南京大学
NANJING UNIVERSITY

图形绘制技术（Rendering）
Chapter 1: Introduction

过 洁
南京大学计算机科学与技术系
guojie@nju.edu.cn


欢迎进入三维图形世界

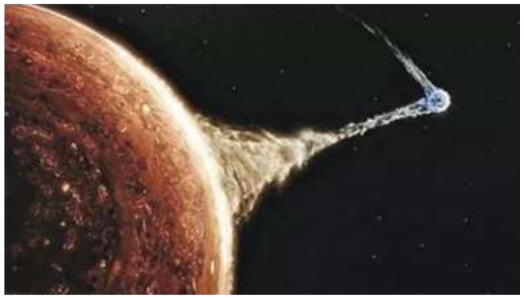
- 游戏引擎课程？
- 战略物资储备！


OpenAI


UNREAL
ENGINE

Rendering is everywhere












Rendering is everywhere

Just for entertainment?

Rendering is everywhere

Rendering is everywhere

Rendering is everywhere

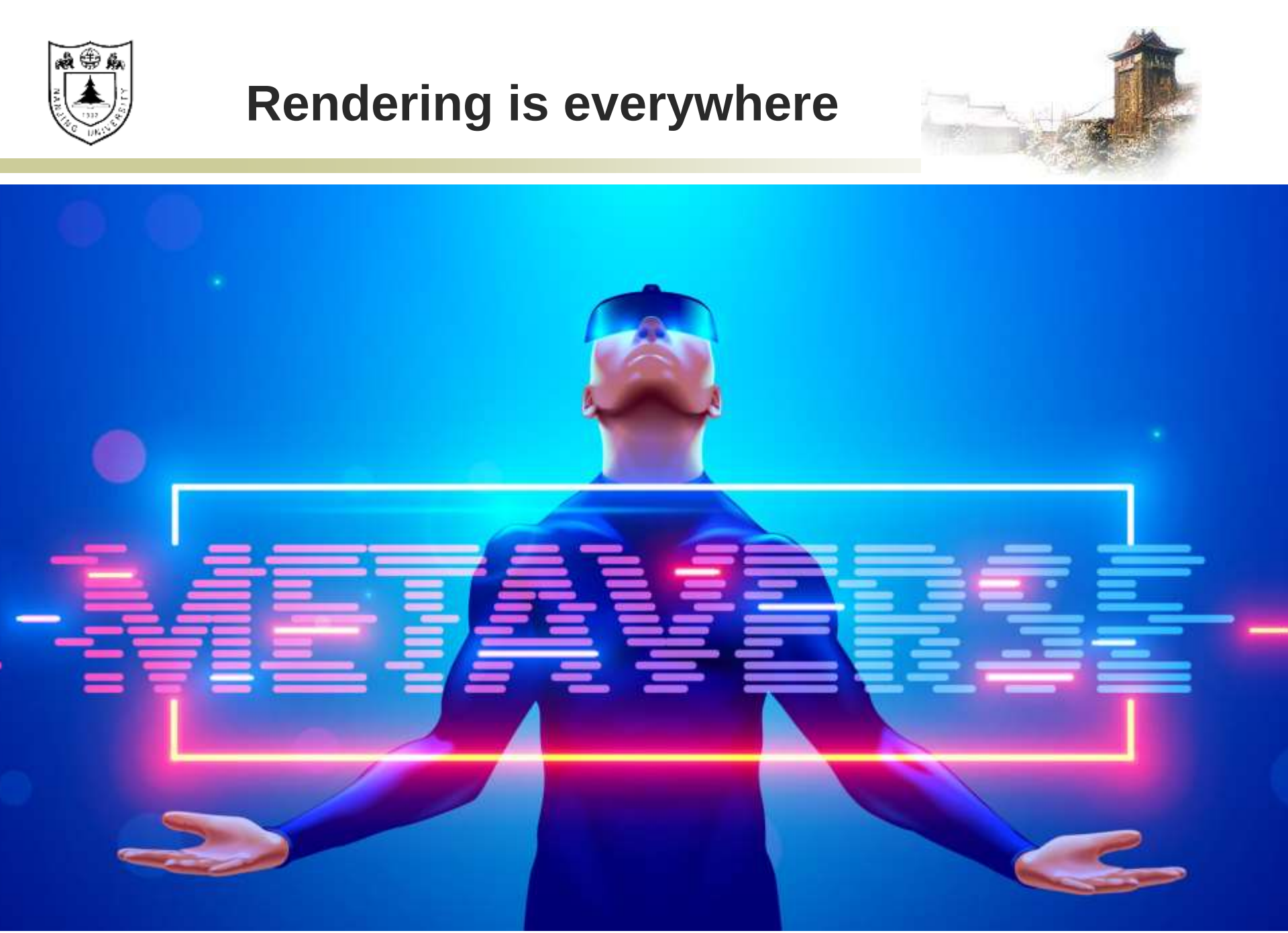
Industrial design

Rendering is everywhere

Product visualization

Rendering is everywhere

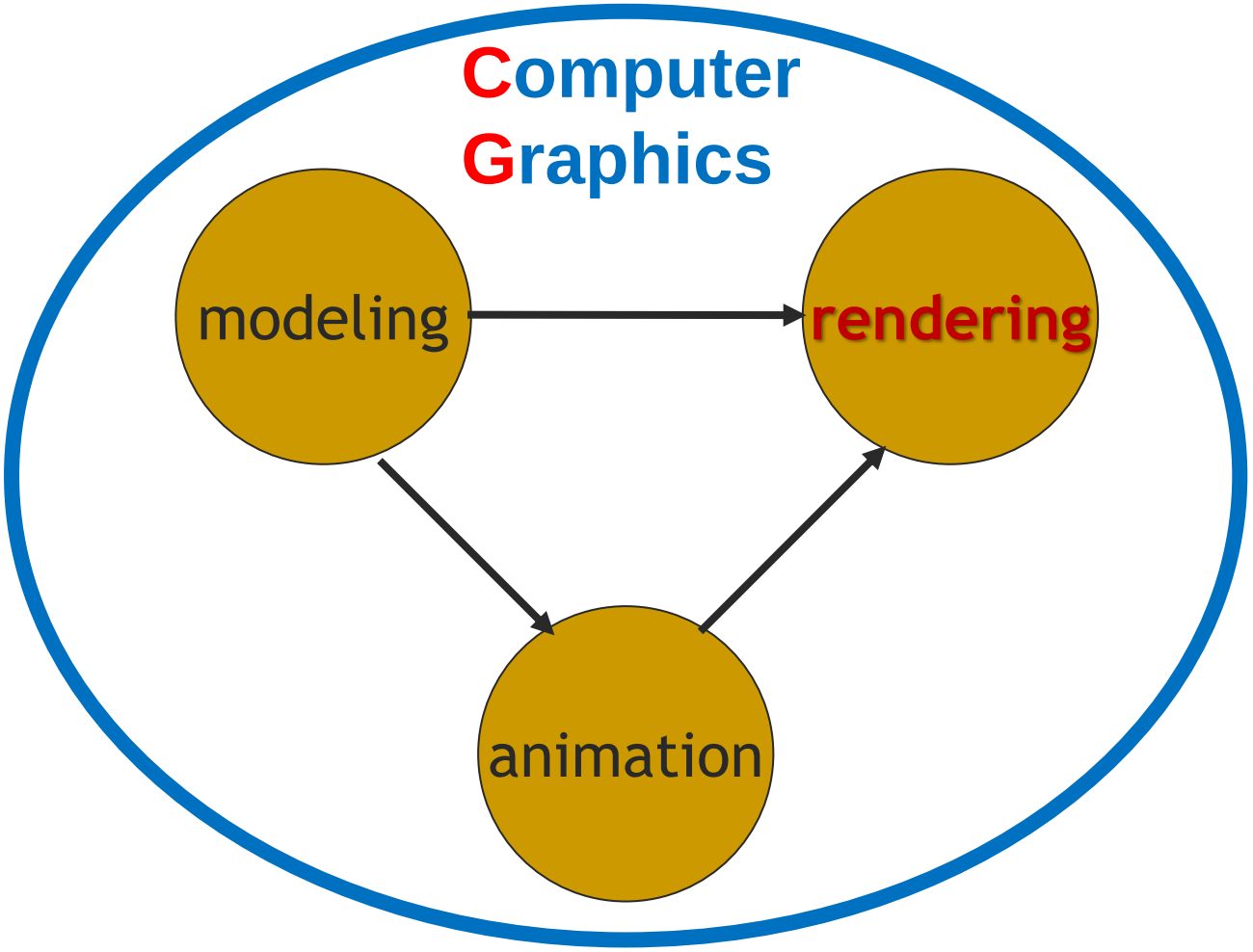
Interior decoration



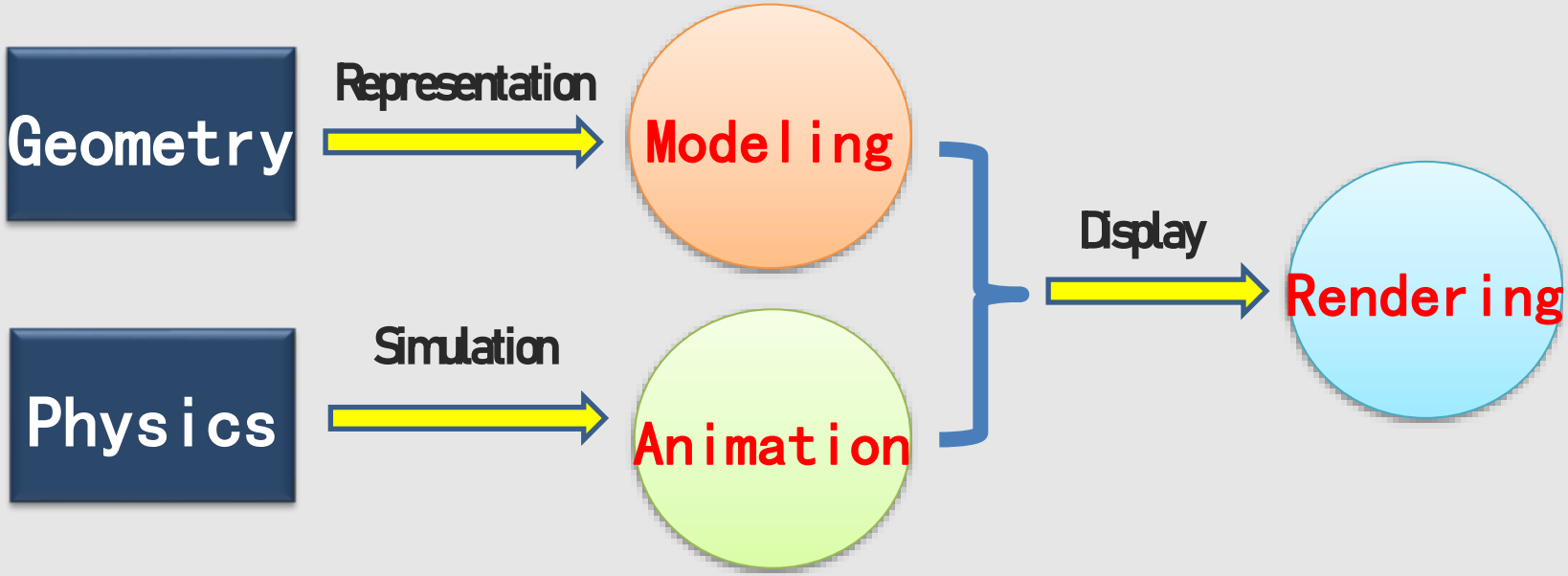
Rendering is everywhere

Rendering is everywhere

Computer Graphics



Computer Graphics

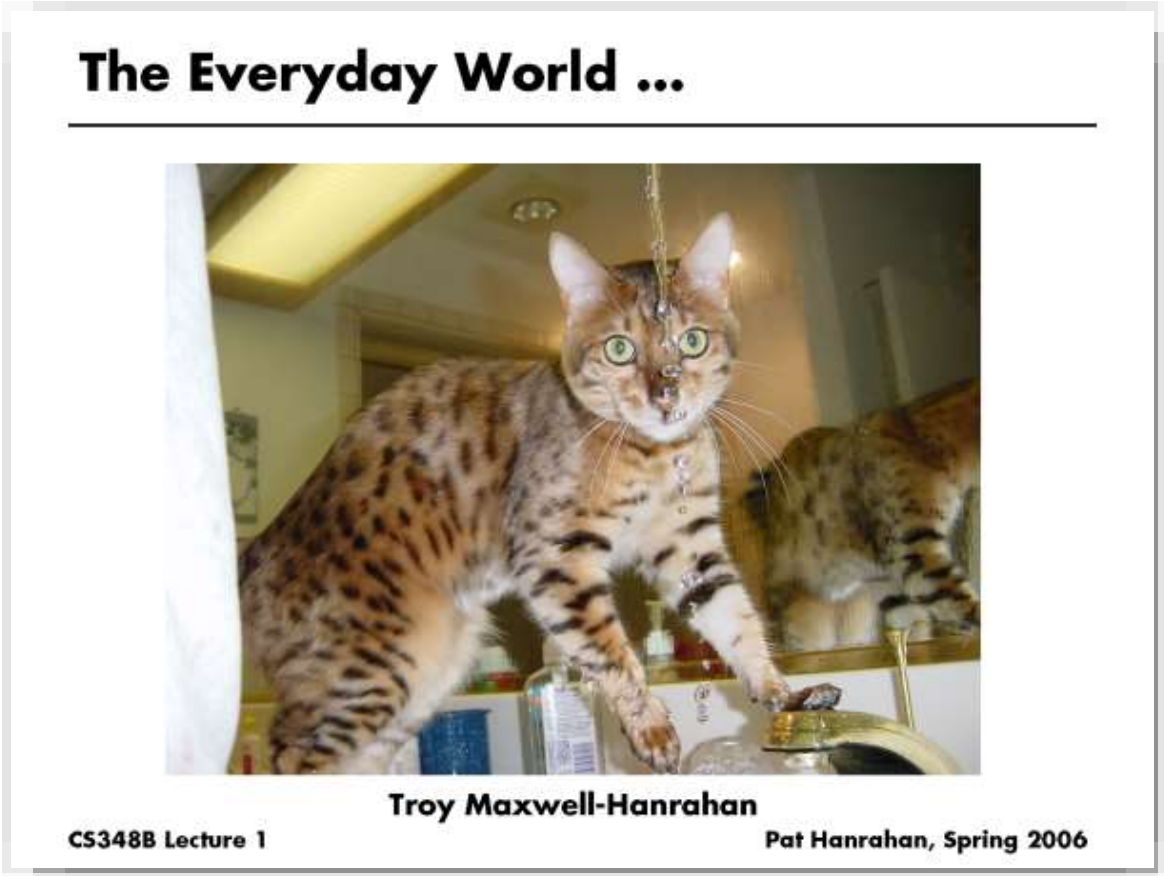


World simulation

Rendering

The goal of rendering

Pat Hanrahan



Rendering

The art of lighting

Course information

- **Schedule**
  - 18:30-20:20, Monday
- **Location**
  - 仙Ⅱ-406
- **Instructor**
  - **Jie Guo (过洁)**
- **Course QQ Group**
  - 715713980
- **Course Website**
  - 教学立方，邀请码：4NNVMSEQ

群名称:2024图形绘制技术

群　号:715713980

Course information

- **Prerequisites**：Calculus、Linear algebra、Probability、Computer Graphics
- **Goal**：Physically-based Realist Rendering
- **Grading**：Three Assignments（50%）

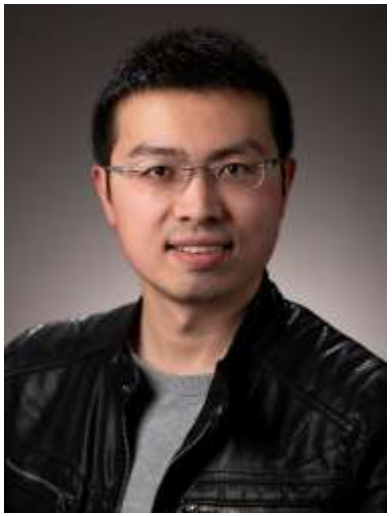
  +Final Project（50%）

- 期末考核安排
  - 本学期的最后一堂课进行作业现场答疑和检查
  - 最后一堂课之后顺延一周提交最终作业

Prerequisites: Computer Graphics

- Not necessary
- Better to known some basic concepts
- Recommended online course:

https://sites.cs.ucsb.edu/~lingqi/teaching/games101.html



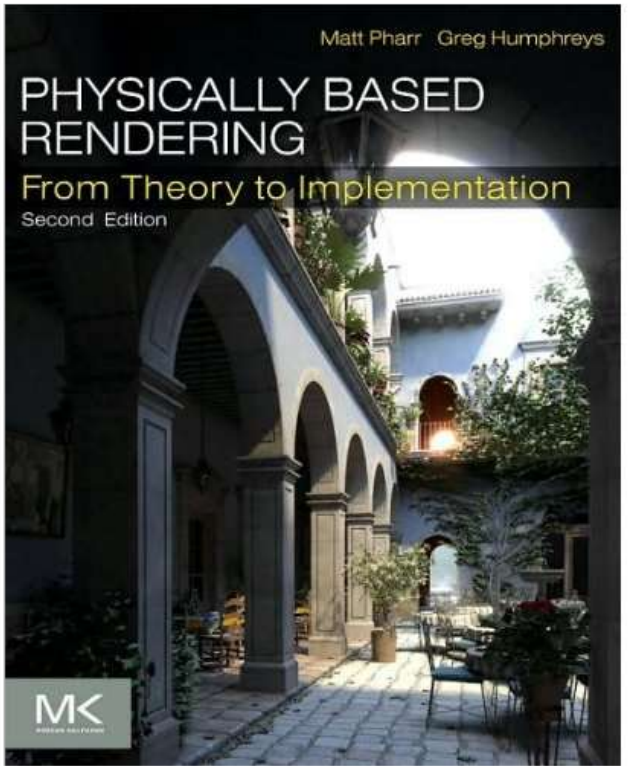
GAMES101: 现代计算机图形学入门

2020 年春季学期（在线直播）

主讲人：闫令琪
（UCSB助理教授，2019年SIGGRAPH杰出博士论文奖）

Textbook

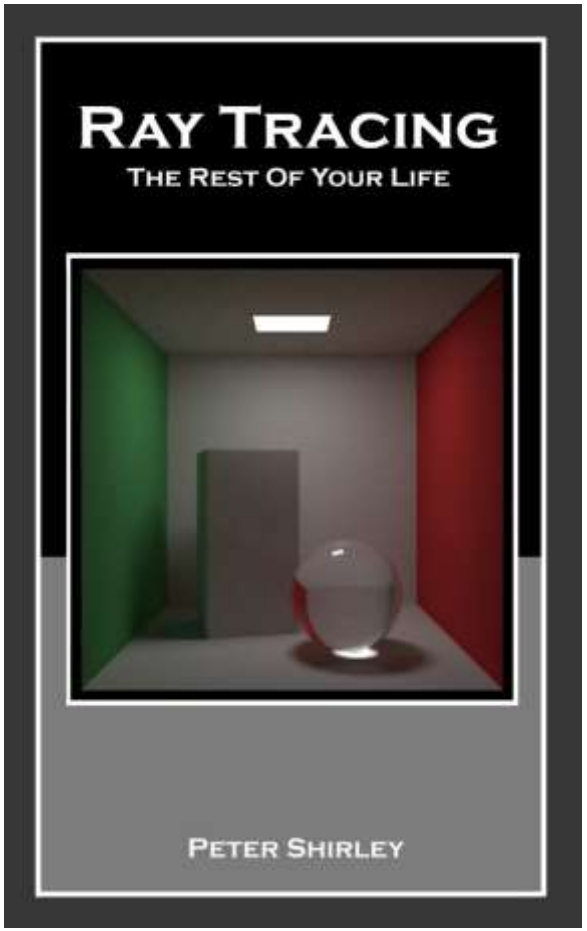
- **Physically Based Rendering: from Theory to Implementation**
  - **Second/Third edition**
  - **Authors**：Matt Pharr, Wenzel Jakob and Greg Humphreys
  - **Webpage**：http://www.pbrt.org/
  - **Source code**：
    https://github.com/mmp/pbrt-v2
    https://github.com/mmp/pbrt-v3
    https://github.com/mmp/pbrt-v4



References

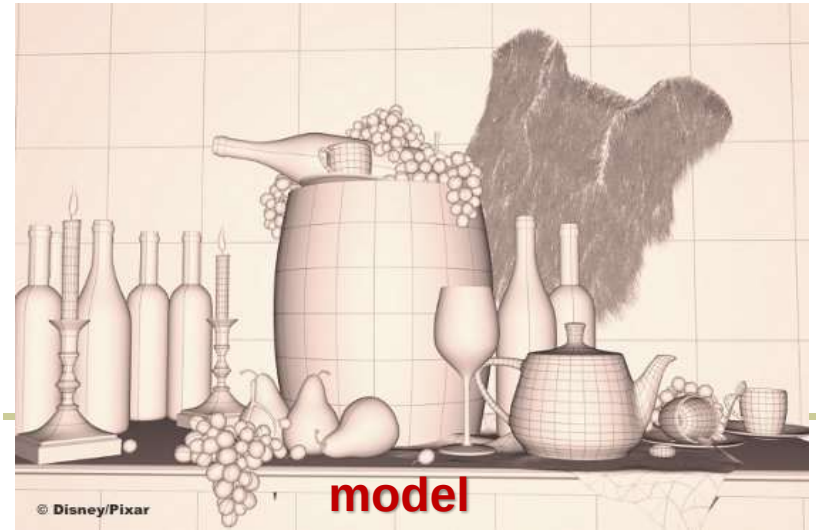

RAY TRACING
IN ONE WEEKEND
PETER SHIRLEY


RAY TRACING
THE NEXT WEEK
PETER SHIRLEY

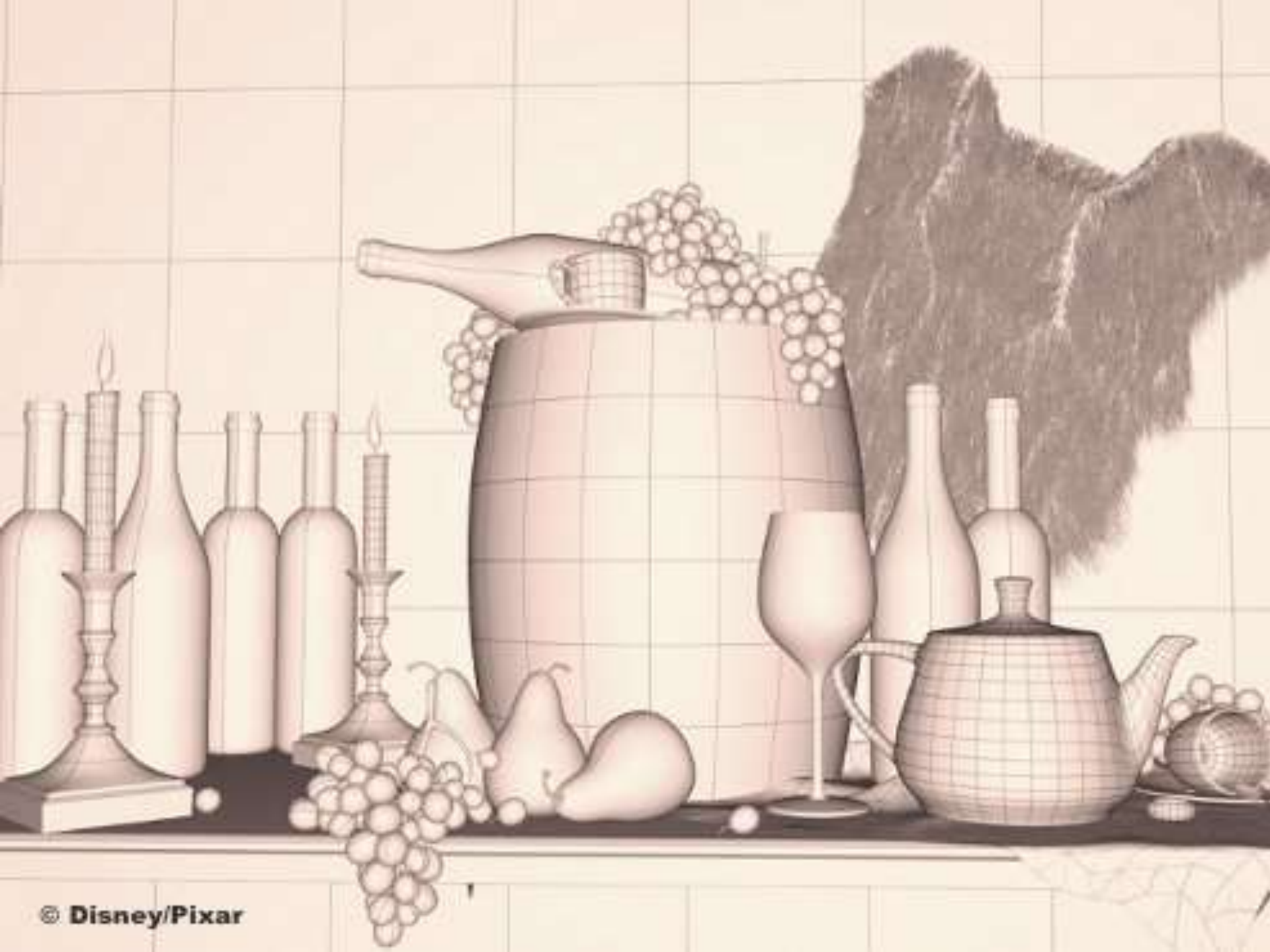

RAY TRACING
THE REST OF YOUR LIFE
PETER SHIRLEY

What is rendering

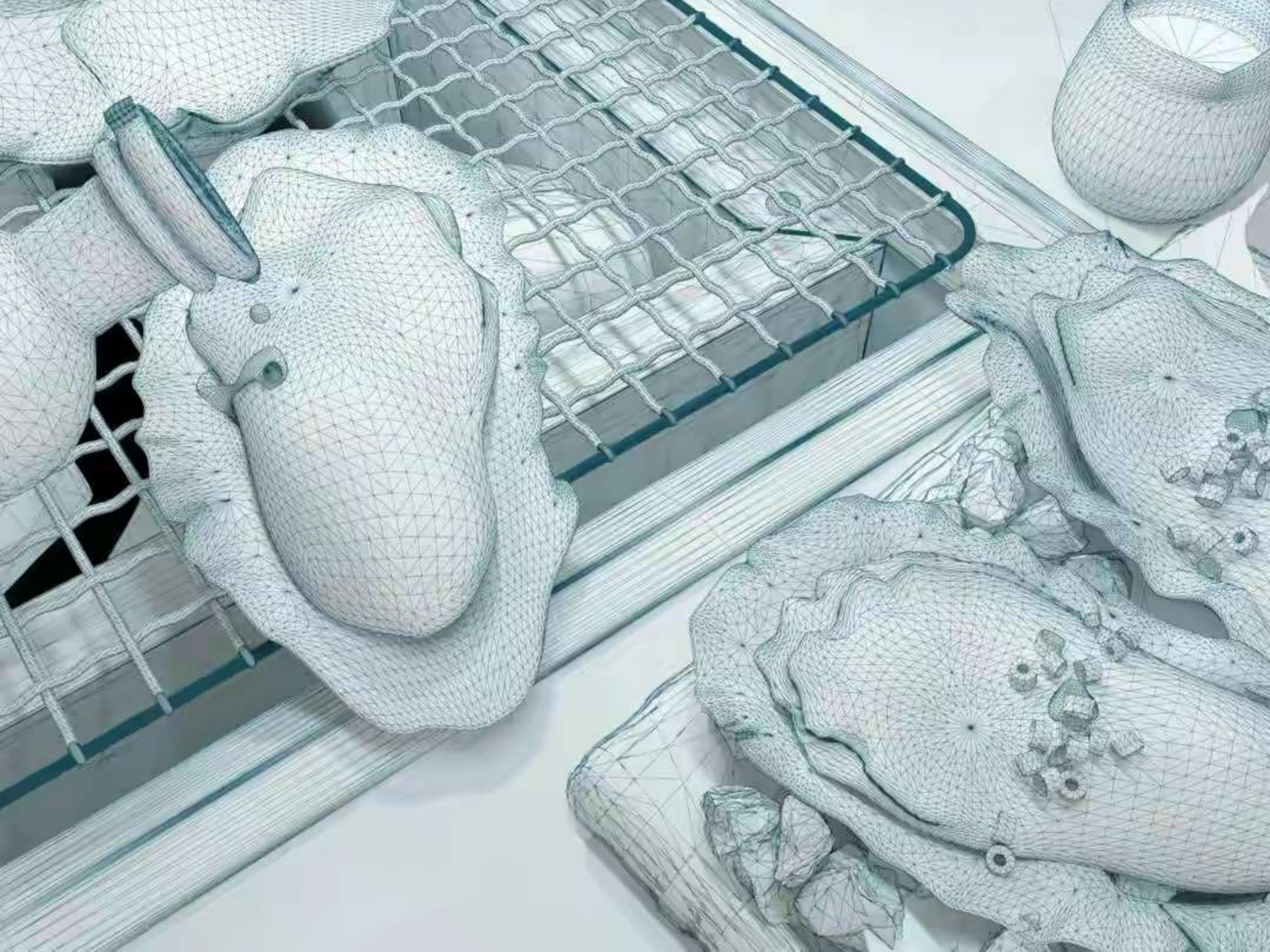
- The process of generating an **image** from a **model**, by means of a **computer program**.



computer program




© Disney/Pixar



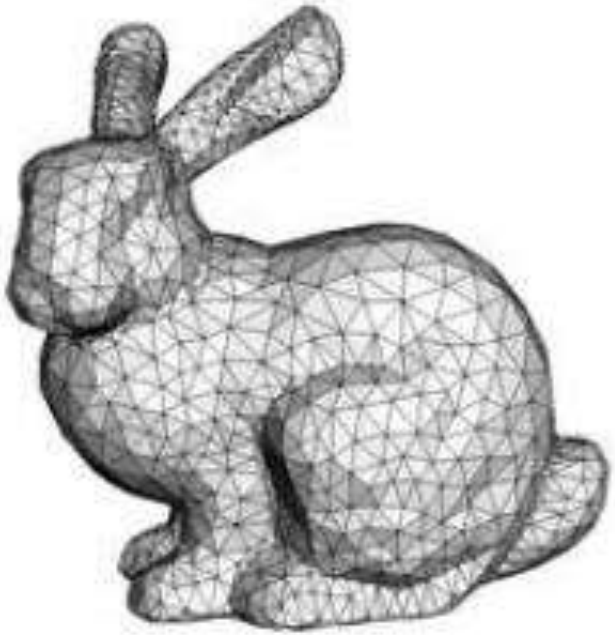
What is rendering

- The process of generating an **image** from a **model**, by means of a **computer program**.

- **image**+ **model**–**comp**=painting, photography
- **model**+ **comp**–**image**=3D printing, hydrographic printing, sound rendering
- **image**+ **comp**–**model**=abstract graphics, image to image translation

3D printing

- 3D printing is the process of making a real physical 3D object from digital file using some material, in a manner similar to printing images on paper.



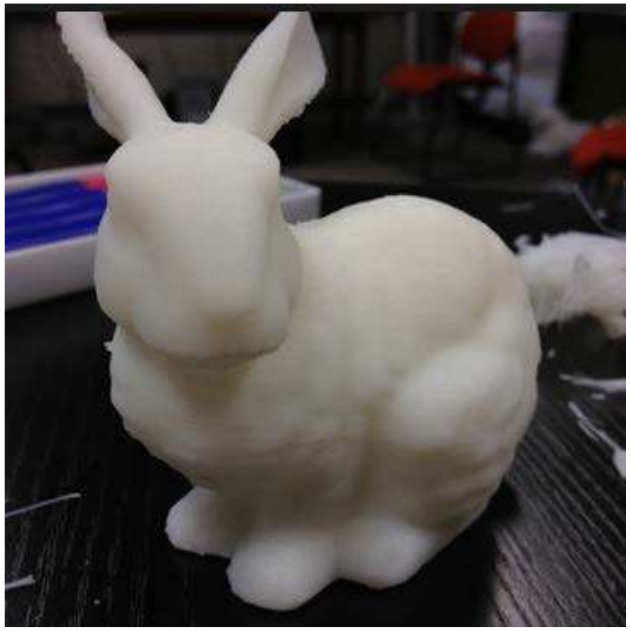
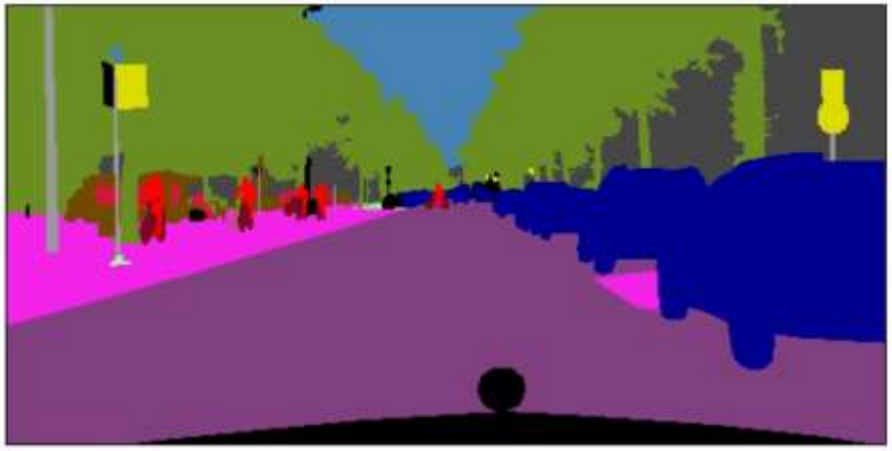


Image to image translation



Text to image translation

TEXT DESCRIPTION

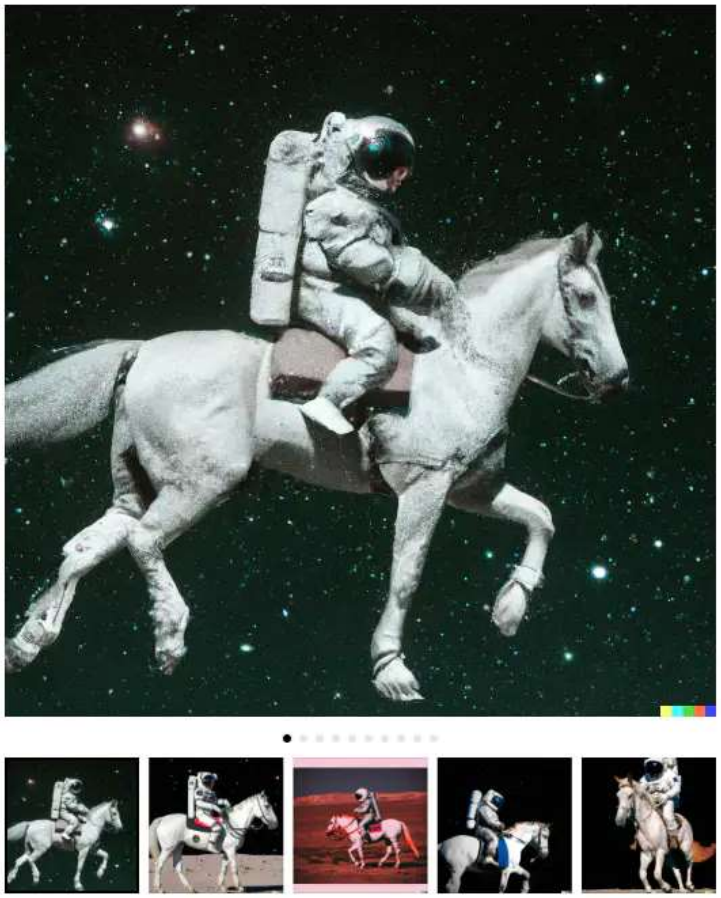
An astronaut Teddy bears A bowl of soup

riding a horse lounging in a tropical resort in space playing basketball with cats in space

in a photorealistic style in the style of Andy Warhol as a pencil drawing

→

DALL·E 2



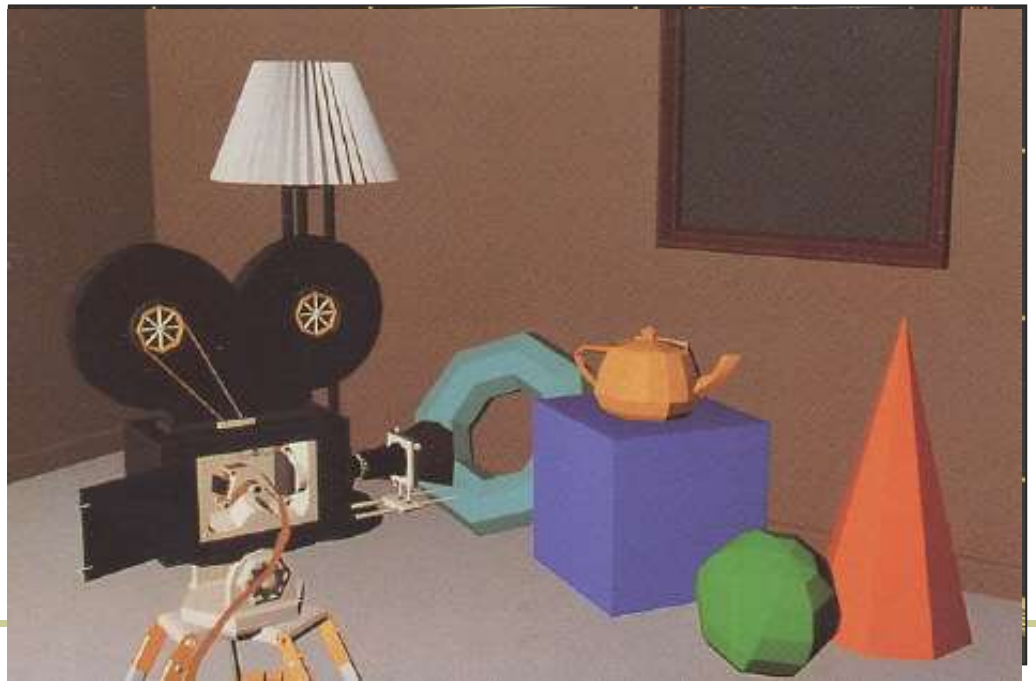
Text to video (Sora)

What is rendering?

- Core area in computer graphics
- Efficiently and easily create visual appearance
- Long history (1960s to current time)
- From basic visibility and shading, to global illumination, to image-based rendering, to data-driven appearance and neural rendering
- Many links to physics, mathematics, psychology, computer science
- We focus on **physically-based rendering** (offline rendering)

Brief history: 1960s (Visibility)

- Roberts (1963), Appel (1967) - hidden-line algorithms
- Warnock (1969), Watkins (1970) - hidden-surface
- Sutherland (1974) - visibility = sorting

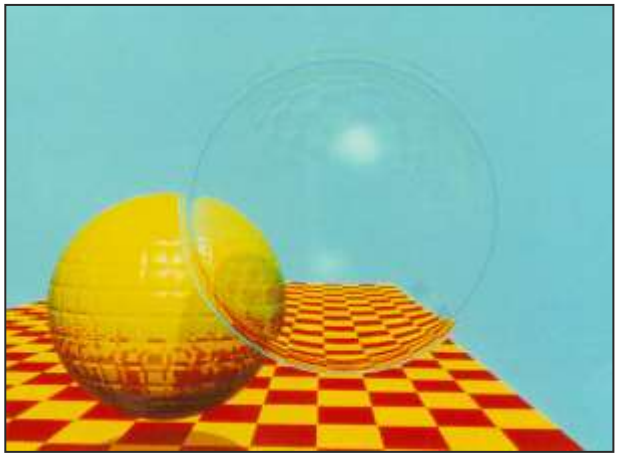


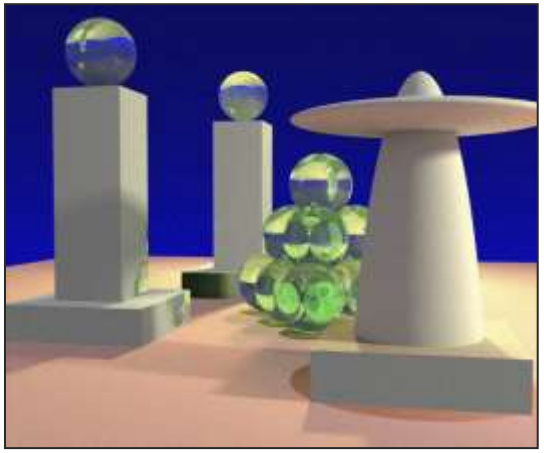
Brief history: 1970s (Lighting)

- 1970s - raster graphics
  - Gouraud (1971) - diffuse lighting, Phong (1974) - specular lighting
  - Blinn (1974) - curved surfaces, texture
  - Catmull (1974) - Z-buffer hidden-surface algorithm

Brief history: 1980s, 90s (Global Illumination)

- early 1980s - global illumination
  - Whitted (1980) - ray tracing
  - Goral, Torrance et al. (1984) radiosity
  - Kajiya (1986) - **the rendering equation**





Brief history: 2010s (Real-time ray tracing)

OptiX

NVIDIA GPU Ray Casting API

High-level GPU accelerated ray-casting API

C-API to setup scene and data

Multiple program domains and per ray payload under developer's control

Flexible single ray programming model

Supports multi-GPU and NVLINK

Develop "to the algorithm"

https://developer.nvidia.com/optix

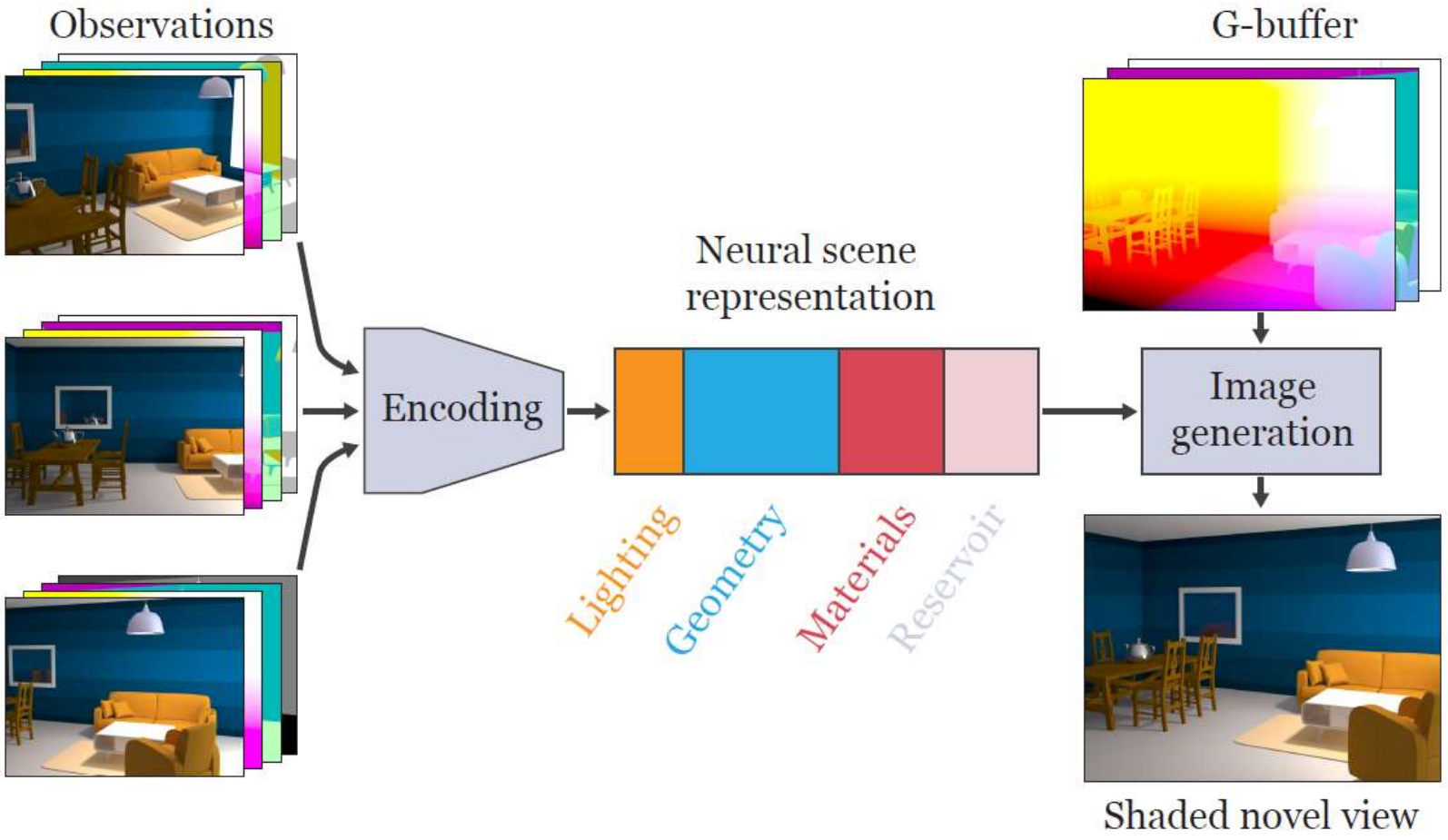
volume scattering and dispersion

hair intersection and shading

Brief history: 2010s (Real-time ray tracing)

Project Sol: A Real-Time Ray-Tracing Cinematic Scene Powered by NVIDIA RTX

Brief history: 2020s (Neural Rendering)



Brief history: 2020s (Neural Rendering)

Compositional Neural Scene Representations for Shading Inference

Jonathan Granskog[1,3] Fabrice Rousselle[1] Marios Papas[2] Jan Novák[1]

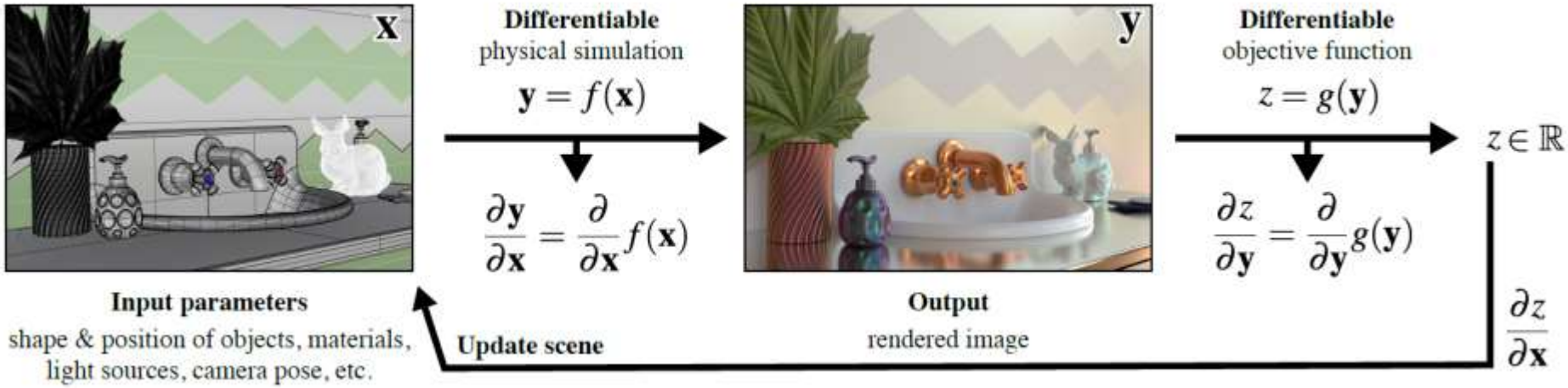
[1]NVIDIA [2]DisneyResearch|Studios [3]ETH Zurich

Brief history: 2020s (Neural Rendering)

Luma AI

Brief history: 2020s (Differentiable Rendering)

- Rendering as an integration problem
- Computing gradients by differentiating the integrals
- Inverse rendering!



2019 A.M. Turing Award

Hanrahan and Catmull's Innovations Paved the Way for Today's 3-D Animated Films

Toy Story

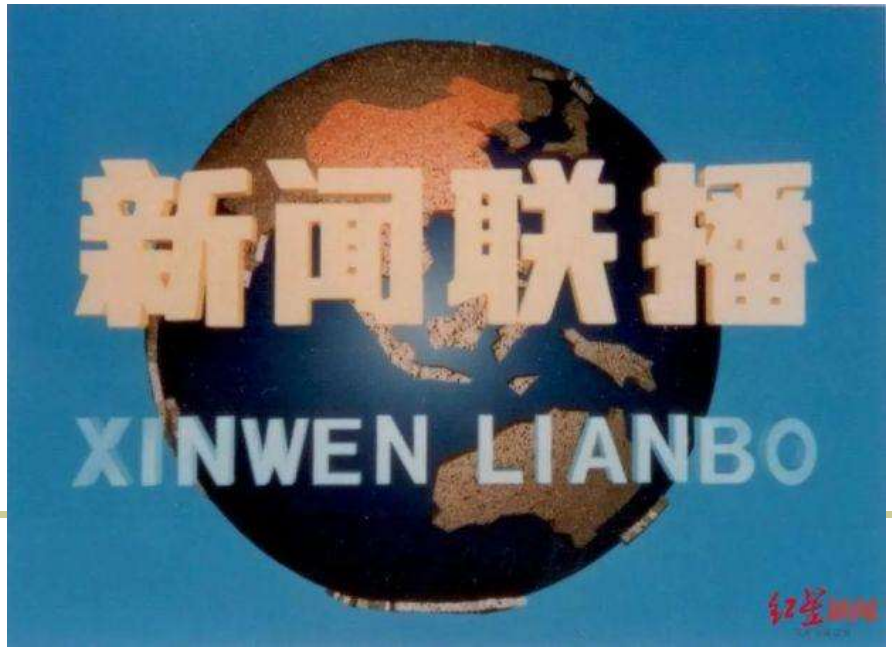
The first computer-animated feature film

2019 A.M. Turing Award

In memory of Dongxu Qi

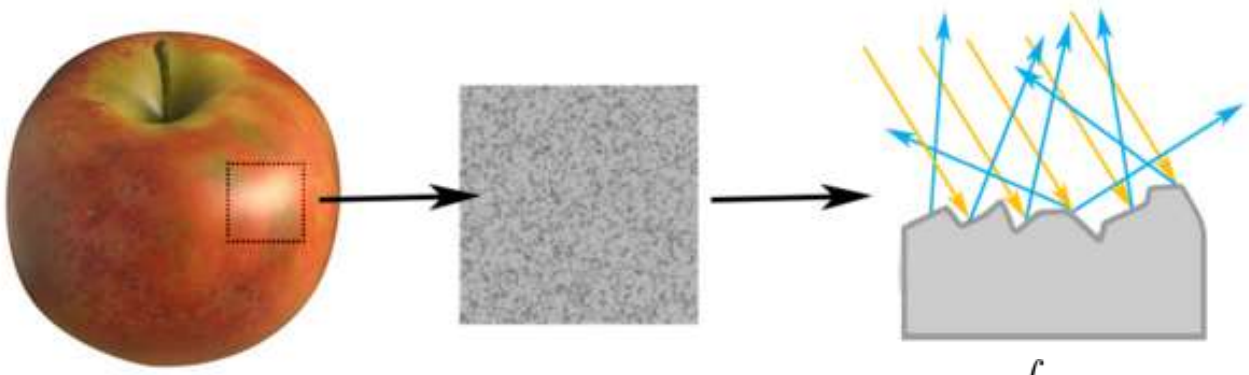
齐东旭教授

中国CAD与计算机图形学领域巨匠



The goal of this course

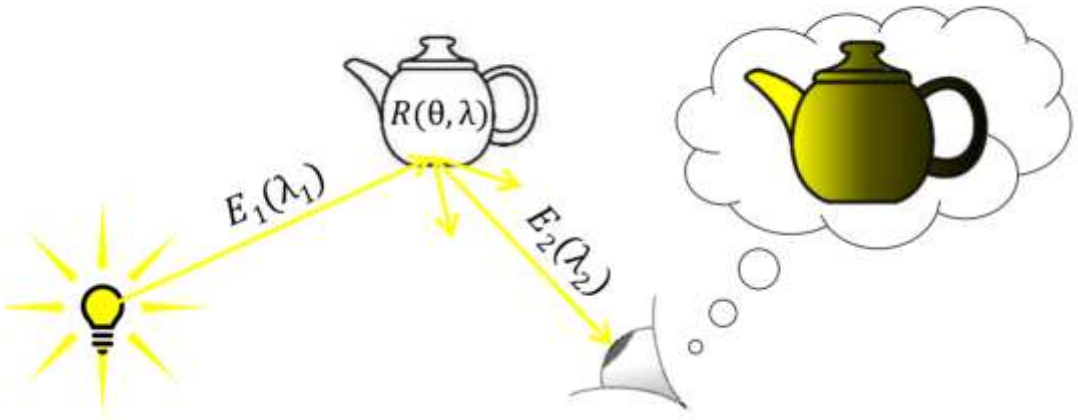
- Physically based rendering (PBR)
  - uses physics to simulate the interaction between matter and light, realism is the primary goal.
  - attempts to model photographic rendering mathematically and computationally.



$$L_o(x,\vec{\omega}) = \underbrace{L_e(x,\vec{\omega})}_{emitted} + \underbrace{\int_\Omega L_i(x,\vec{\omega}')\, f_r(\vec{\omega},x,\vec{\omega}')\, \cos\theta\, d\vec{\omega}'}_{reflected\ incoming\ light}$$

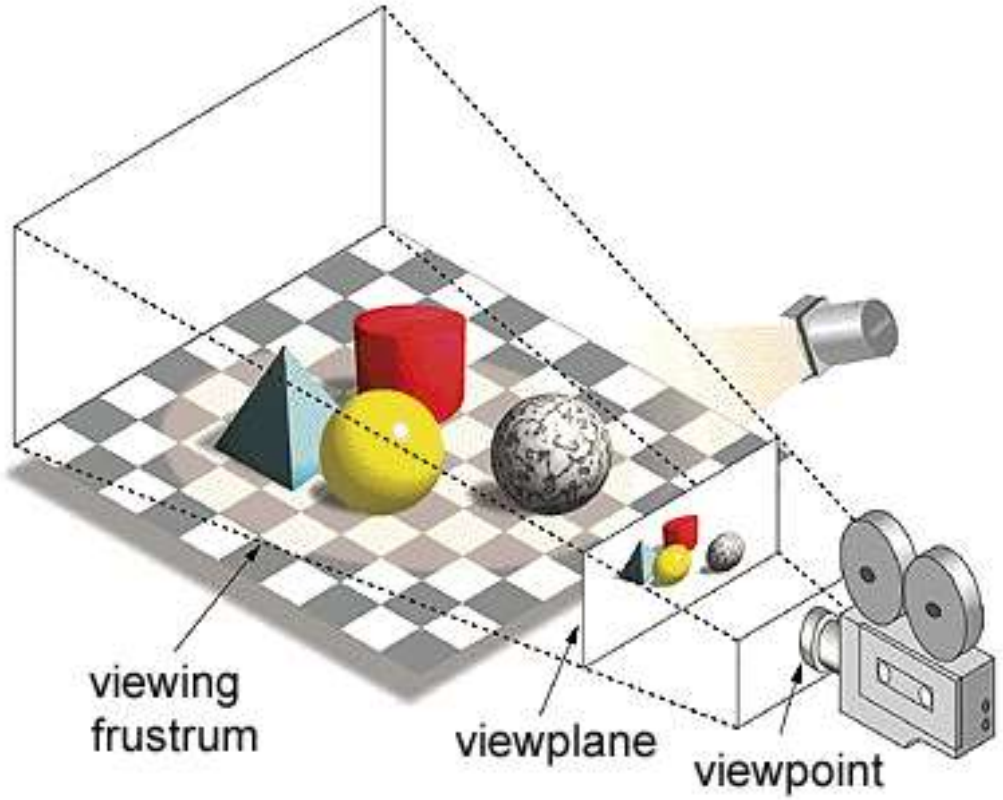
How light works in real life

- Light starts from light sources and carries energy defined by spectrum in straight line,
- Path modified according to optical properties of materials it encounters,
- Until it reaches the human eye to be interpreted as color and intensity

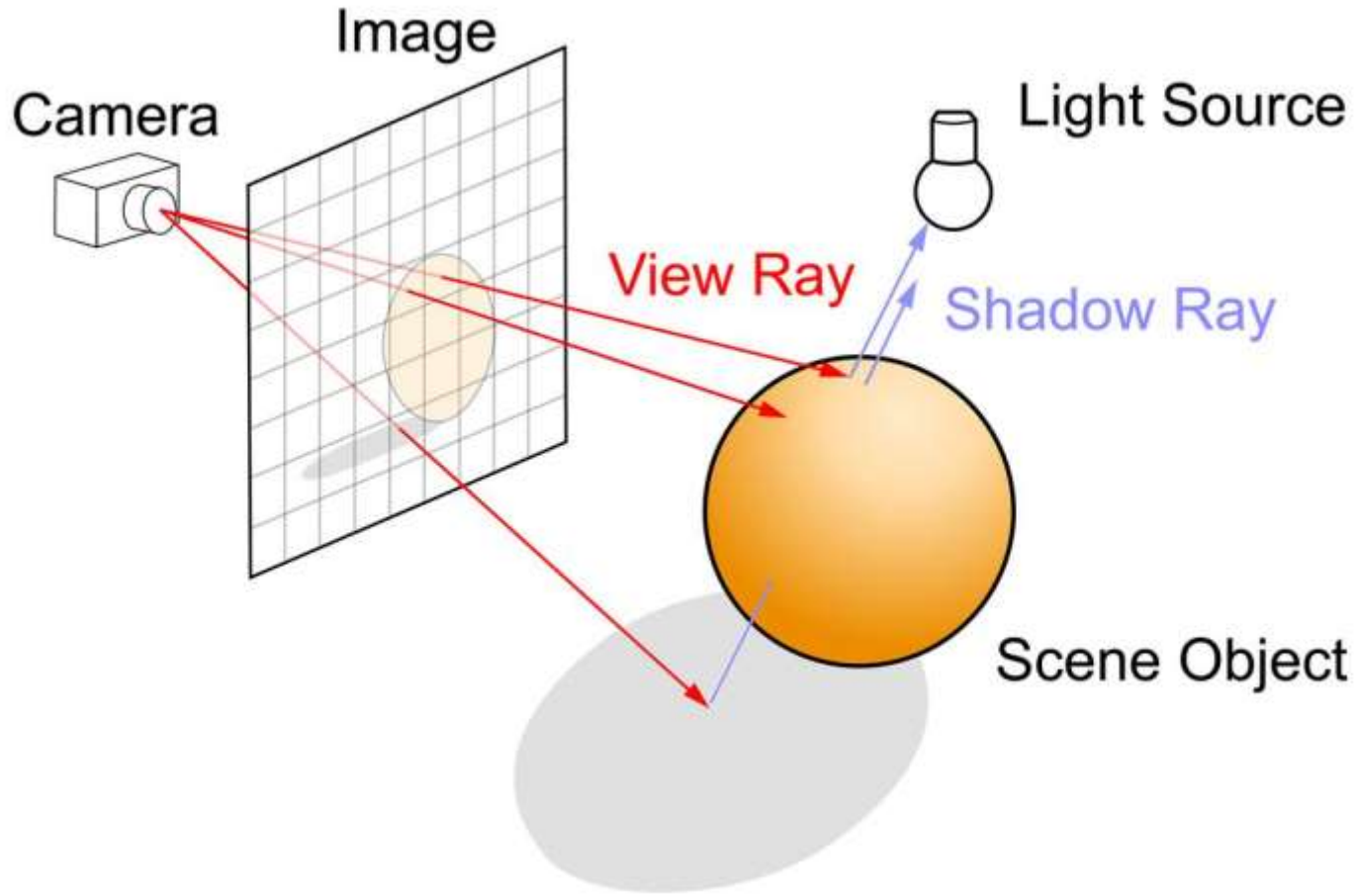


How we simulate?

From Computer Desktop Encyclopedia
Reprinted with permission.
© 1998 Intergraph Computer Systems



“Taking a Picture”



Realism

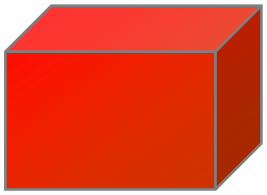
- Shadows
- Reflections (Mirrors)
- Transparency
- Translucency
- Interreflections
- Textures
- Complex Illumination
- Realistic Materials
- And many more

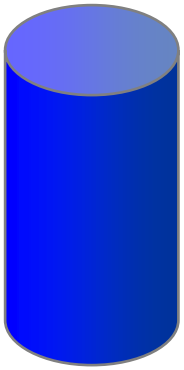
Models needed for PBR

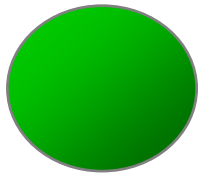
- Consider the experiment: taking a picture".
- What do we need to model it?
  - Scene geometry
  - Camera
  - Light sources
  - Materials
  - Participating media
  - Light propagation
- Mathematical models for these physical phenomena are required as a minimum in order to render an image.

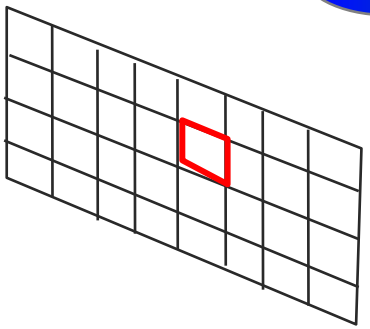
Ray Casting (Appel, 1968)

- a simple version of ray tracing

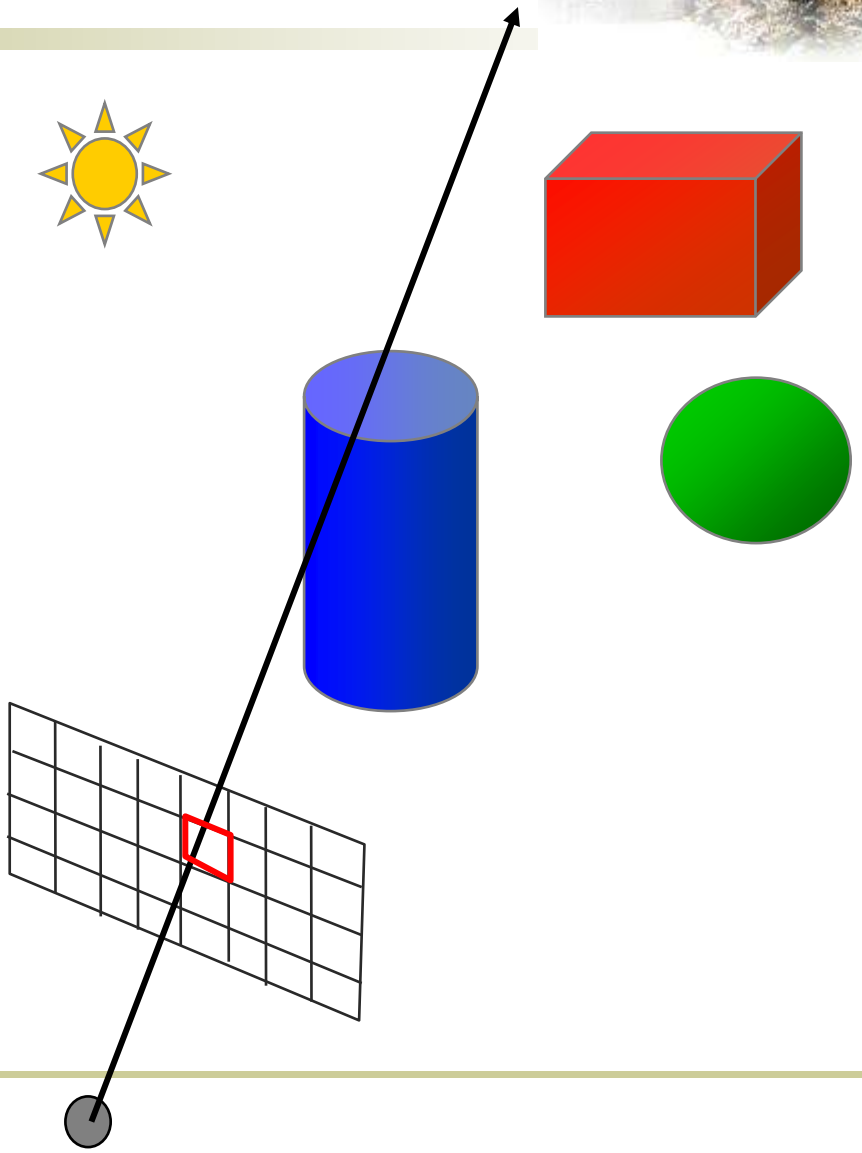




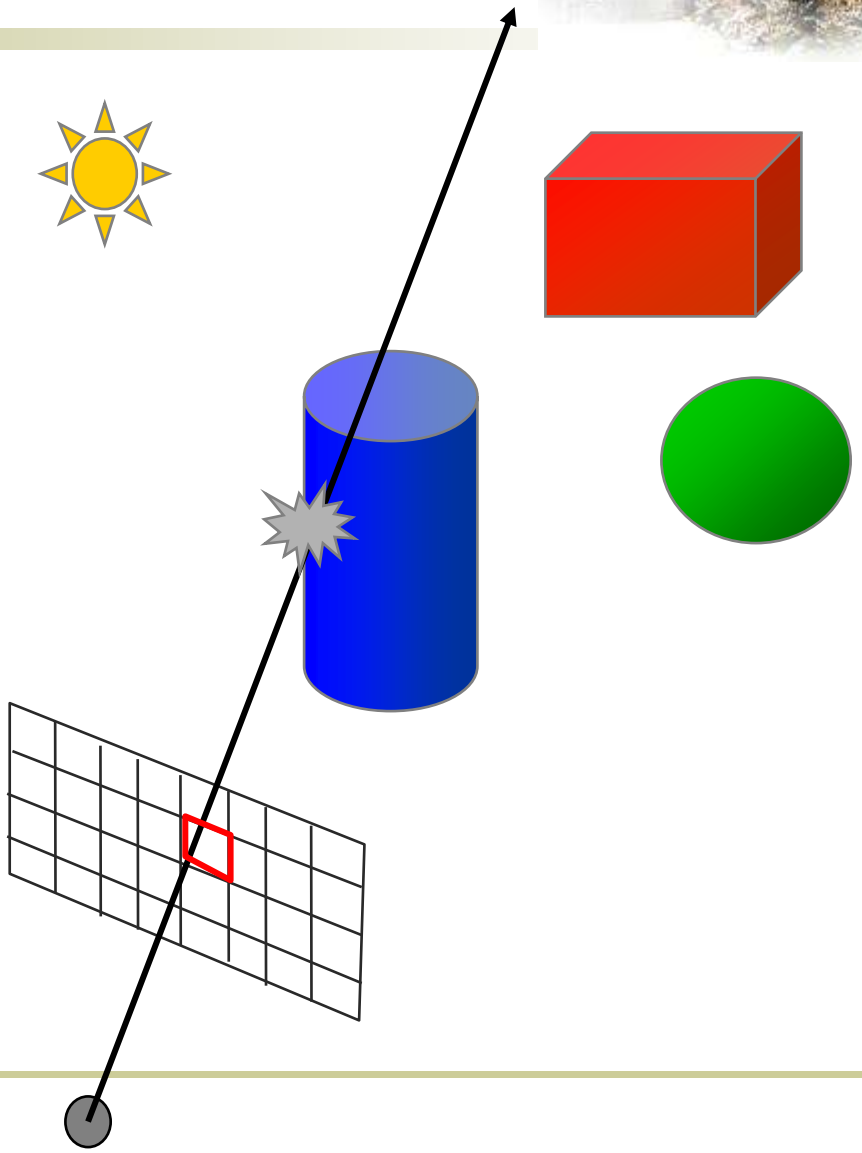




Ray Casting (Appel, 1968)

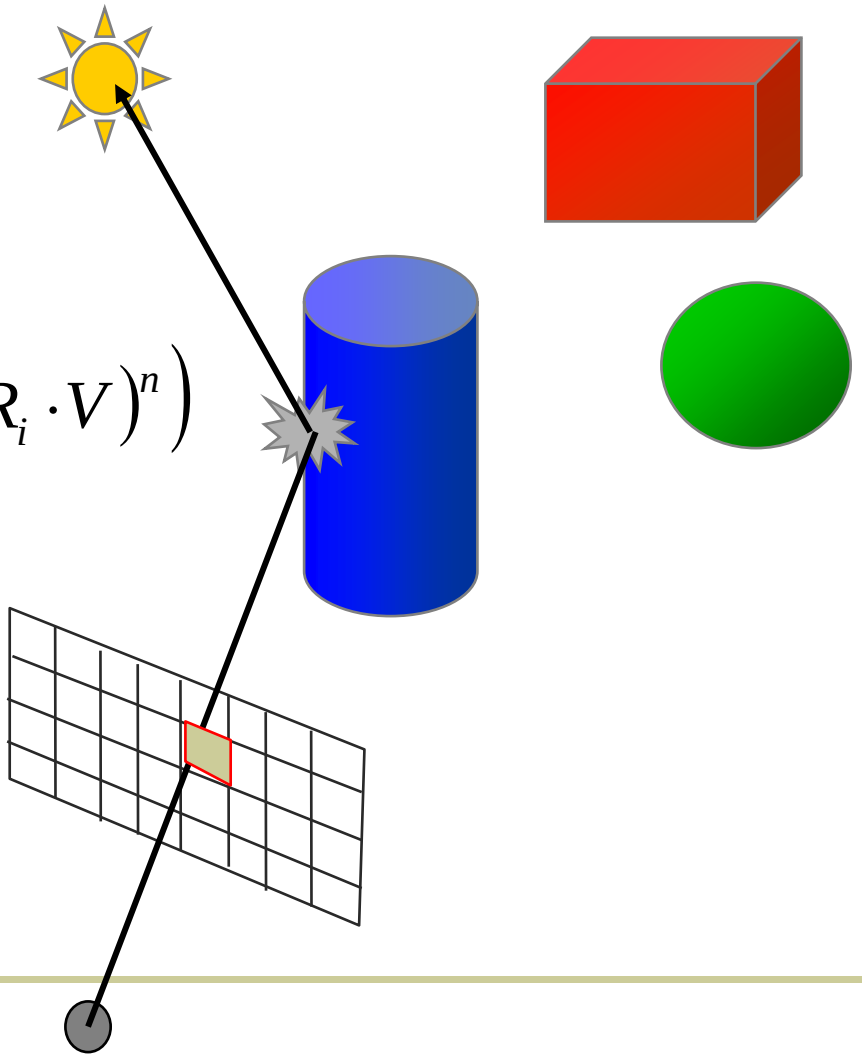


Ray Casting (Appel, 1968)

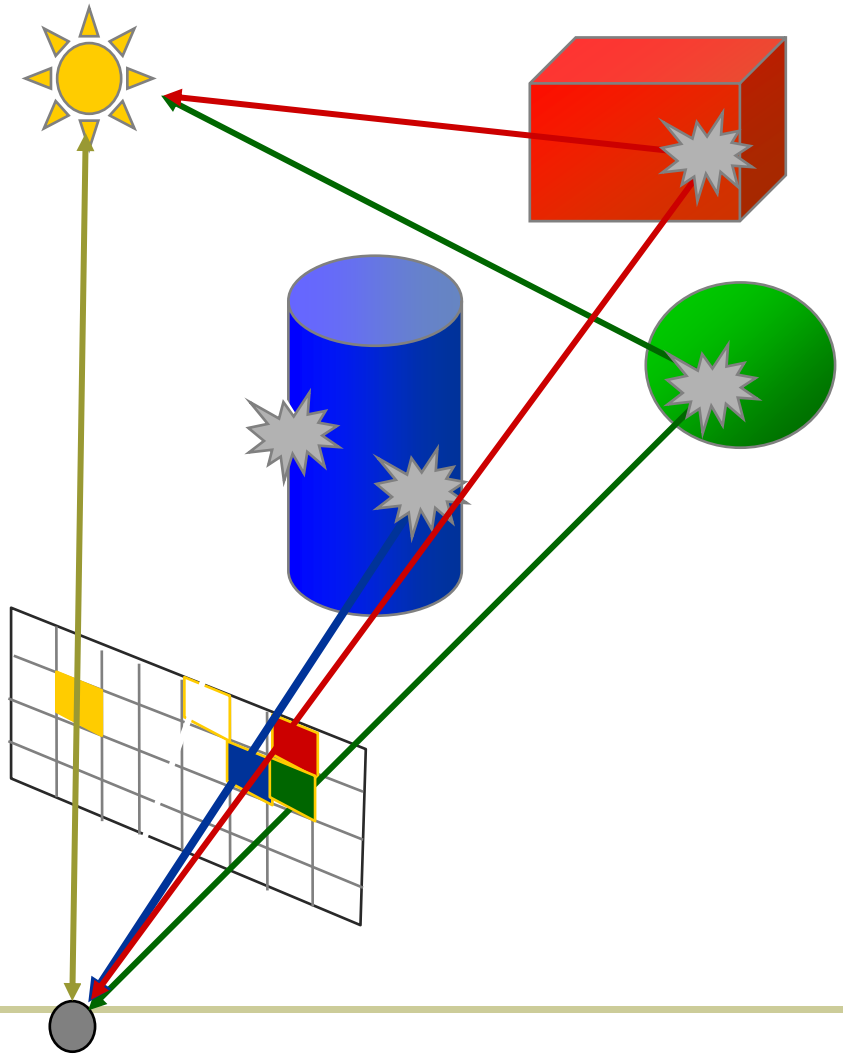


Ray Casting (Appel, 1968)

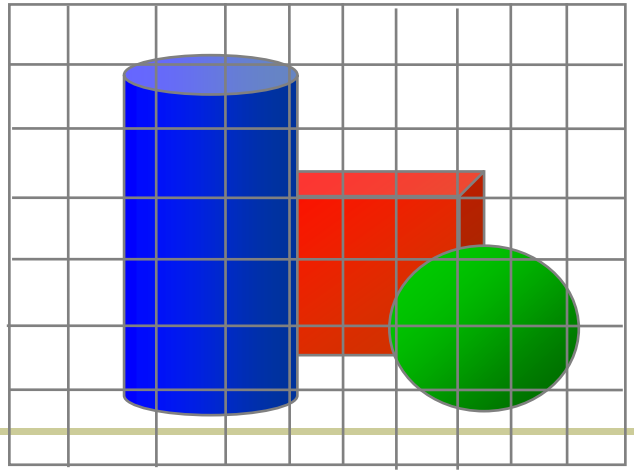
$$k_a I_a + \sum_{i=1}^{nls} I_i \left(k_d \left(L_i \cdot N \right) + k_s \left(R_i \cdot V \right)^n \right)$$



Ray Casting (Appel, 1968)



direct illumination



Ray Casting (Appel, 1968)

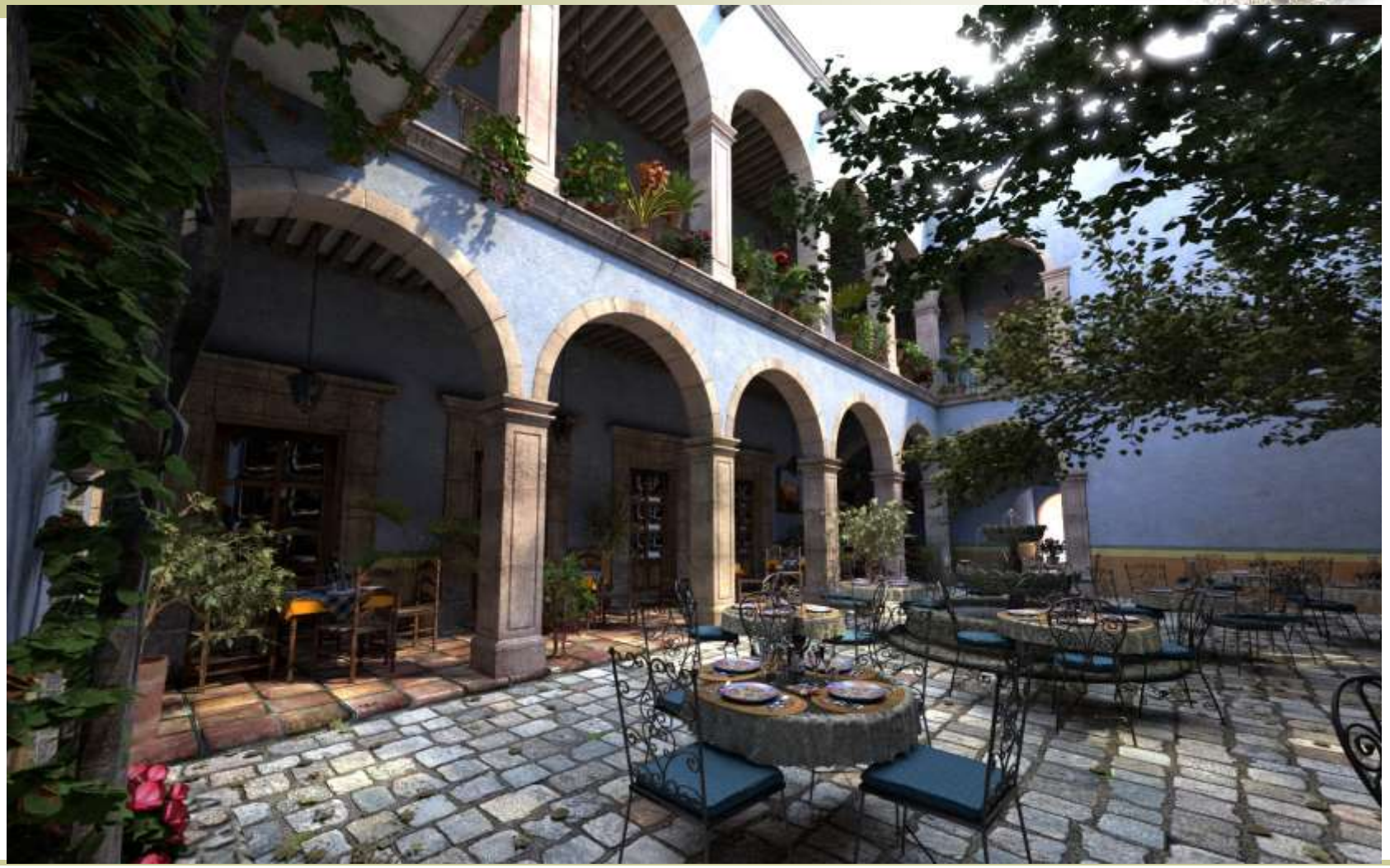
```
Image Raycast (Camera cam, Scene scene, int width, int
height){
        Image image = new Image (width, height) ;
        for (int i = 0 ; i < height ; i++)
        for (int j = 0 ; j < width ; j++)
        {
                Ray ray = RayThruPixel (cam, i, j) ;
                Intersection hit = Intersect (ray, scene) ;
                image[i][j] = FindColor (hit) ;
        }
        return image ;
}
```

Ray Casting (Appel, 1968)

```
Image Raycast (Camera cam, Scene scene, int width, int height){
        Image image = new Image (width, height) ;
        for (int i = 0 ; i < height ; i++)
        for (int j = 0 ; j < width ; j++)
        {
                Ray ray = RayThruPixel (cam, i, j) ;
                Intersection hit = Intersect (ray, scene) ;
                image[i][j] = FindColor (hit) ;
        }
        return image ;
}
```

“Taking a Picture”

Complex scene



Complex scene

Massive models

Highly detailed outdoor scene with over 3.1 billion triangles.

Massive models

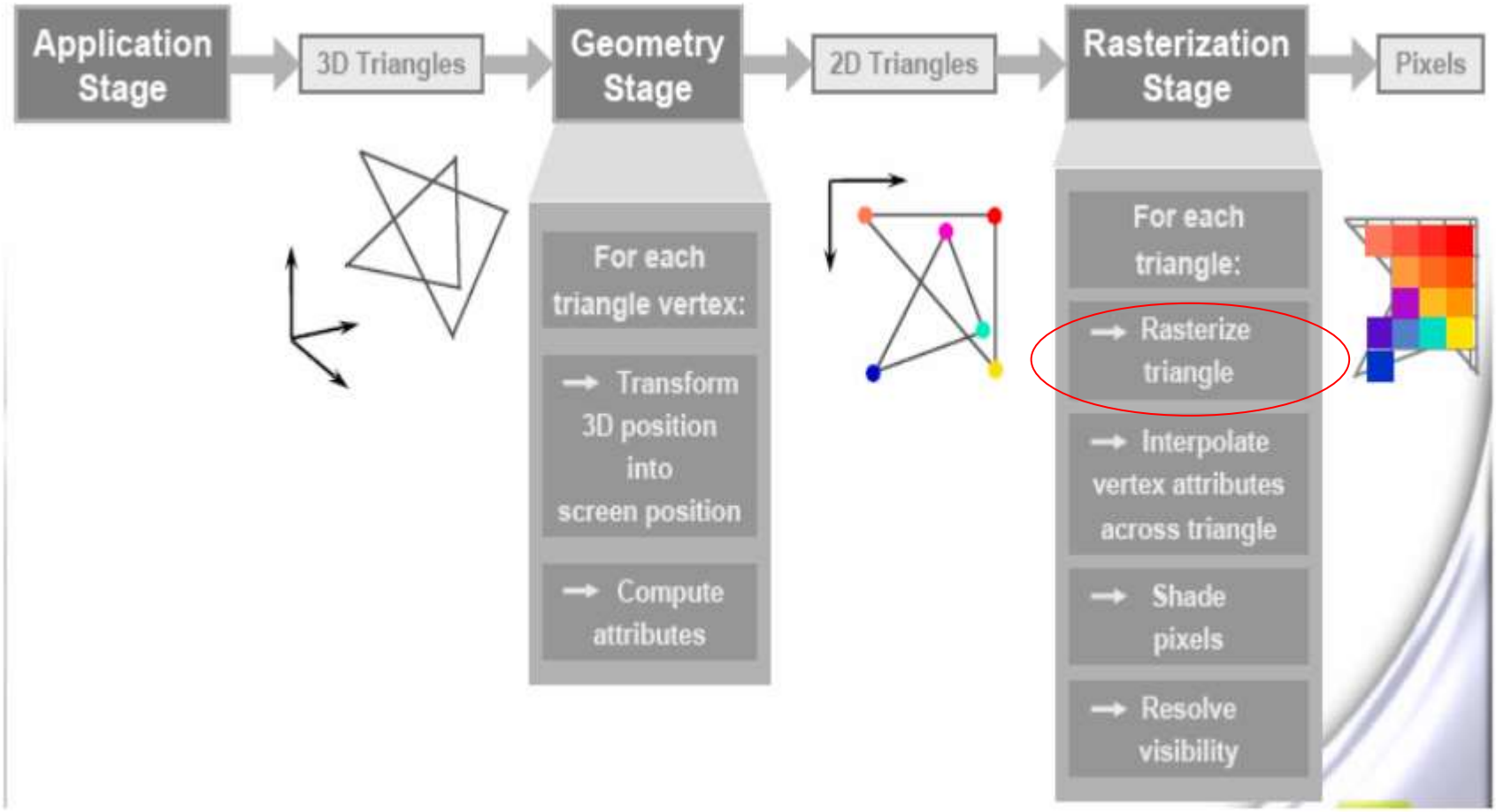
The original CAD model of a Boeing 777 consisting of 365 million polygons (30 GB). Ray tracing was the first method

Furry surfaces

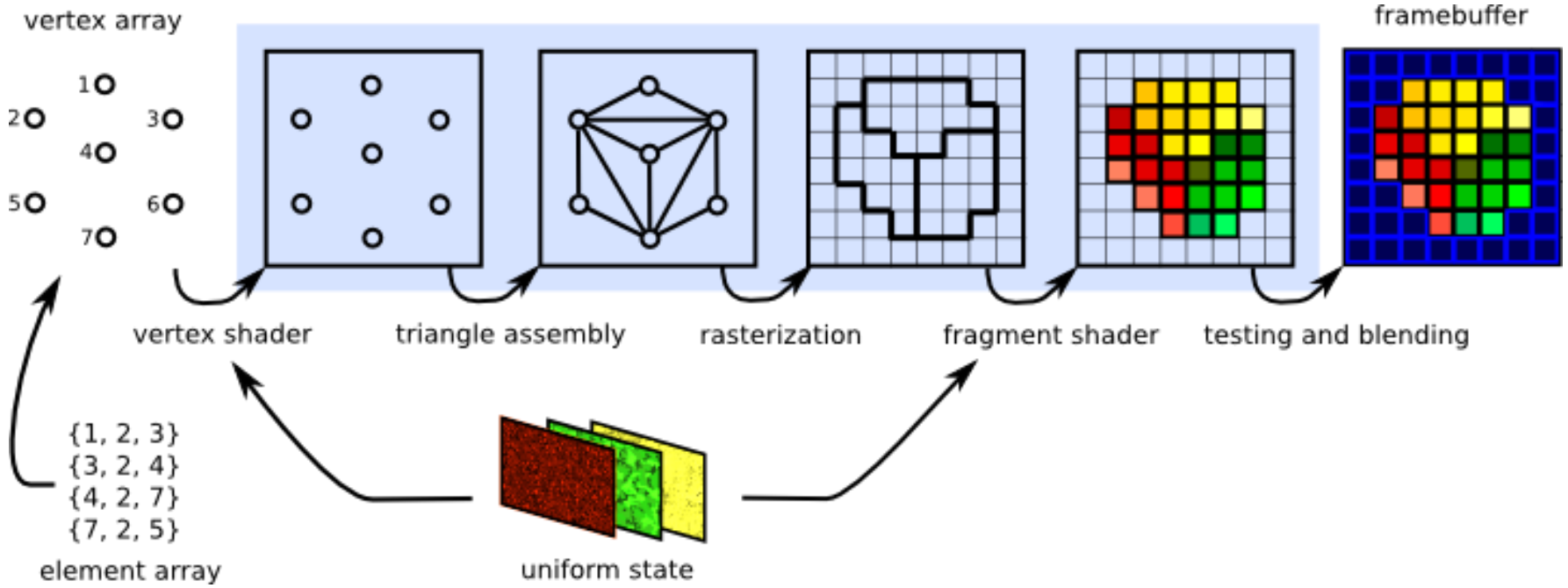
Refraction/Dispersion/Caustics

Translucency

PBR v.s. OpenGL Pipeline

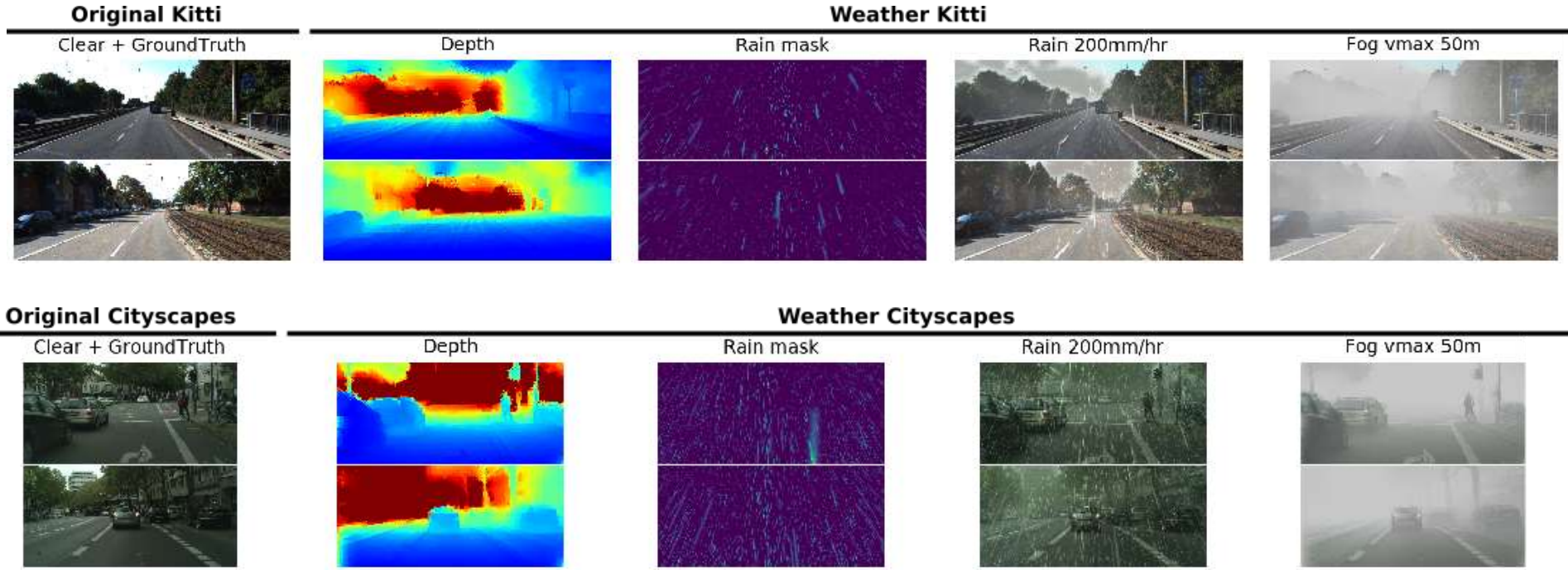


PBR v.s. OpenGL Pipeline



Rendering for ML/AI

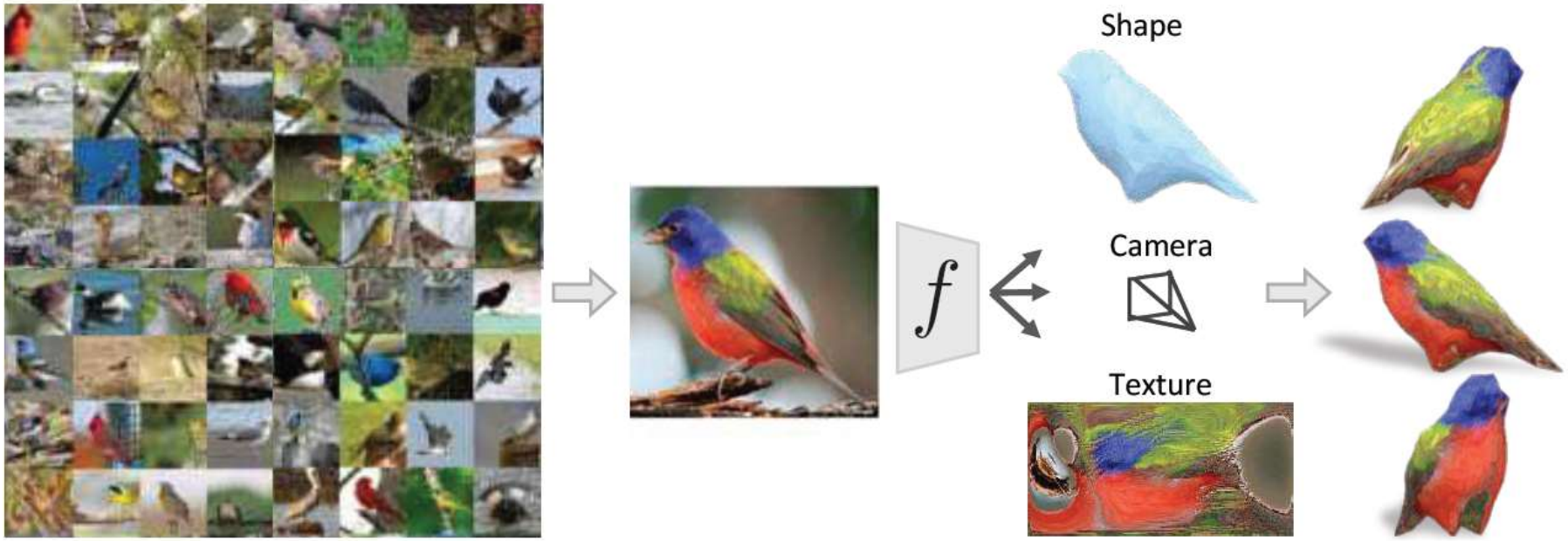
- Training data generation



Physics-Based Rendering for Improving Robustness to Rain (ICCV'19)

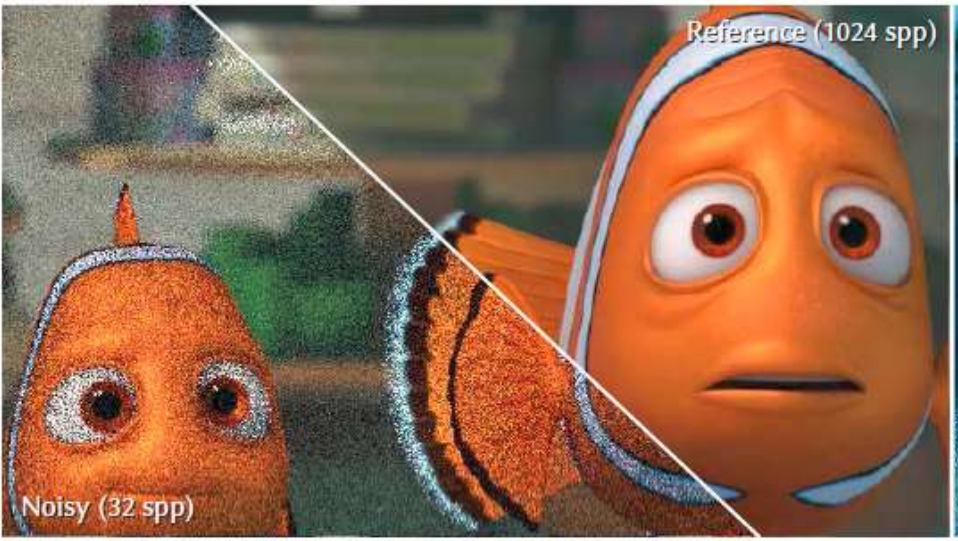
Rendering for ML/AI

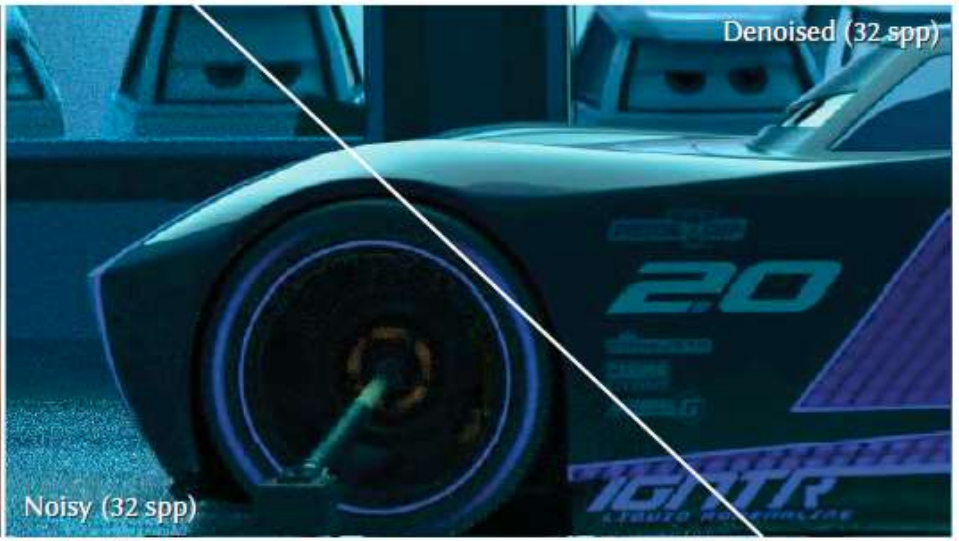
- Render loss design (differentiable rendering layer)



3D reconstruction

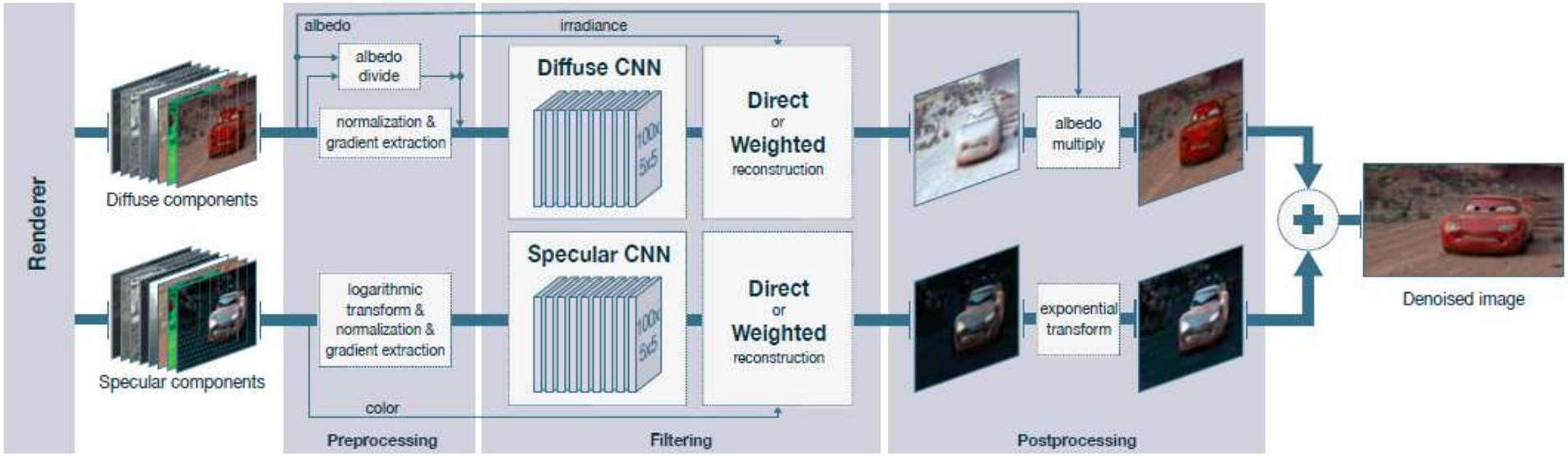
ML/AI for Rendering



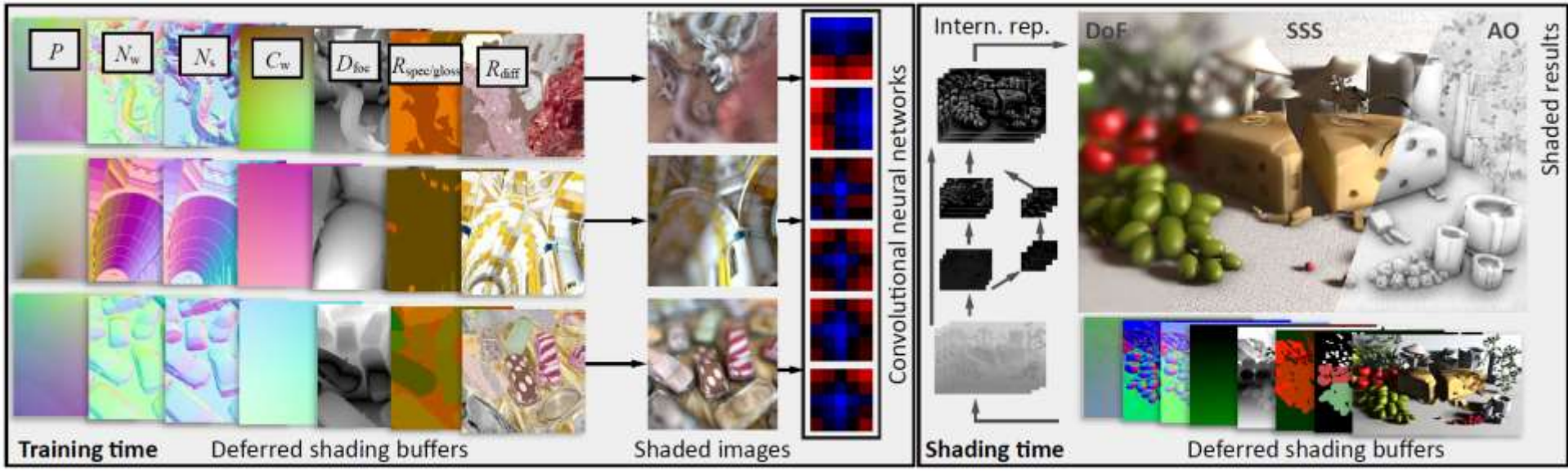


TRAINING

TEST

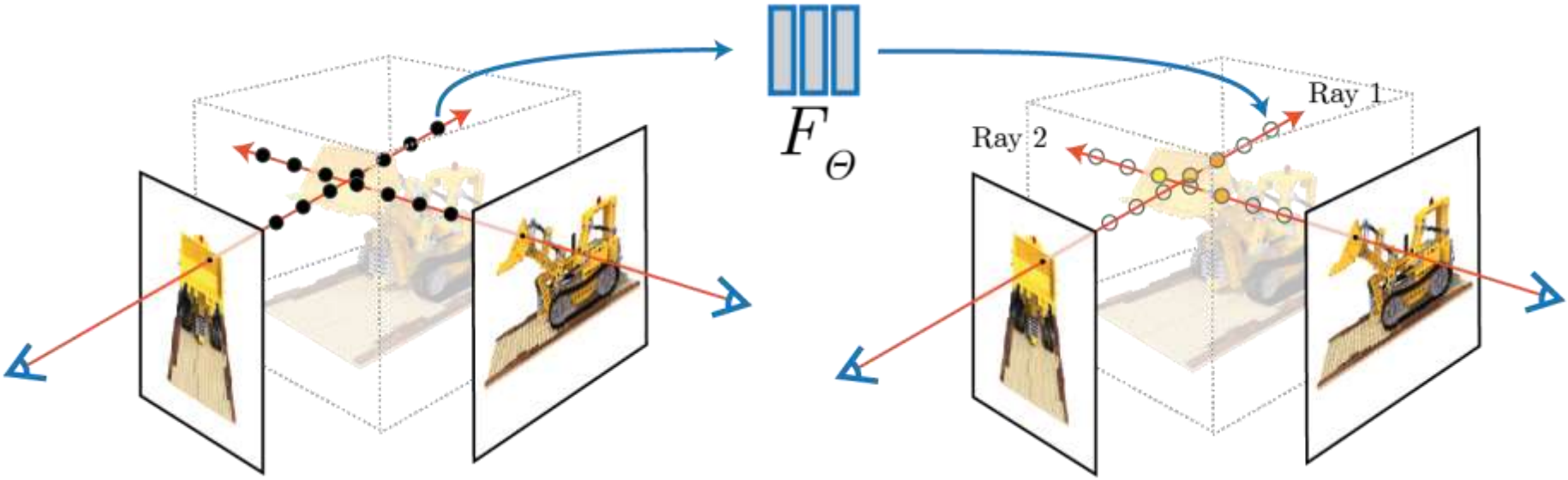


ML/AI for Rendering



DeepShading

ML/AI for Rendering



NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

ML/AI for Rendering