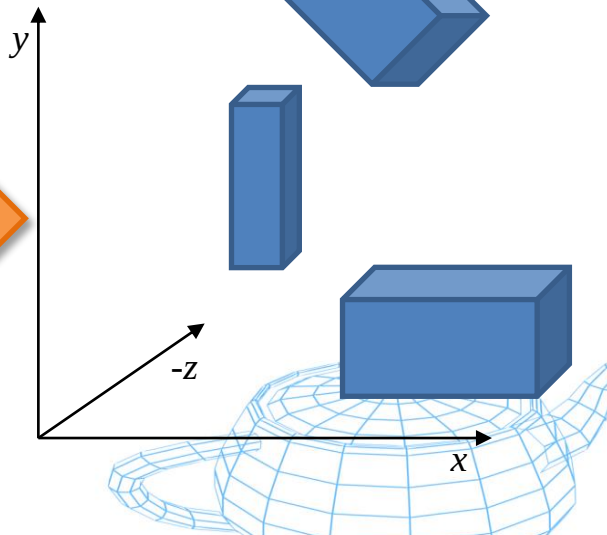


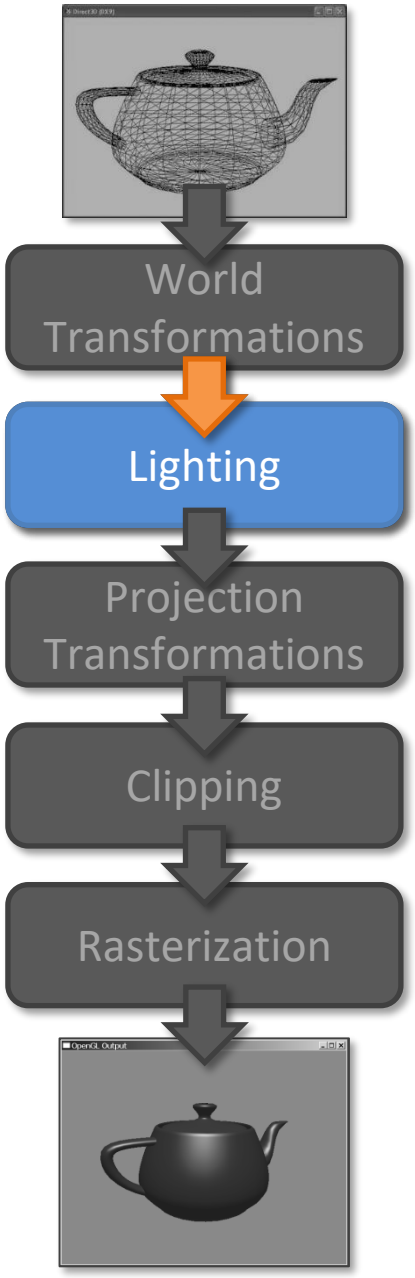
RotateZ -45°



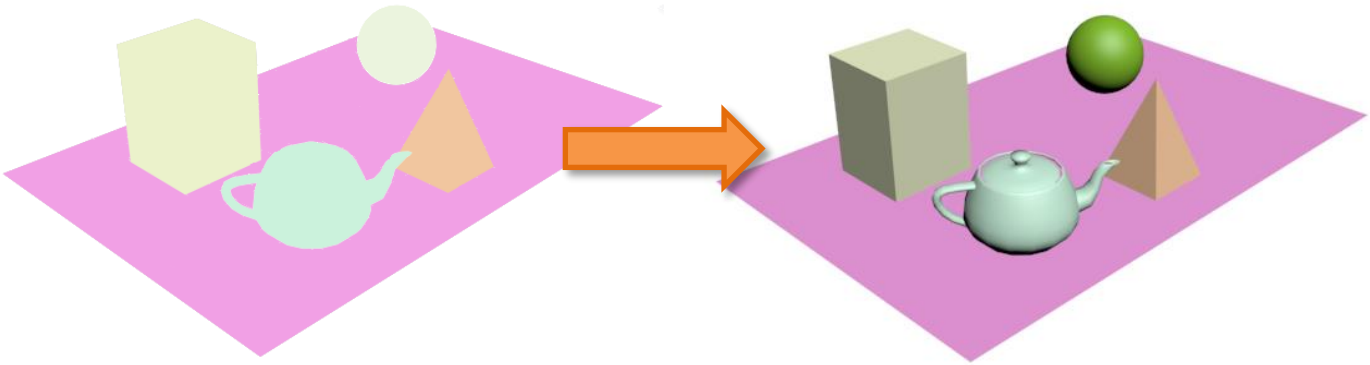


TranslateX 5

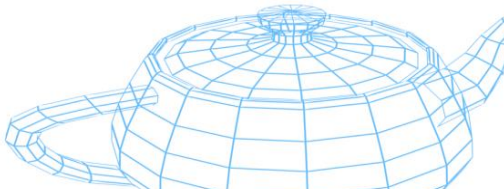


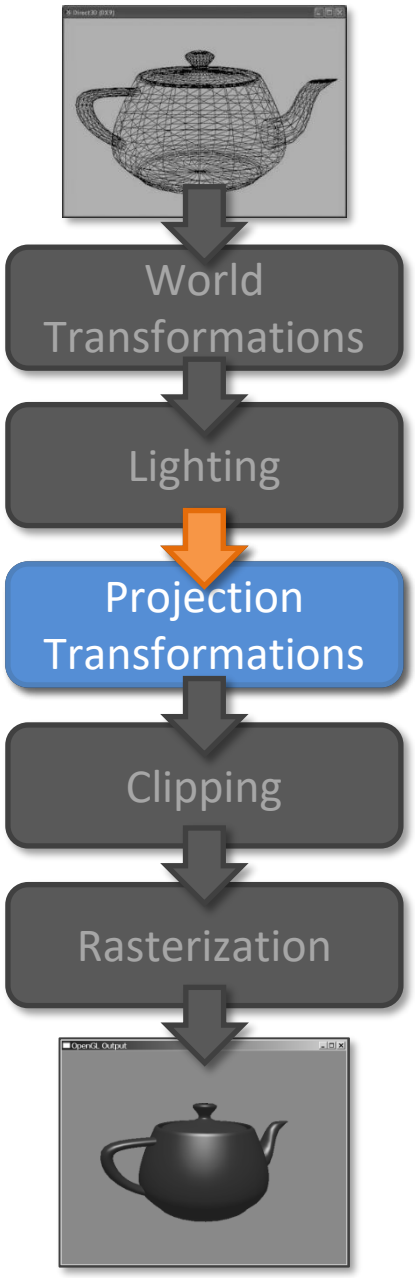


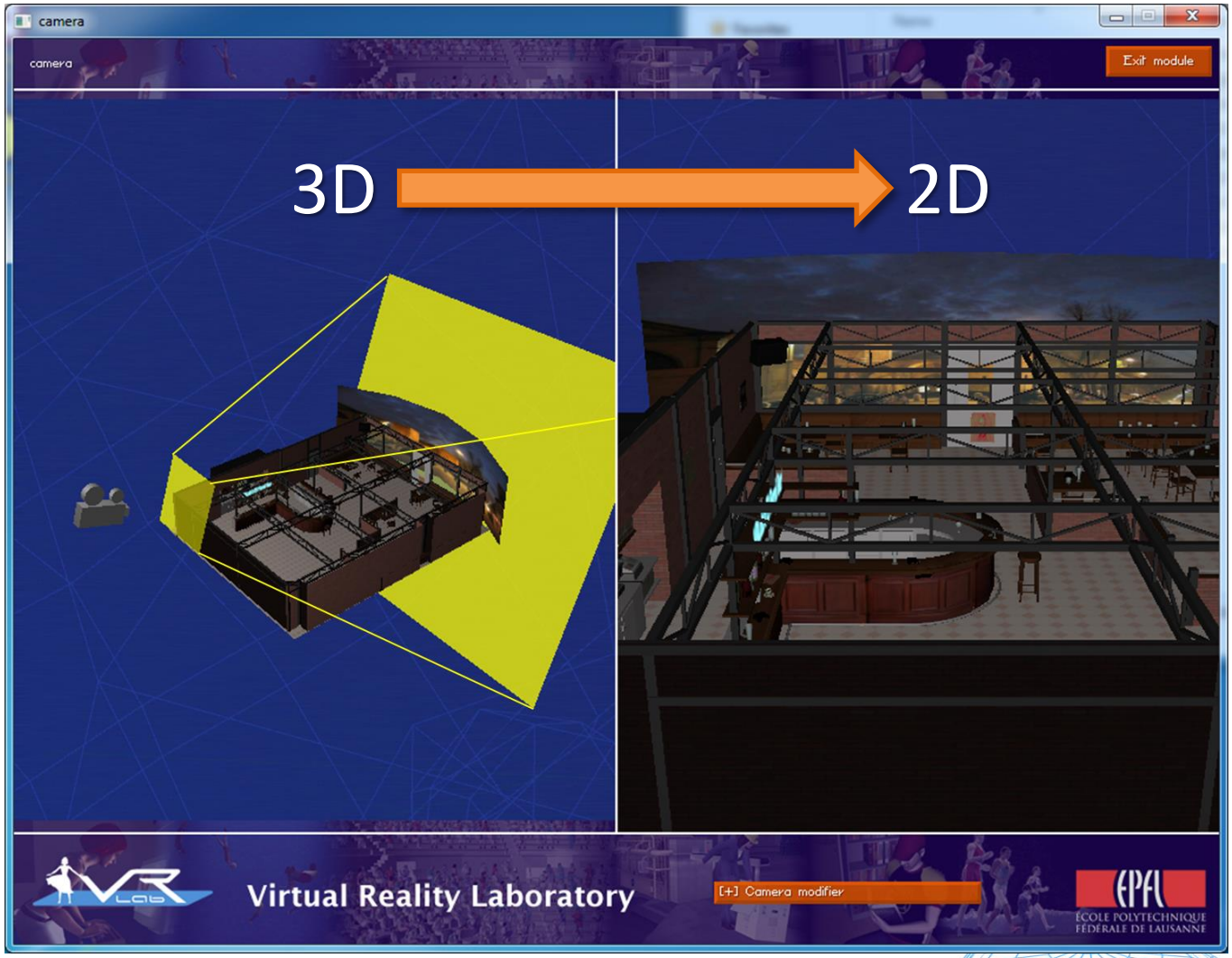
(direct illumination)

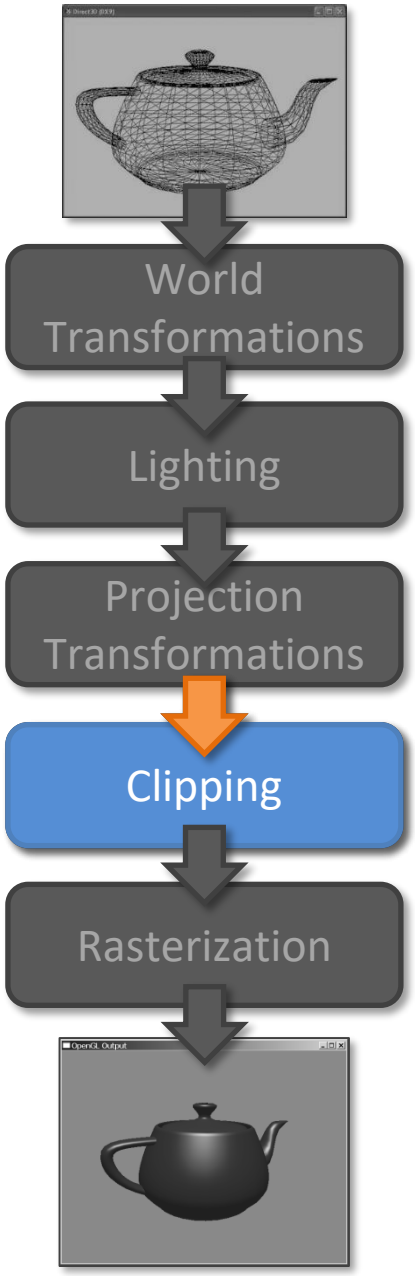


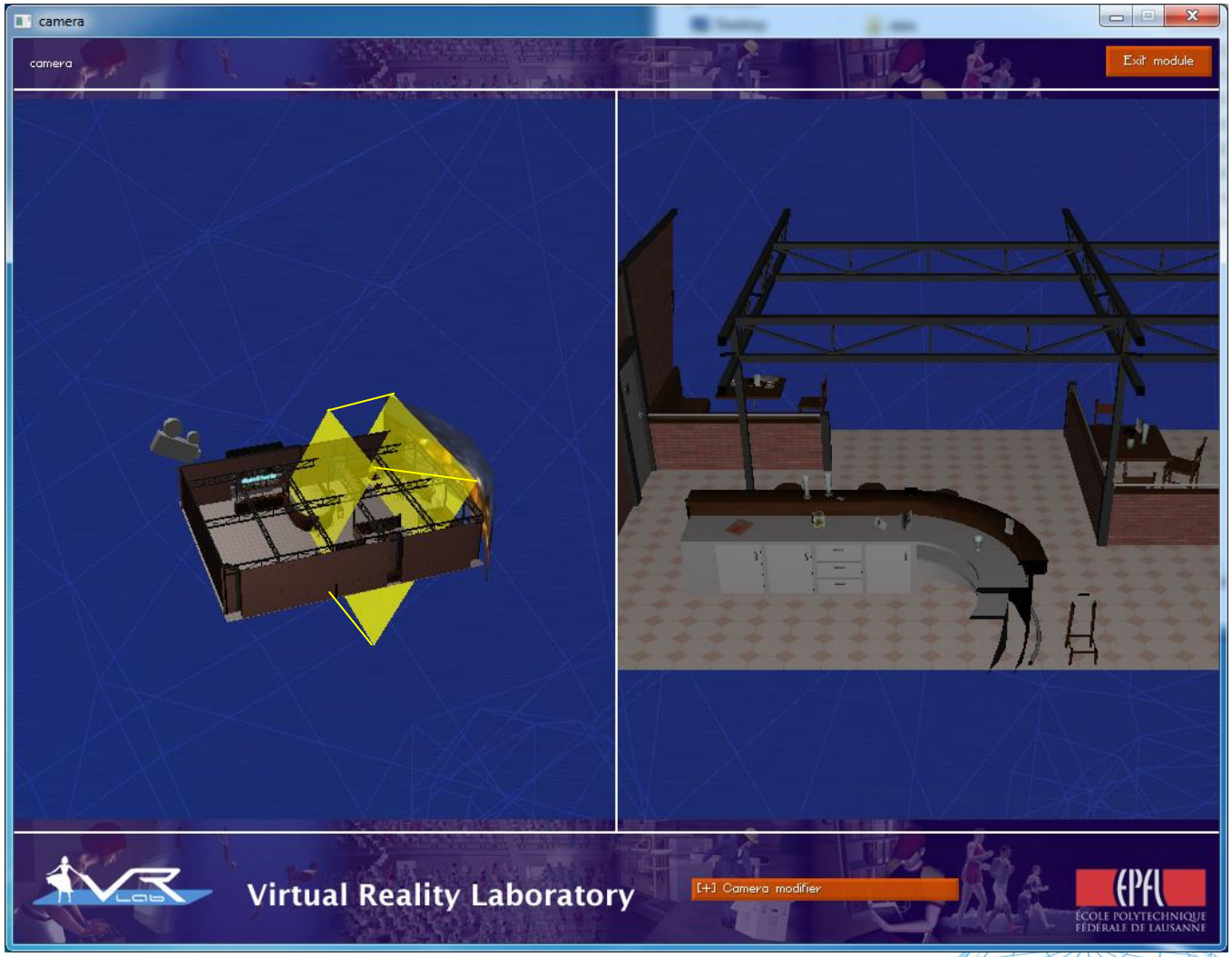
- No shadows
- No reflections
- No transparency
- No global illumination

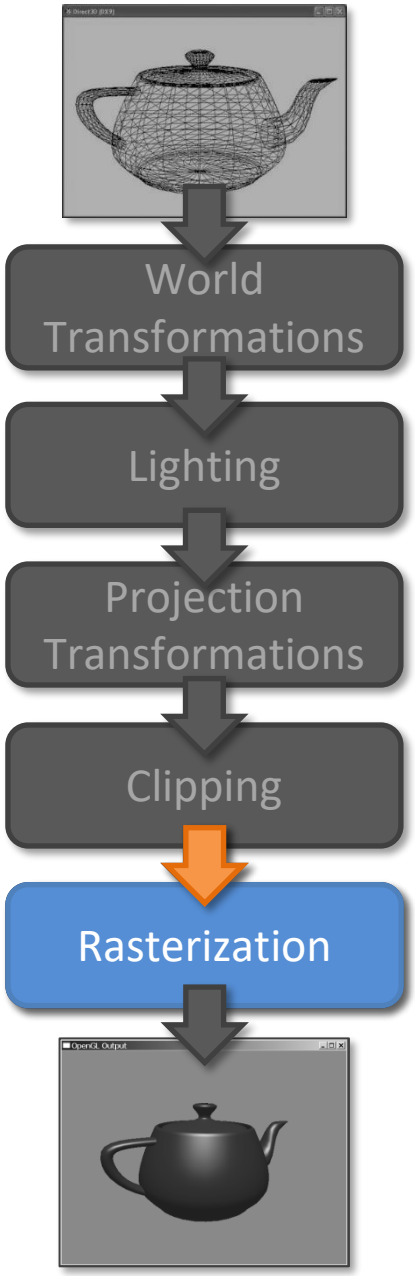


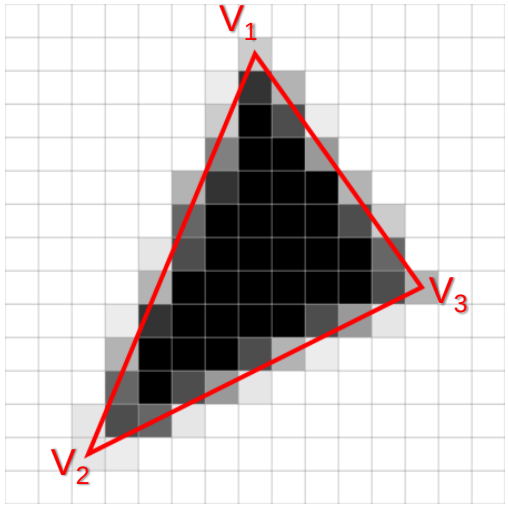

World Transformations
Lighting
Projection Transformations
Clipping
Rasterization

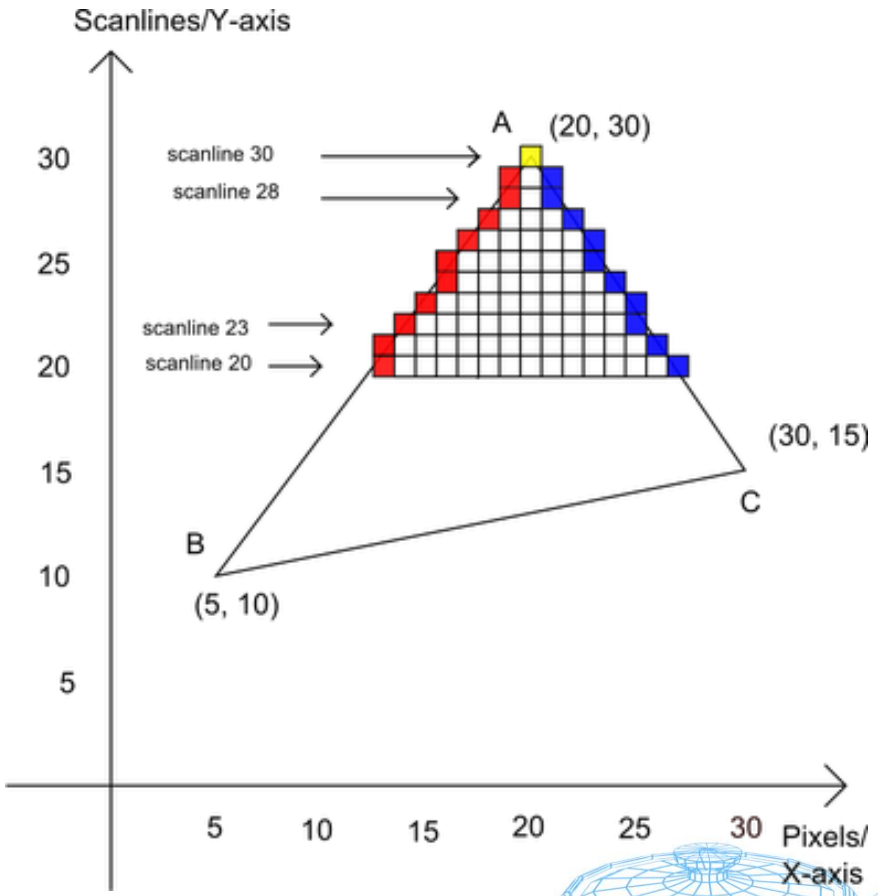

camera
camera
Exit module
3D
2D
Virtual Reality Laboratory
[+] Camera modifier
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE


World Transformations
Lighting
Projection Transformations
Clipping
Rasterization

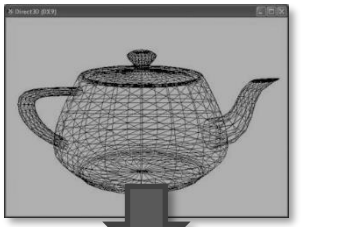

camera
camera
Exit module
Virtual Reality Laboratory
[+] Camera modifier
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE







[source: codeproject.com]



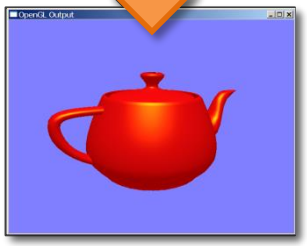
World Transformations

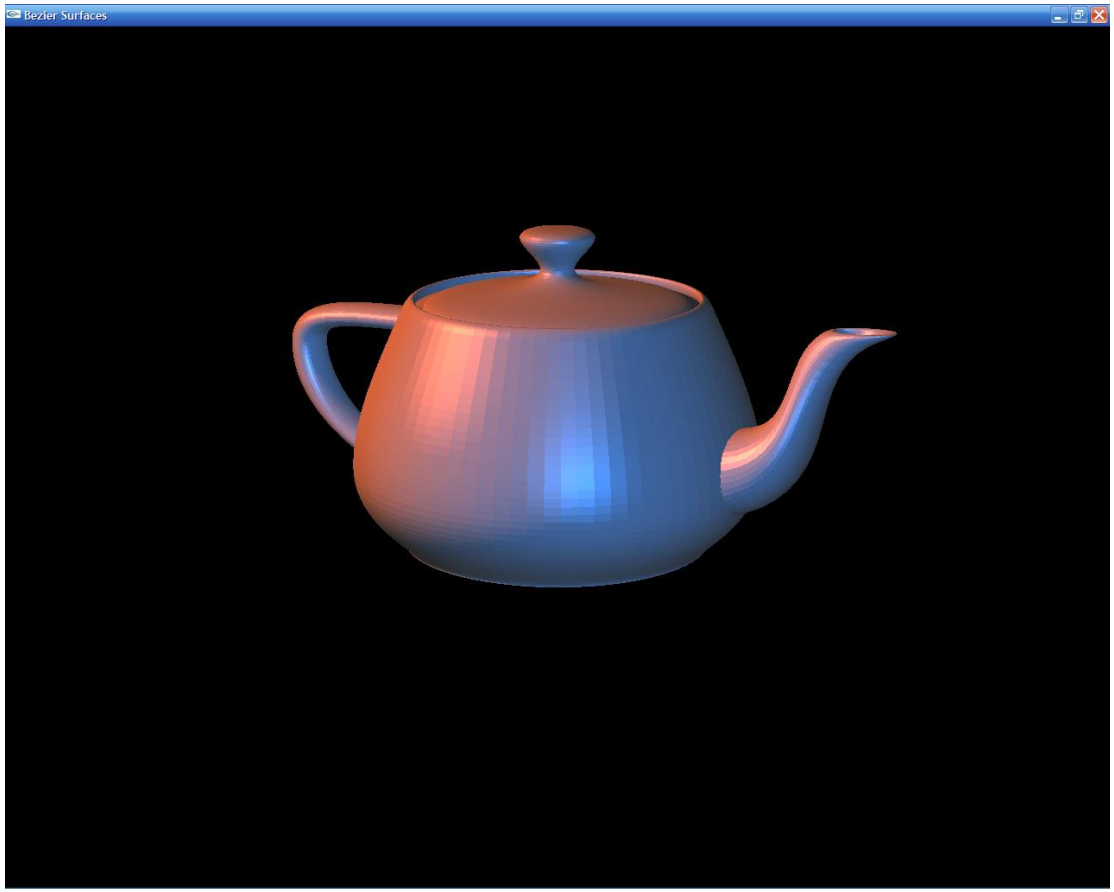
Lighting

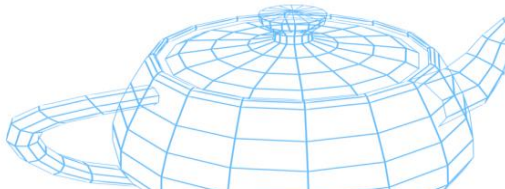
Projection Transformations

Clipping

Rasterization



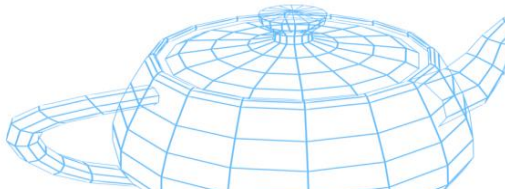




Rasterization

- Advantages:
  - Very fast:
    - Efficient way to map primitives into 2D pixels.
    - Local, coherent information.
    - Primitives can be streamed to reduce memory consumption.
    - Hardware-friendly:
      - GPUs were originally 2D-accelerators to improve performance in the final rasterization step only.

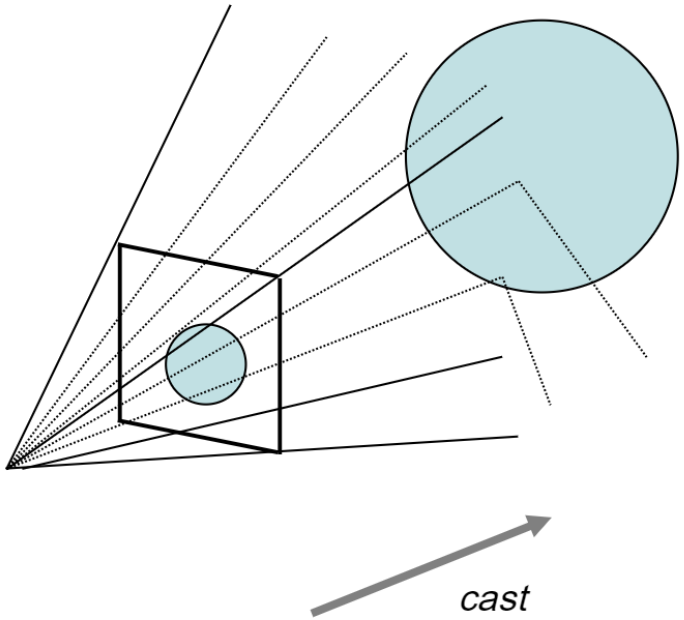
- Disadvantages:
  - Less intuitive than ray tracing.
  - Some effects are difficult to simulate (refractions, shadows).



“Rasterization is fast, but needs cleverness to support complex visual effects. Ray tracing supports complex visual effects, but needs cleverness to be fast.”

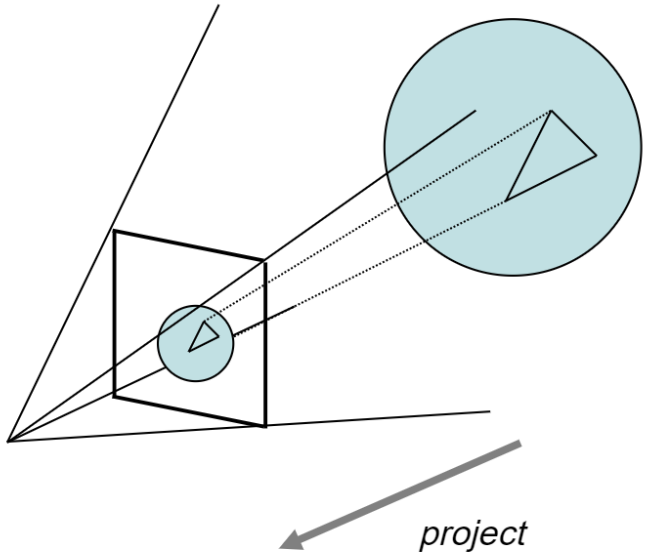
David Luebke, Nvidia

Ray tracing

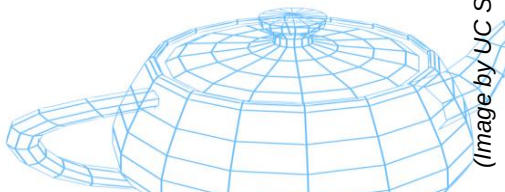


Works by casting rays into the scene through the image plane

Rasterization

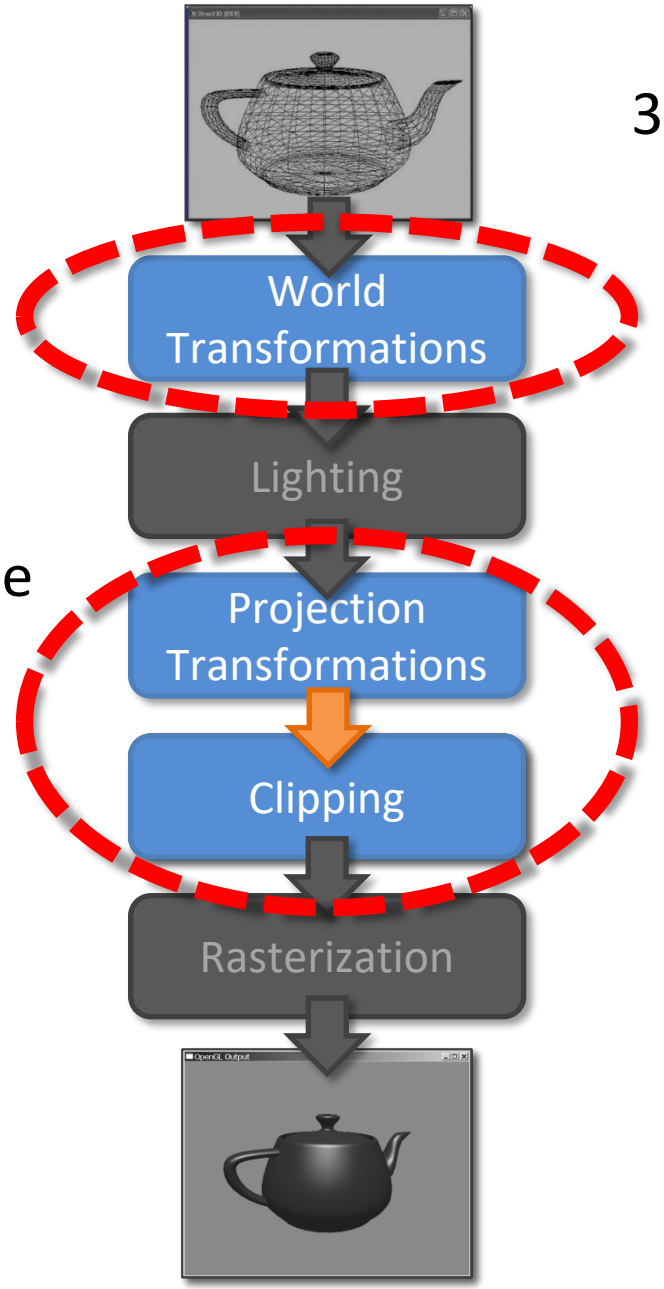


Works by projecting polygons on the image plane



(Image by UC Santa Barbara)

Rendering pipeline



3D primitives

2D images