

Welcome!

CS-E5520 Advanced Computer Graphics Spring 2022

Jaakko Lehtinen with TAs Pauli Kemppinen and Heikki Timonen

CS-E5520 Spring 2022 – Lehtinen

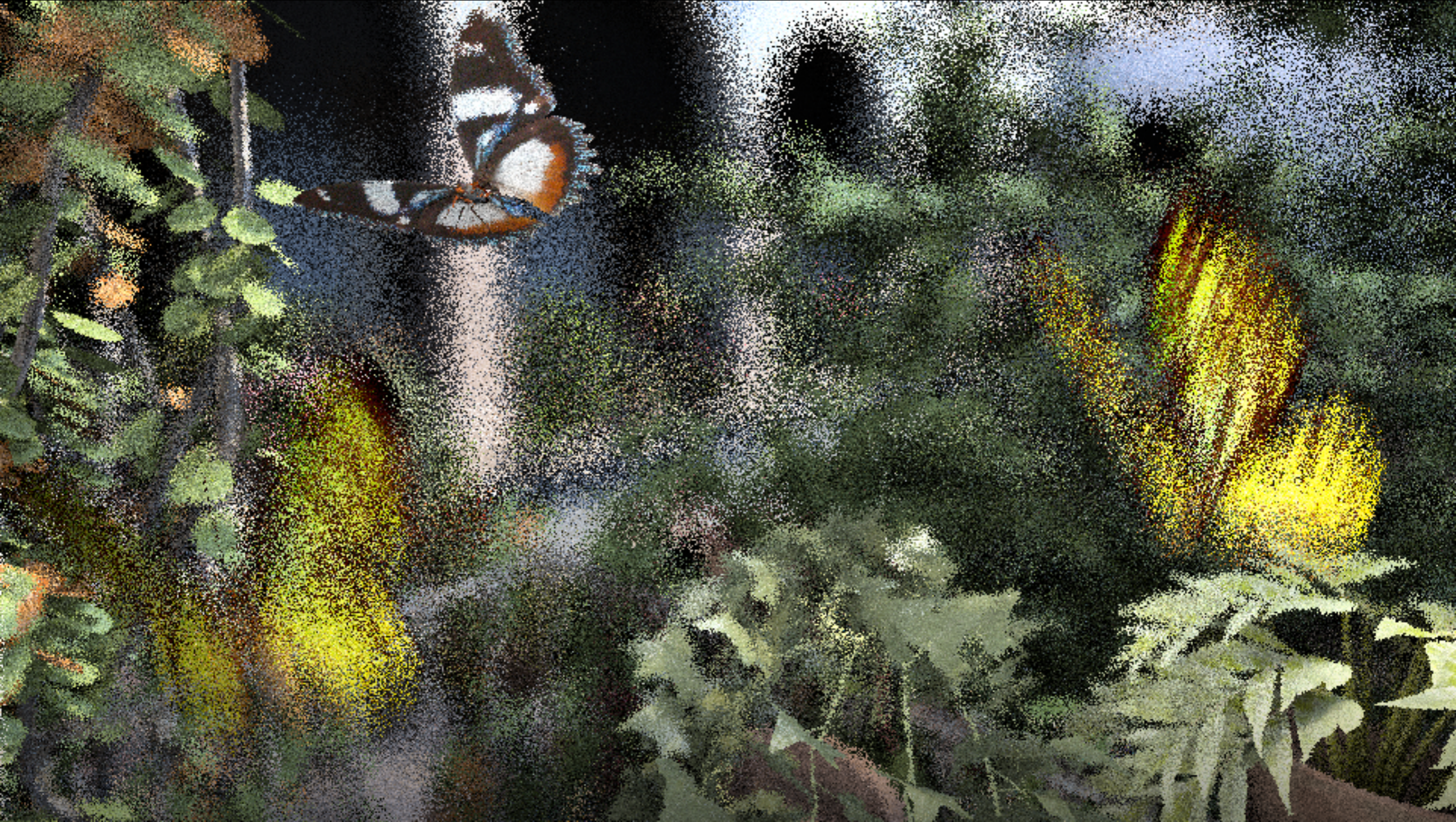
~1 quadrillion (10^{15}) FLOPs



1000x fewer FLOPs

4

Lehtinen et al. 2011





Smarter reconstruction from the **same input**



Smarter reconstruction from the **same input**



**Massachusetts
Institute of
Technology**



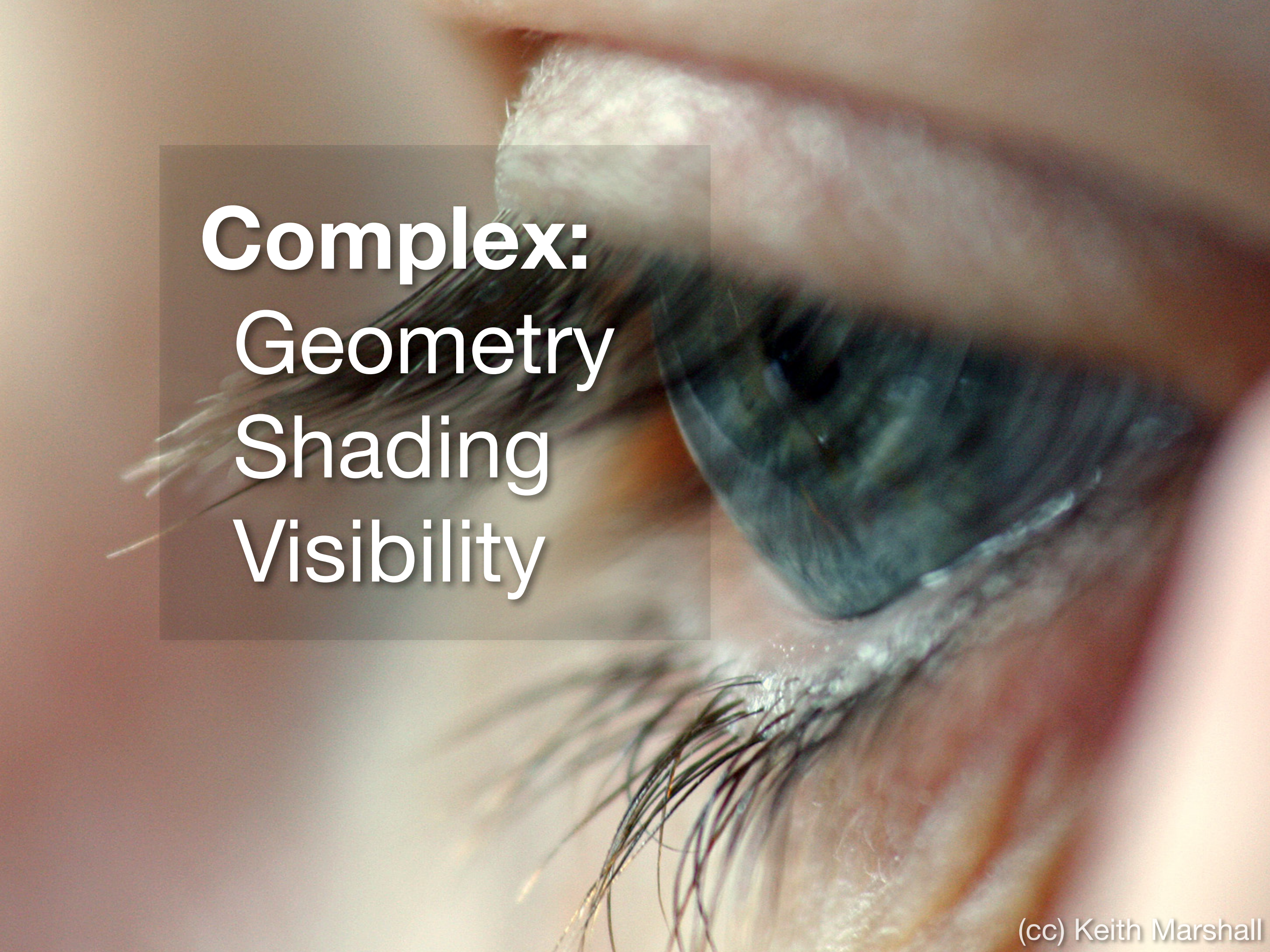
REMEDY



nVIDIA®

REMEDY





**Complex:
Geometry
Shading
Visibility**

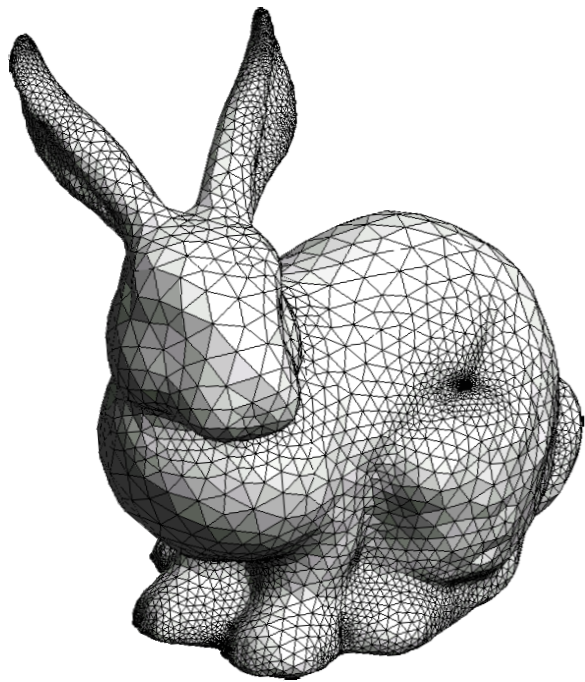
Rendering:

Compute what's visible.

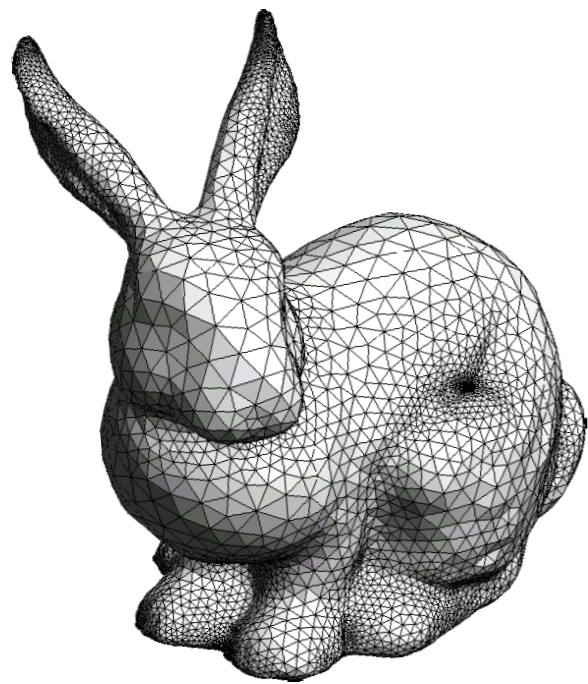
Compute what color it is.

What color it is:

Determined by interaction
of light and matter.



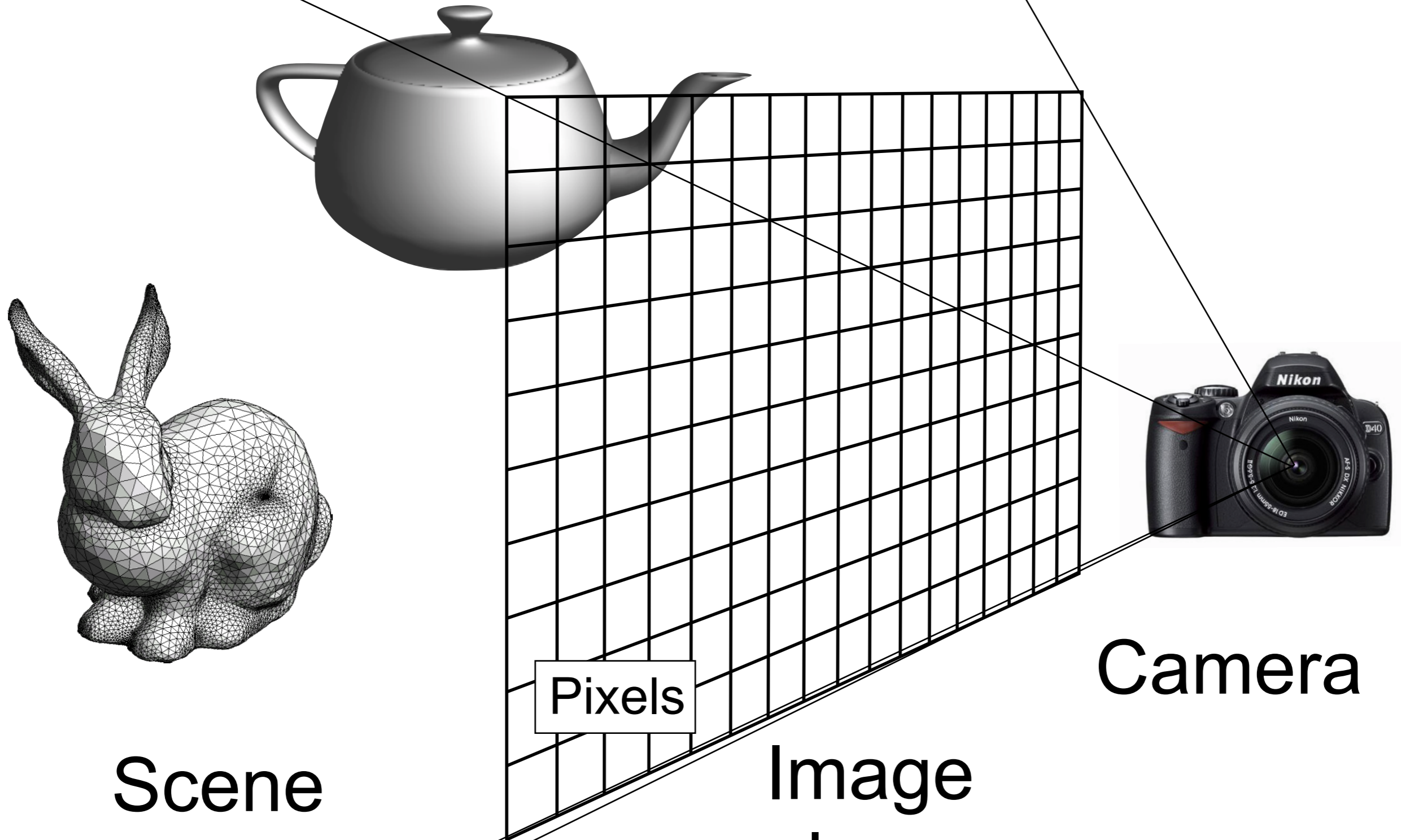
Scene



Scene



Camera



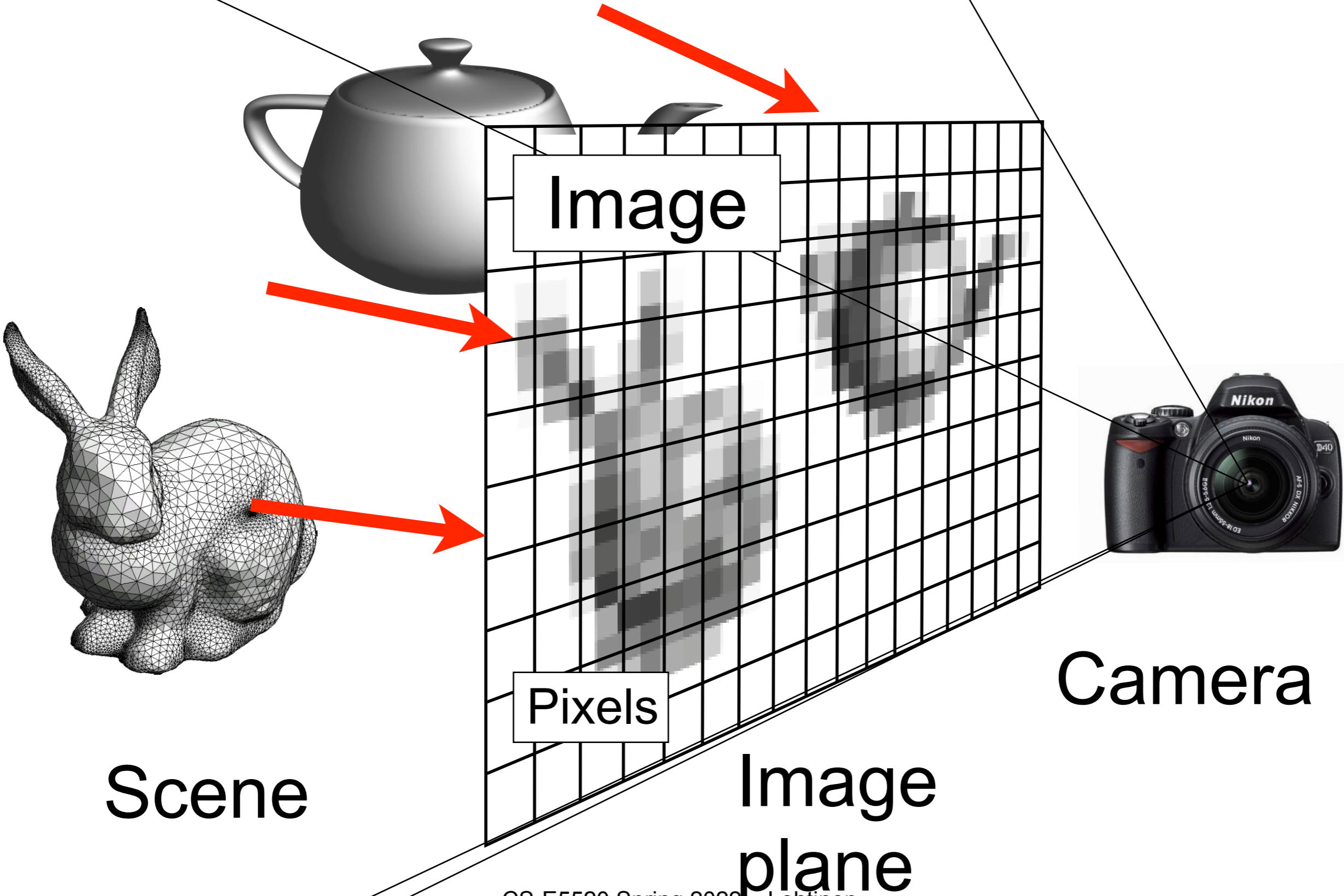
Scene

Pixels

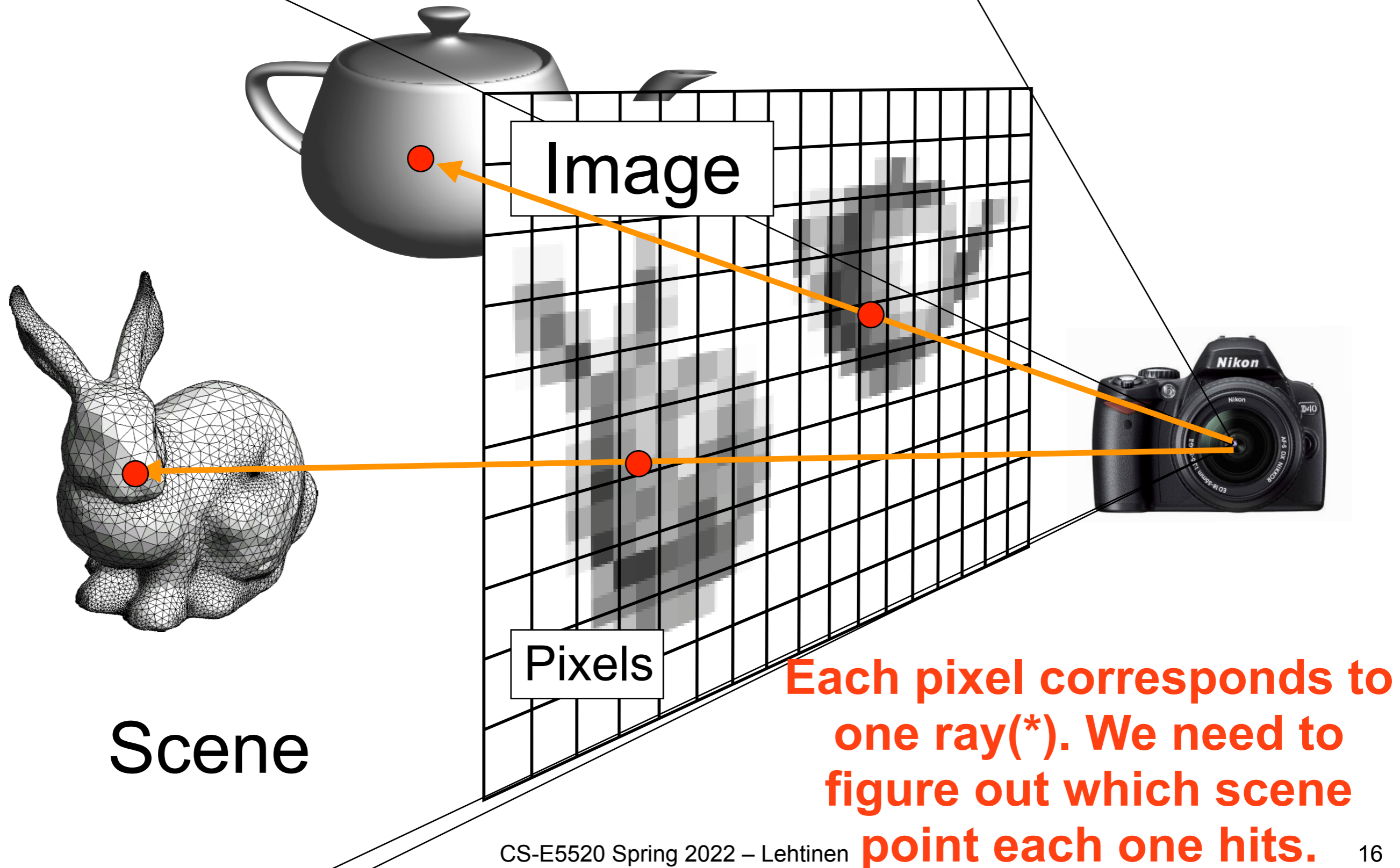
Image plane

Camera

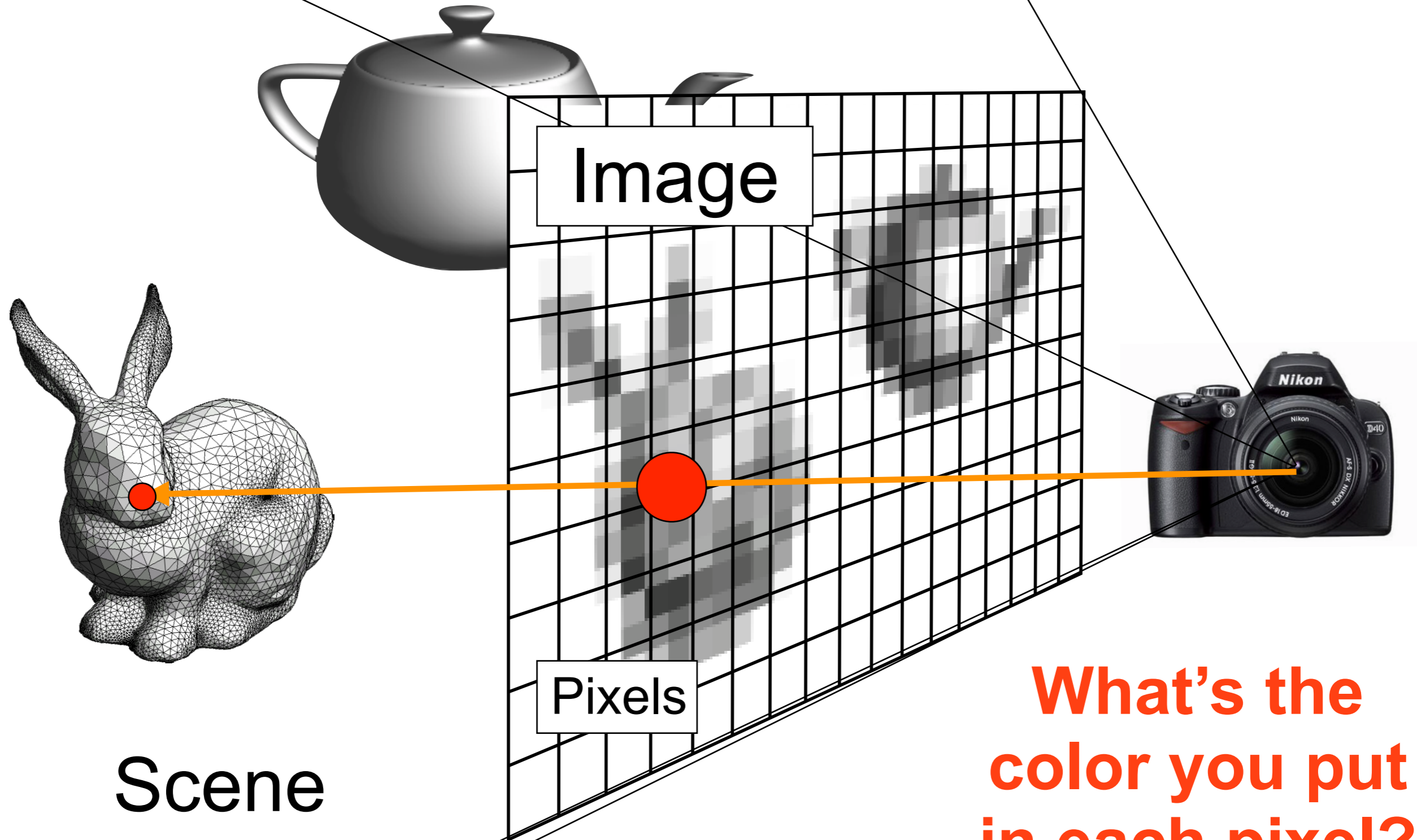
Rendering = Scene to Image



Rendering – Pinhole Camera



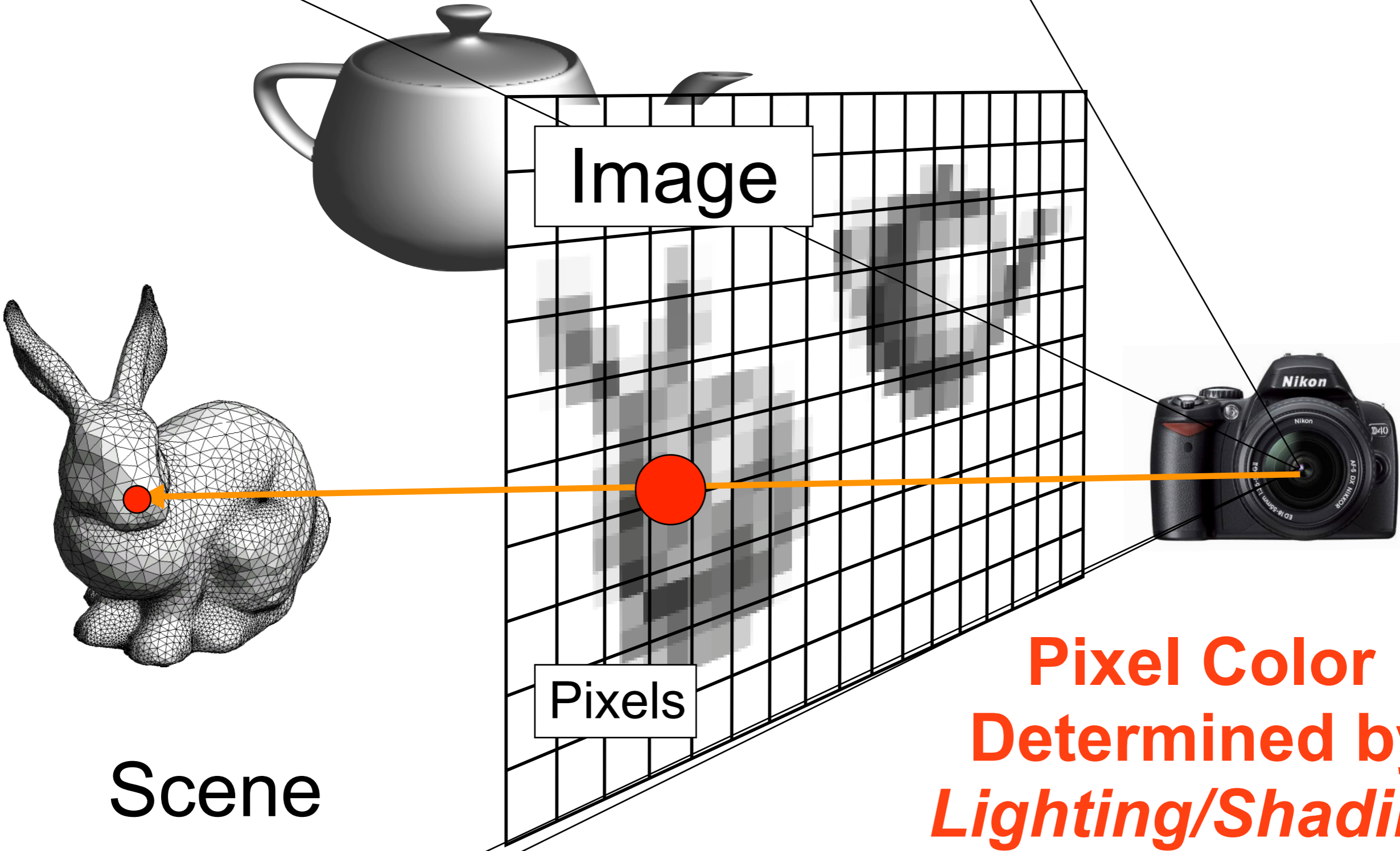
Rendering



Scene

What's the color you put in each pixel?

Rendering



Scene

Shading = What Surfaces Look Like

- Surface/Scene Properties

- surface normal
- direction to light
- viewpoint

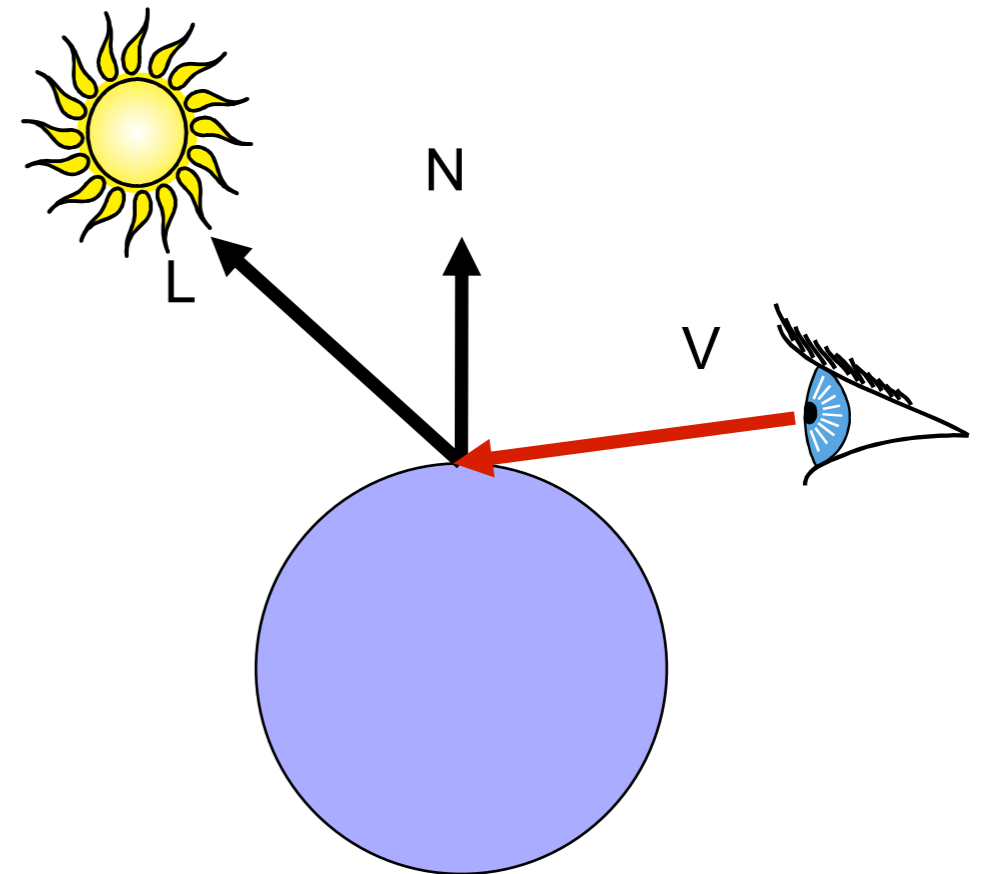
- Material Properties

- Diffuse (matte)
- Specular (shiny)
- ...

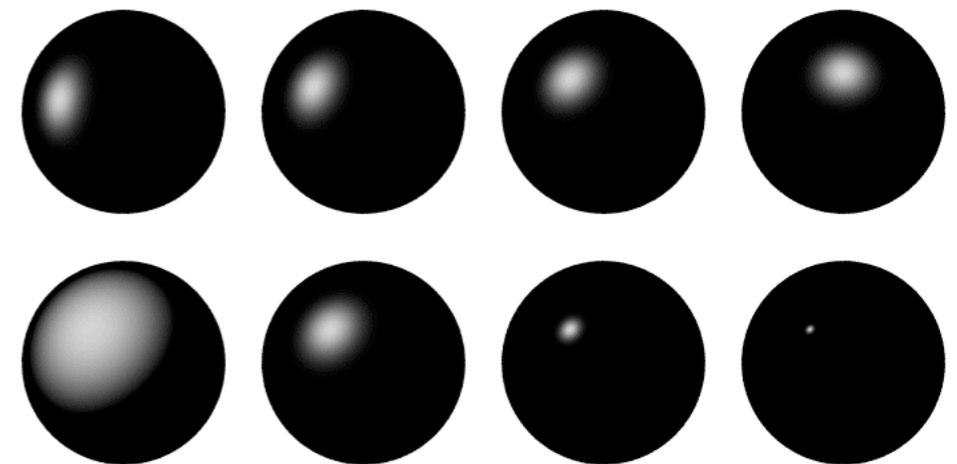
- Light properties

- Position
- Intensity, ...

- Much more!



Diffuse sphere



Specular spheres

Interlude

Reflectance Capture By Parametric Texture Synthesis

Miika Aittala

Timo Aila

Jaakko Lehtinen

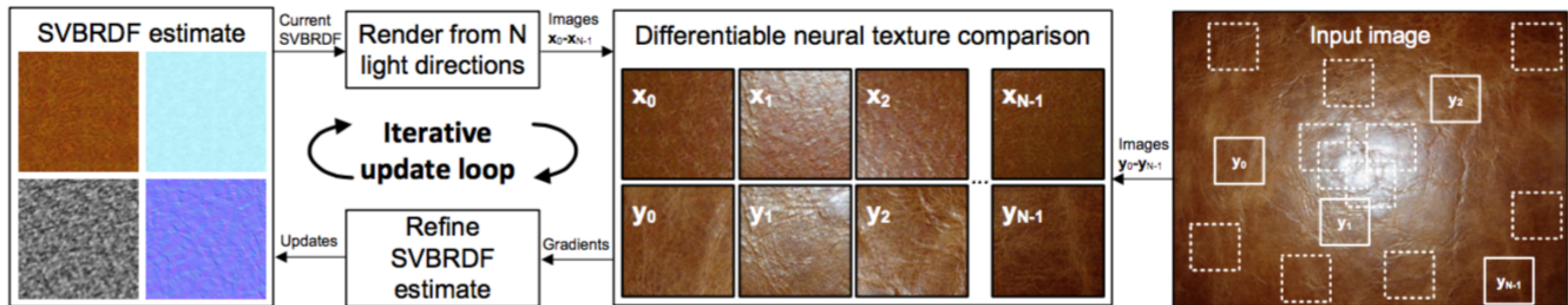


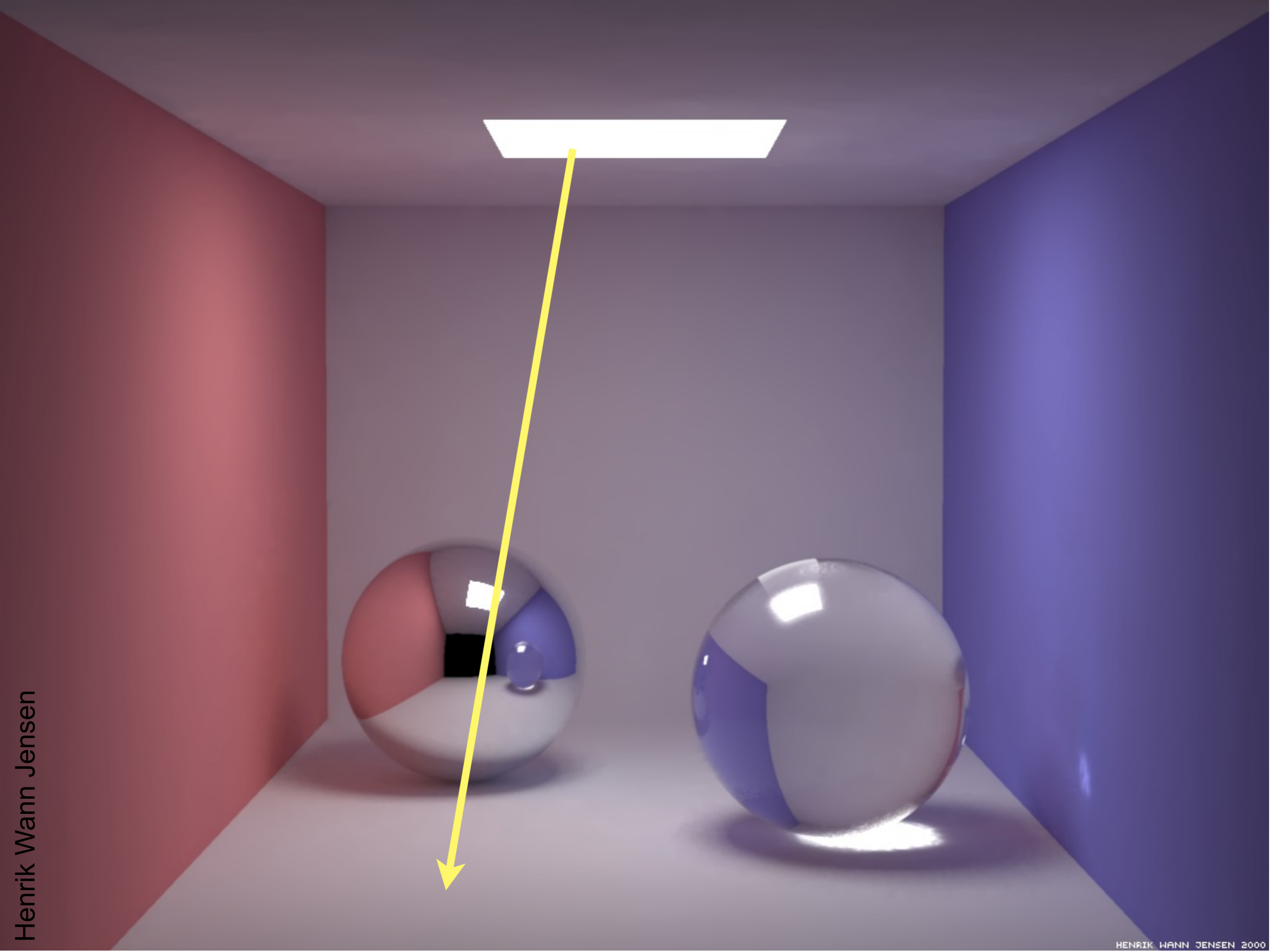
Figure 1: Our algorithm synthesizes a spatially varying BRDF that closely matches the input flash image when rendered from different lighting directions. The optimization process iteratively updates the current estimate based on neural network-based texture statistics comparisons that are able to ignore the precise pixel arrangement inside image tiles. This snapshot is from an early stage of the optimization.

Image Synthesis is Radiative Transport

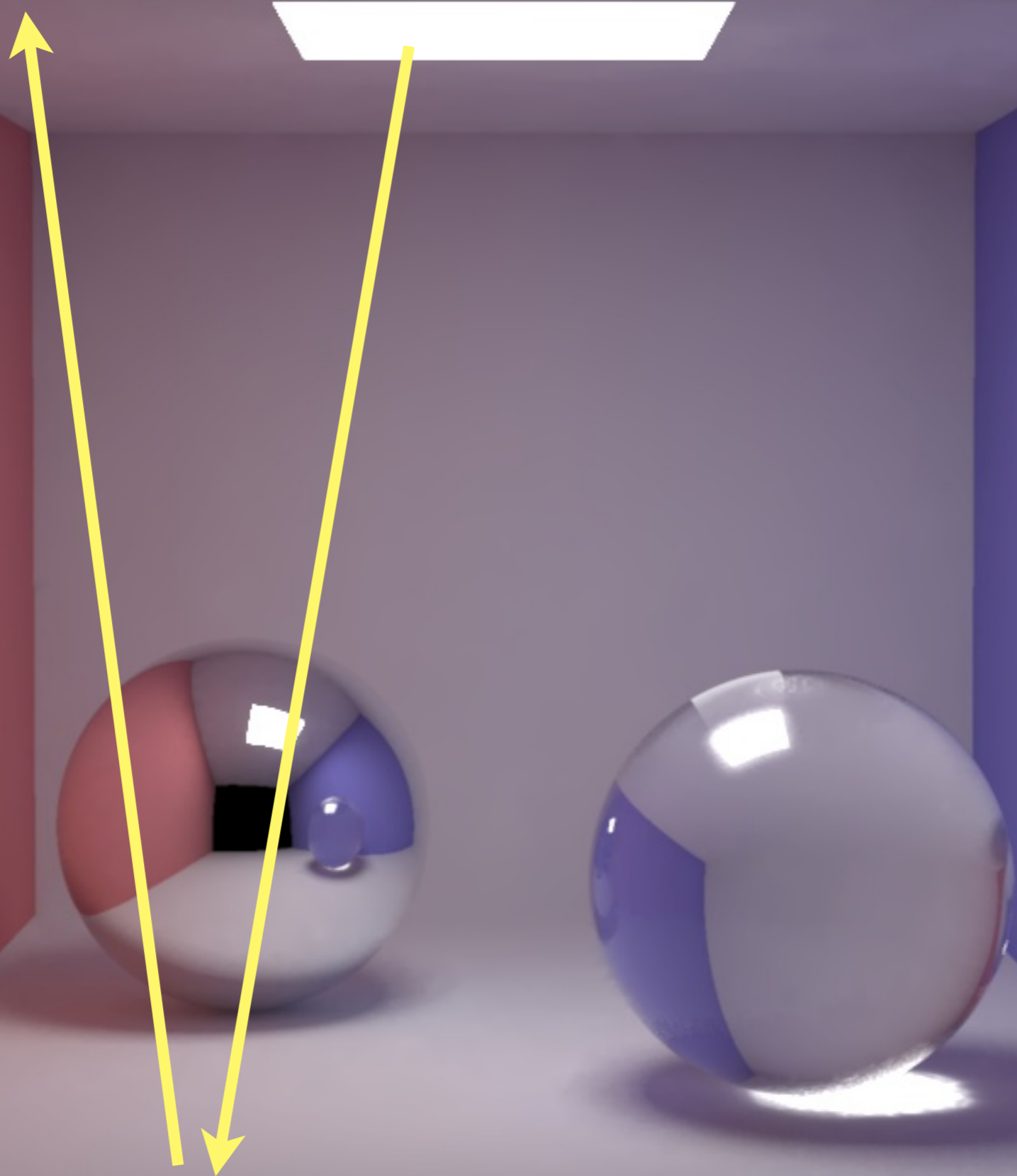
$$\begin{aligned}\omega \cdot \nabla_x L(x, \omega) = & \epsilon(x, \omega) \\ & - \sigma_t(x) L(x, \omega) \\ & + \sigma_s(x) \int_{4\pi} p(x, \omega, \omega') L(x, \omega') d\omega'\end{aligned}$$

“Volumetric Rendering Equation”

= Radiative Transport Equation (Chandrasekhar, 1960)

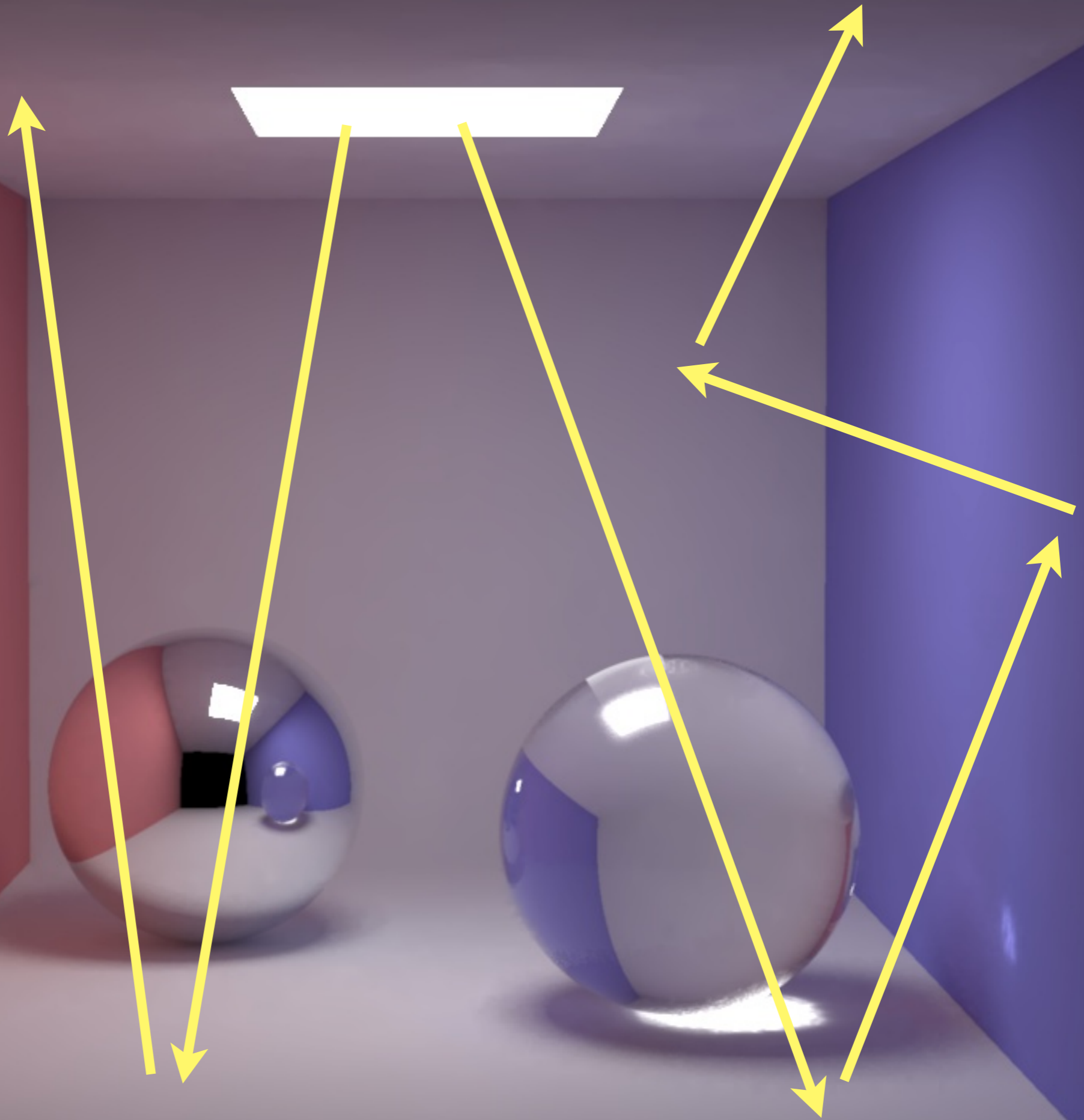


Indirect illumination

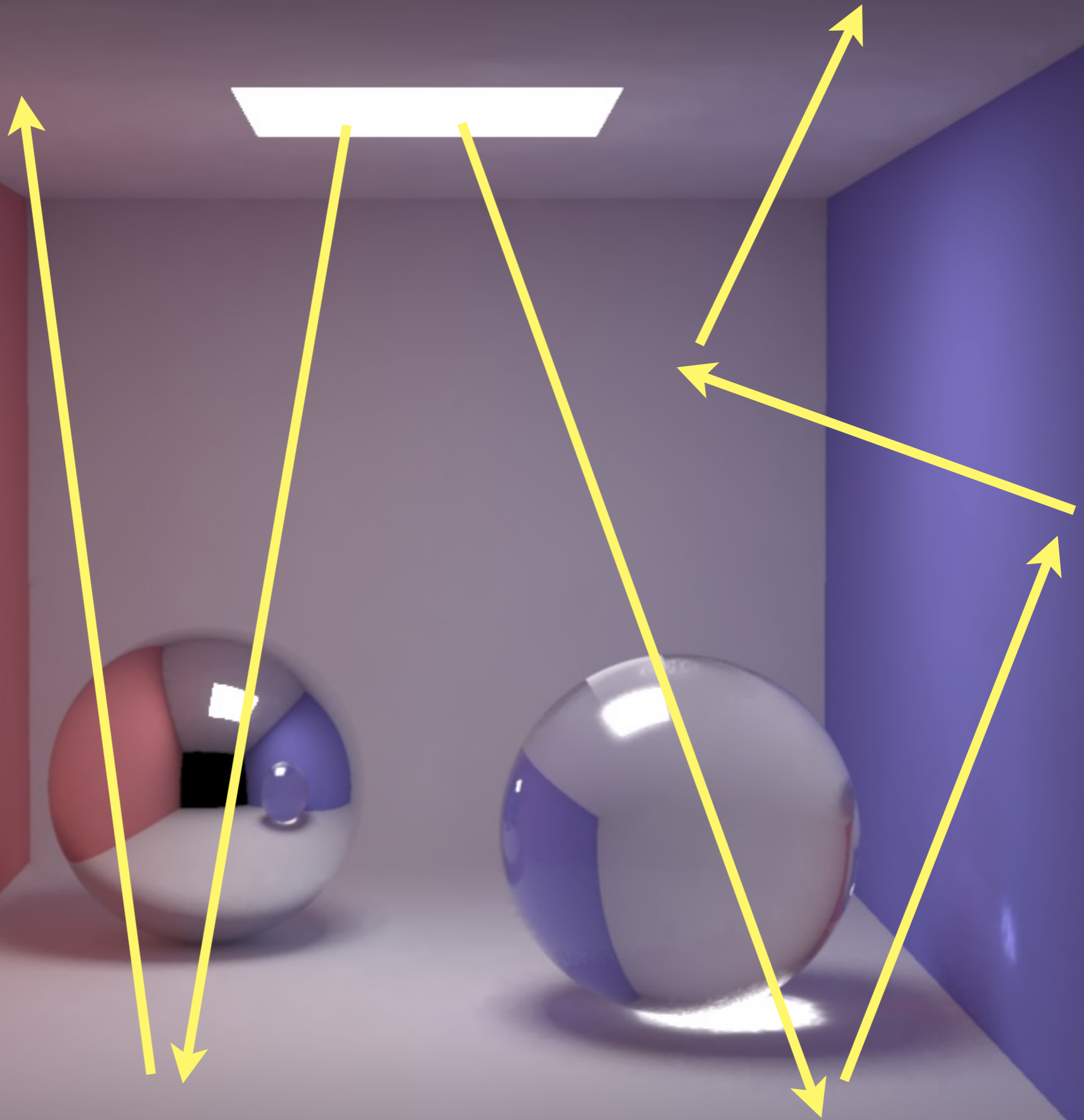


Henrik Wann Jensen

Indirect illumination

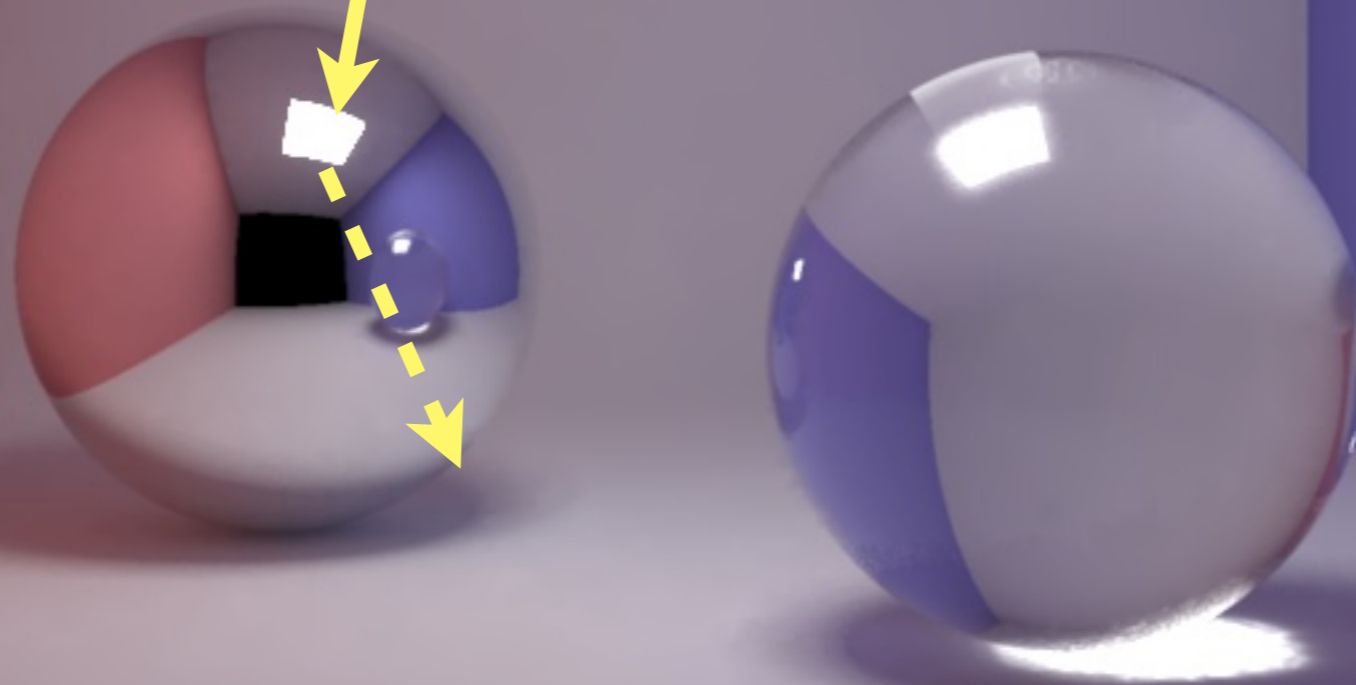


Indirect illumination

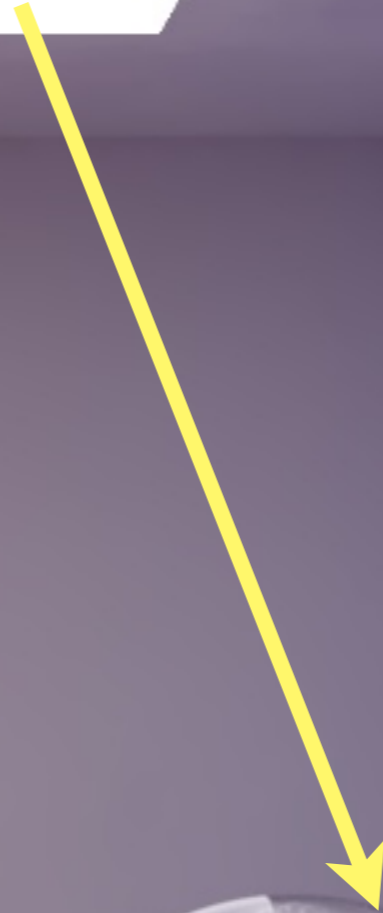


Indirect illumination

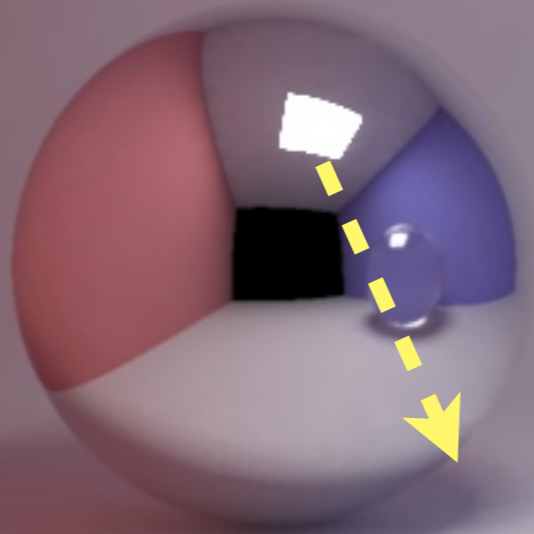
Reflections



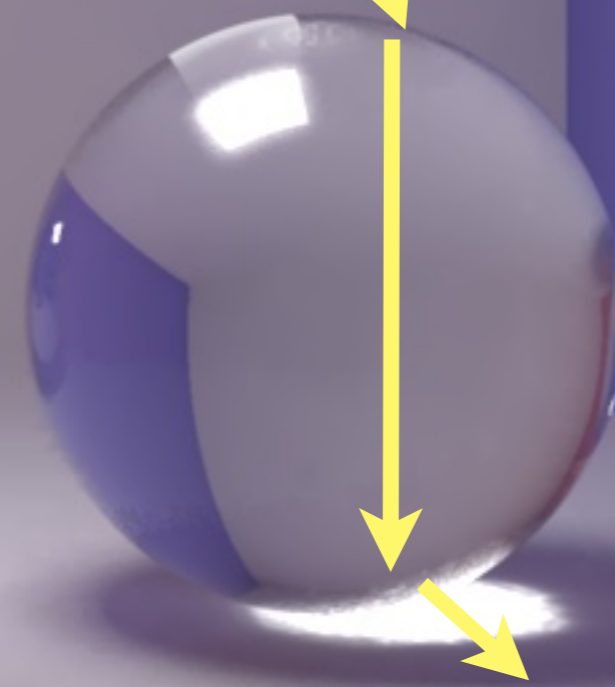
Indirect illumination



Reflections



Refractions



Caustics

Ray Casting vs. Rasterization

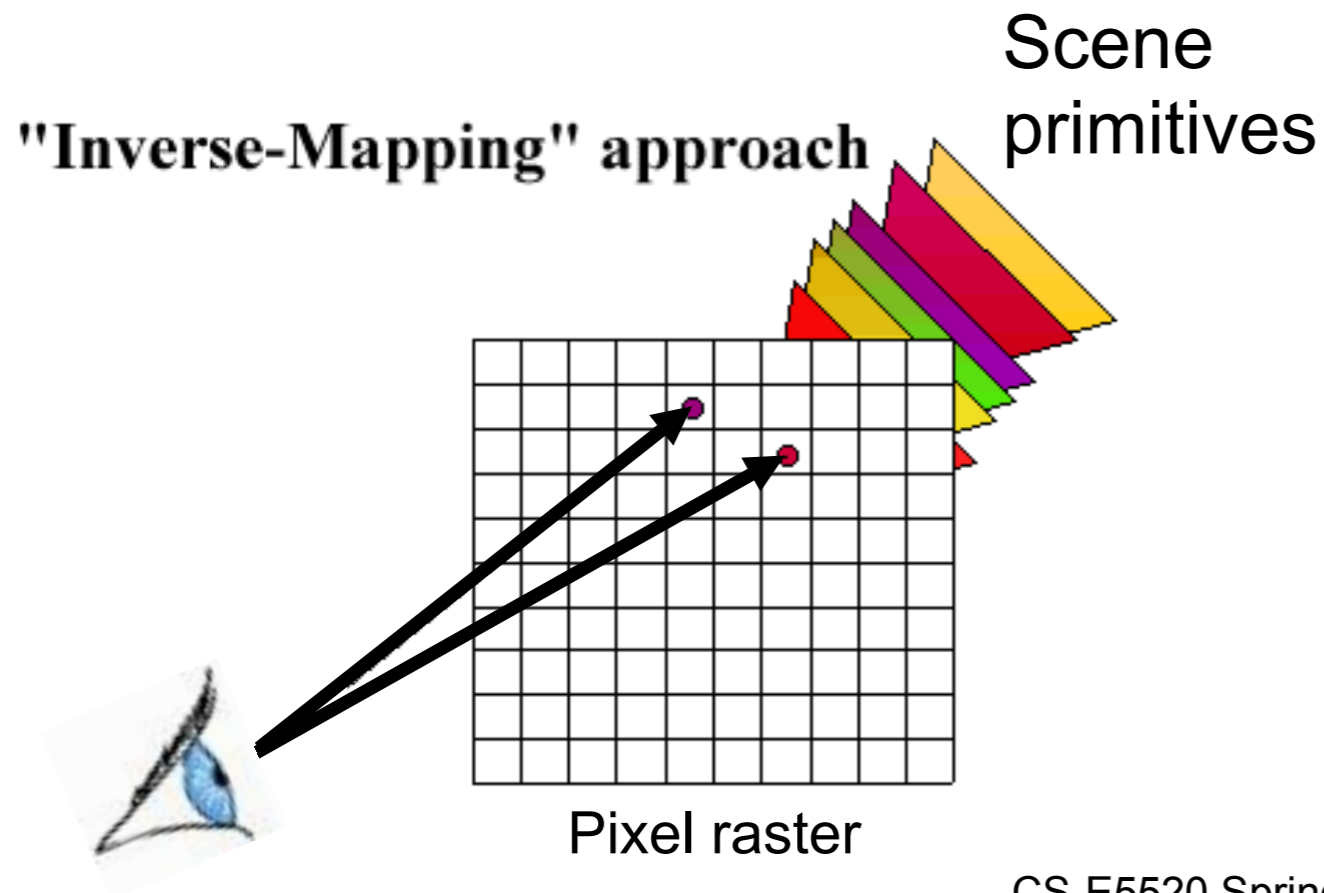
Ray Casting

For each pixel (ray)

For each object

Does ray hit object?

Keep closest hit



Ray Casting vs. Rasterization

Ray Casting

For each pixel (ray)

For each object

Does ray hit object?

Keep closest hit

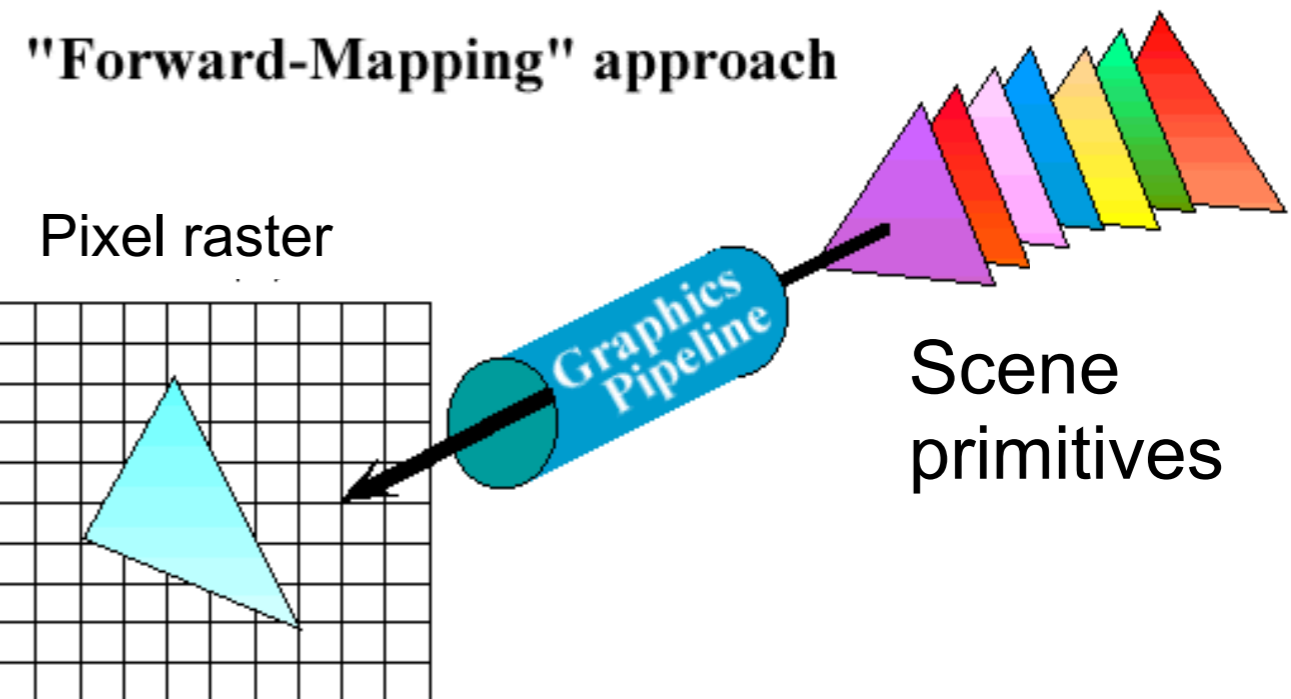
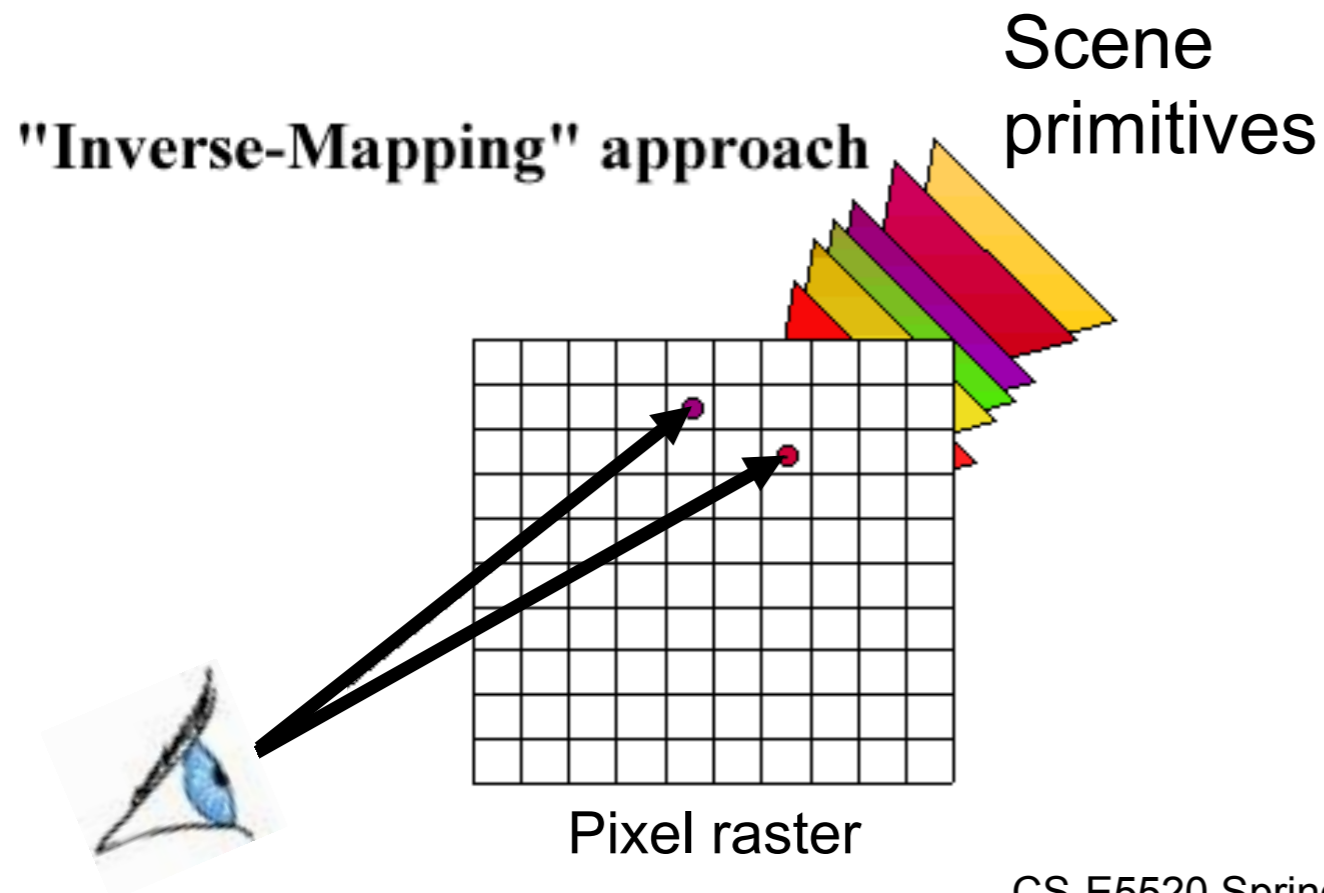
Rasterization

For each triangle

For each pixel

Does triangle cover pixel?

Keep closest hit



Ray Casting vs. Rasterization

Ray Casting

For each pixel (ray)

For each object

Does ray hit object?

Keep closest hit

Rasterization

For each triangle

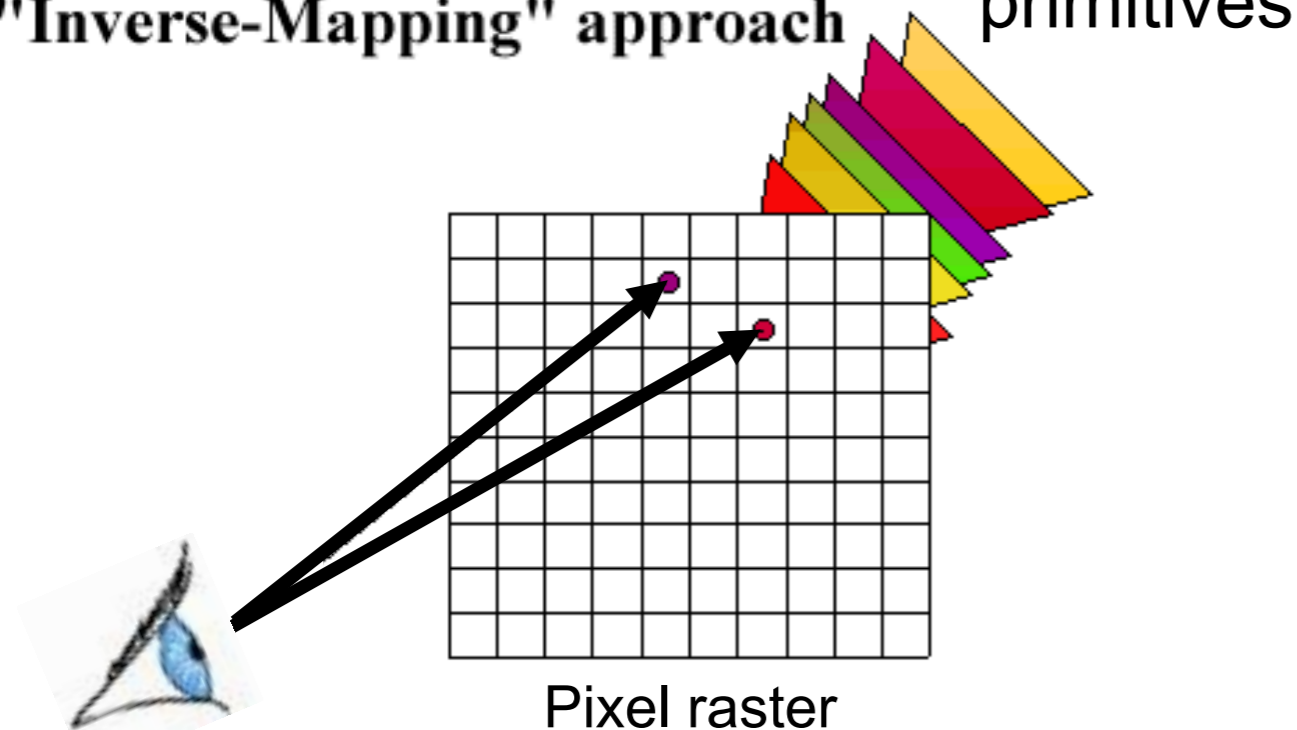
For each pixel

Does triangle cover pixel?

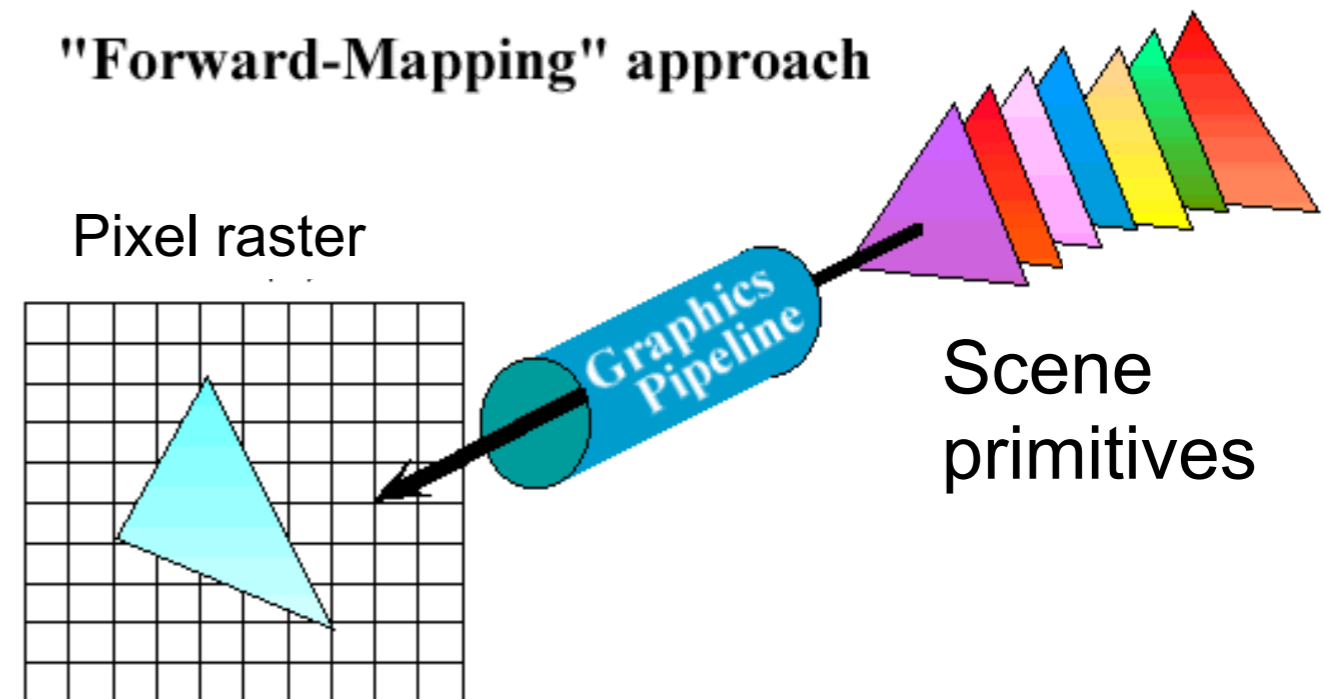
Keep closest hit

It's just a different order of the loops!

"Inverse-Mapping" approach



"Forward-Mapping" approach



Ray Tracing



- Advantages

- Generality: can render anything that can be intersected with a ray
- Easily allows recursion (shadows, reflections, etc.)

- Disadvantages

- Harder to implement in hardware (less computation coherence, must fit entire scene in memory, worse memory behavior)
 - Not such a big point any more given general purpose GPUs
- Has traditionally been too slow for interactive applications..
- **..but today, interactive ray tracing is reality!**

Ray Casting / Tracing



- Advantages

- Generality: can render anything that can be intersected with a ray

- Easily allows recursion (shadows, reflections, etc.)

- Disadvantages

- Harder to implement in hardware (less computation coherence, must fit entire scene in memory, worse memory behavior)

- Not such a big point any more given general purpose GPUs

- Has traditionally been too slow for interactive applications..

- ..but today, interactive ray tracing is reality!

Our focus in this class


“Speed of Light”

Real-time ray tracing on an NVIDIA RTX GPU



Markus Otto/Winzenrender, Rendered using [Maxwell](#)



 winzenrender

Stack Studios, Rendered using Maxwell



New Line Cinema



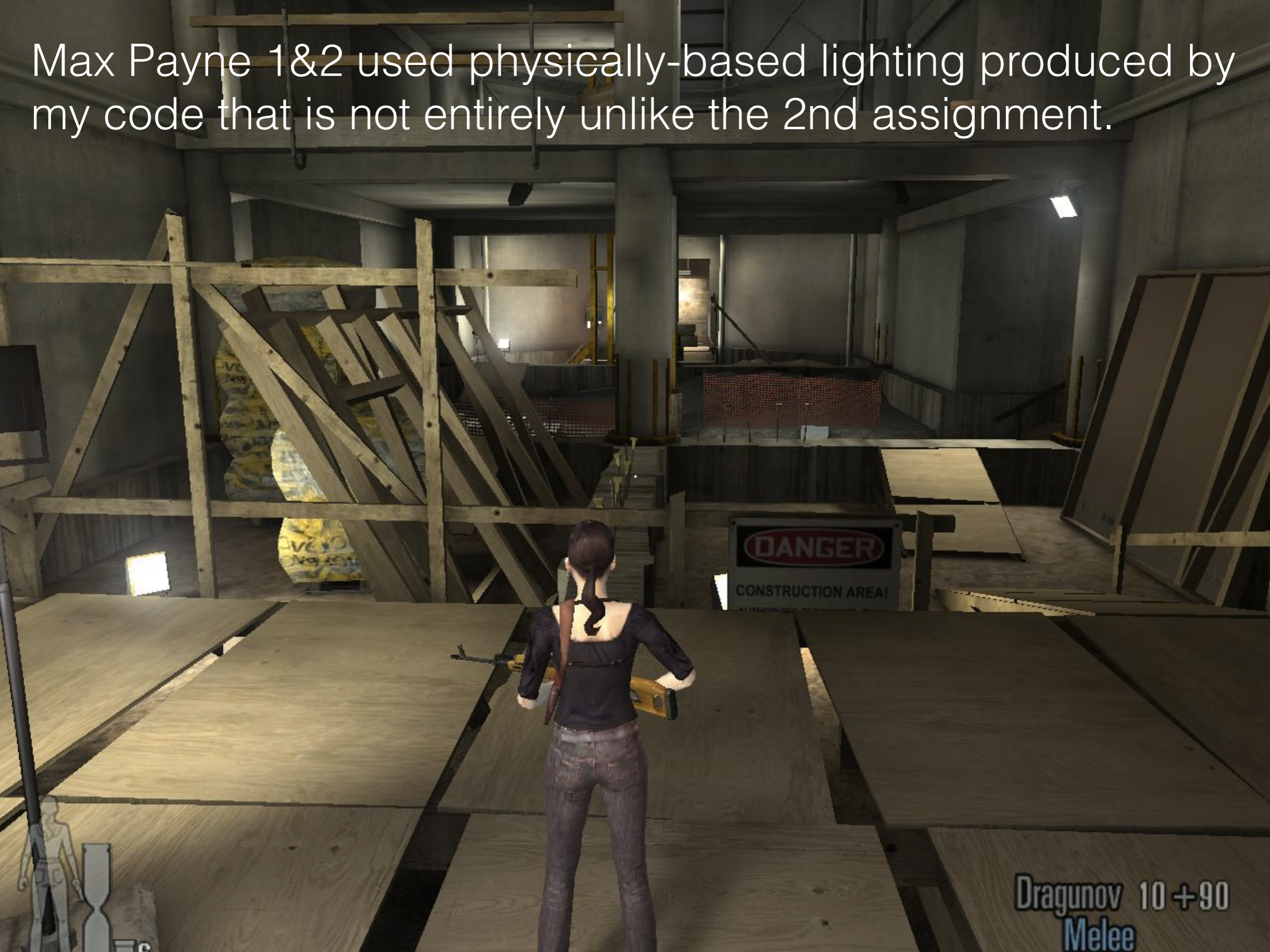
THE
HOBBIT
AN UNEXPECTED JOURNEY
DECEMBER 14, 2012
SEE IT IN REAL D 3D AND IMAX 3D



Also Real-Time using Traditional Methods
(video@58s)

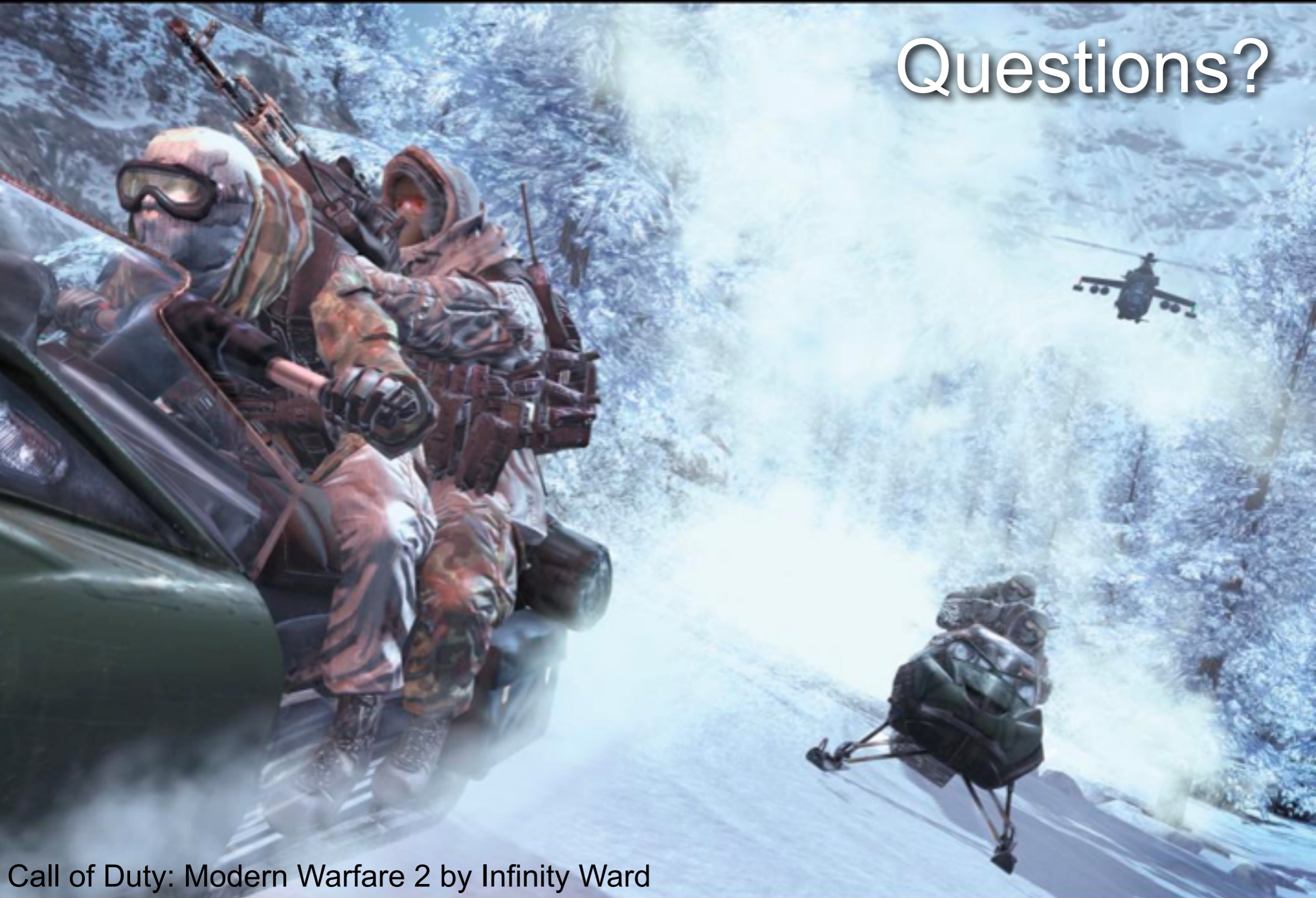
– Hedman, Karras, Lehtinen 2016. “Sequential Monte Carlo Instant Radiosity”, Proc. SIGGRAPH Symposium on Interactive 3D Graphics and Games 2016

Max Payne 1&2 used physically-based lighting produced by my code that is not entirely unlike the 2nd assignment.



Dragunov 10 +90
Melee

Questions?



Call of Duty: Modern Warfare 2 by Infinity Ward

Class Topics

- Efficient Ray Tracing
 - Basis for everything
- The Rendering Equation...
 - The single most important equation in graphics
 - Radiometry (how light is measured)
- ... and how to solve it
 - The bulk of the class and your assignments
 - Monte Carlo methods, Finite Element Methods (FEM)

You Will Code a Lot

- Four assignments
 - Accelerated ray tracer: Build and use a Bounding Volume Hierarchy, Soft Shadows, Ambient Occlusion
 - Radiosity: Compute diffuse global illumination at vertices, display interactively using OpenGL
 - What I did for Max Payne 1&2, except we used textures
 - Instant radiosity: The Real-Time Technique I just showed
 - Path Tracing (Monte Carlo Global Illumination)
 - Lots of extensions available for almost unbounded extra credit
- Still time to make suggestions!

C3100



E5520



Practical Details: Assignments

- Less starter code than in intro class
 - you will use **your own ray tracer in all of the assignments**
 - Well, in the first one you write it
 - If you fail, we'll provide a binary library you can link against, but this will impose a maximum on your score
 - Framework still there, don't worry

Practical Details: Assignments

- One-person projects
 - Code yourself, BUT talking to others highly encouraged
- *New this year*: Slack for asking questions and giving answers related to assignments (link on MyCourses)
- MyCourses is the official communications channel

- Grading
 - 100% of grade based on assignments
 - Strong effort on helping others on Slack may get bonuses

Final Showdown

- The last assignment will conclude with a rendering competition
 - **Your rendering code only**
 - Models and stuff like that can come from any legal source
 - Loader & framework code can be someone else's
 - Draw on everything you've learned

Assignments: Extra Credit

- Do everything you're asked for, you get a 5
 - It's a little harder than in C3100, but not much. No panic.
- BUT each assignment has a long list of things you can do for extra credit

- Why bother?
 - 1. It's fun
 - 2. Do cool stuff and you will be on the radar of people who might want to hire you (anecdotes)

Up to YOU





No Upper Limit

Practical Details: Assignments

- **Deadlines are absolute:** 0 if not on time
 - If you need extra time, must ask for it 1 week in advance
- **Coding environment:** MS Visual Studio 2019
 - Same environment as CS-C3100 this past fall
- **You must turn in code that compiles in VDI “Windows 10 3D” instance**
 - You can code on your own setup
- **Always turn in README file where you tell us how long the assignment took, who you collaborated with, what was unclear/difficult, etc.**

Tentative Schedule 2020

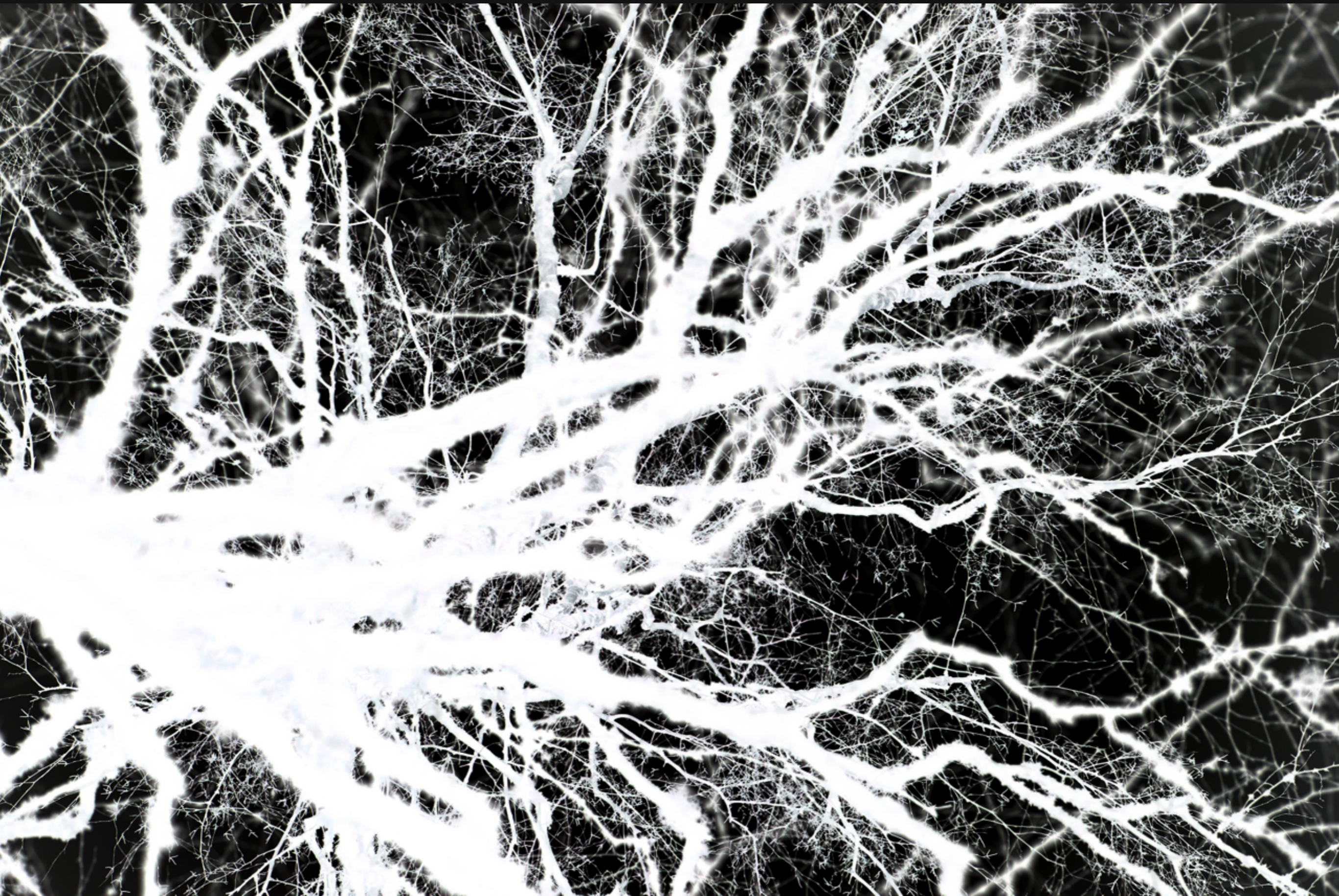
- Assignment 1: Fast Ray Tracing, DL 13.2.
- Assignment 2: Radiosity, DL 6.3.
- Assignment 3: Instant Radiosity, DL 3.4.
- Assignment 4 + Rendering Compo, DL 3.5. (Tue!)
- **MyCourses is the definitive schedule!**
- Lectures will be mostly up front, will go out of sync with exercises
 - Must get out of the way, will take parental leave from March

Practical Details: Admin

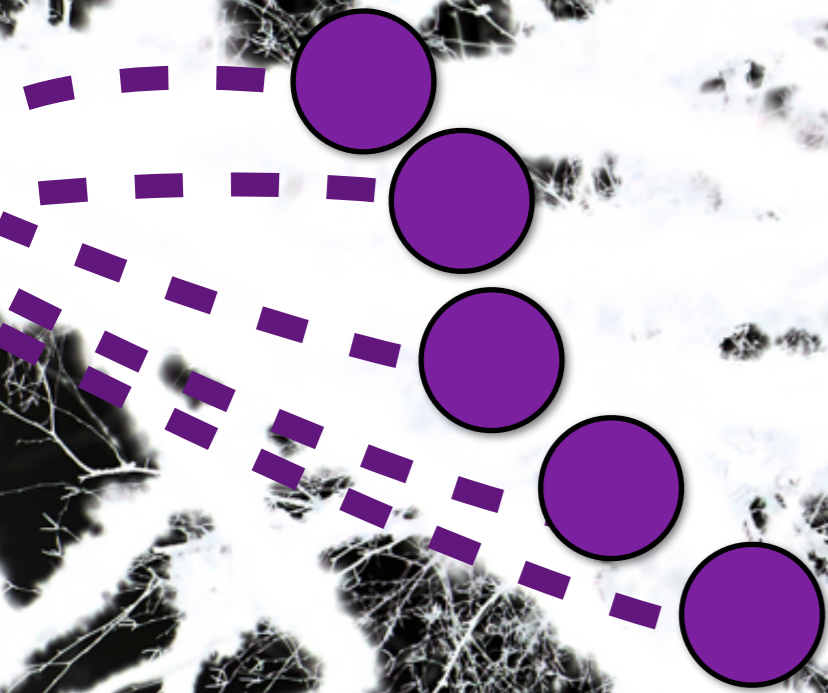
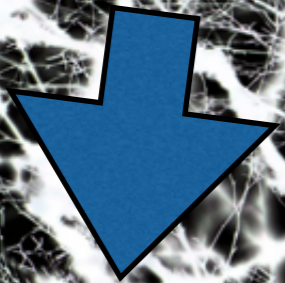
- Class email: cs-e5520@aalto.fi
 - *All communication not related to assignments here!*
- TAs
 - Pauli Kemppinen, Heikki Timonen
 - Will patrol Slack

Concluding remarks: The Role of Research

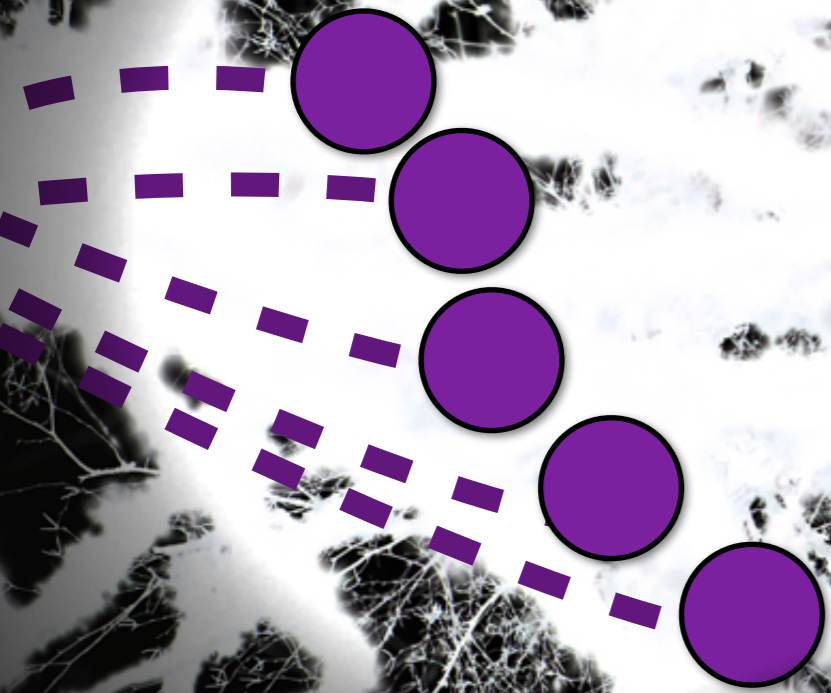




State of the art in applications

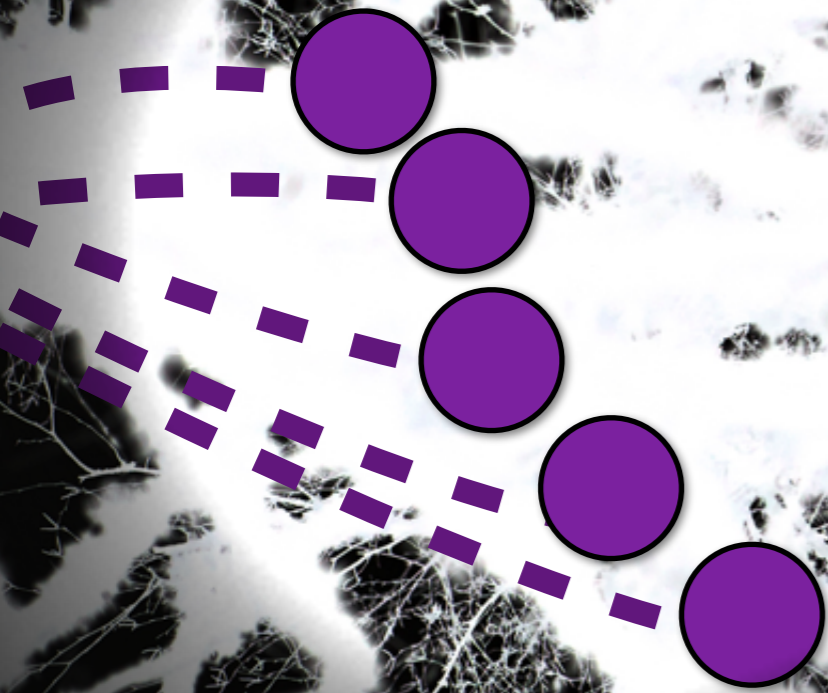


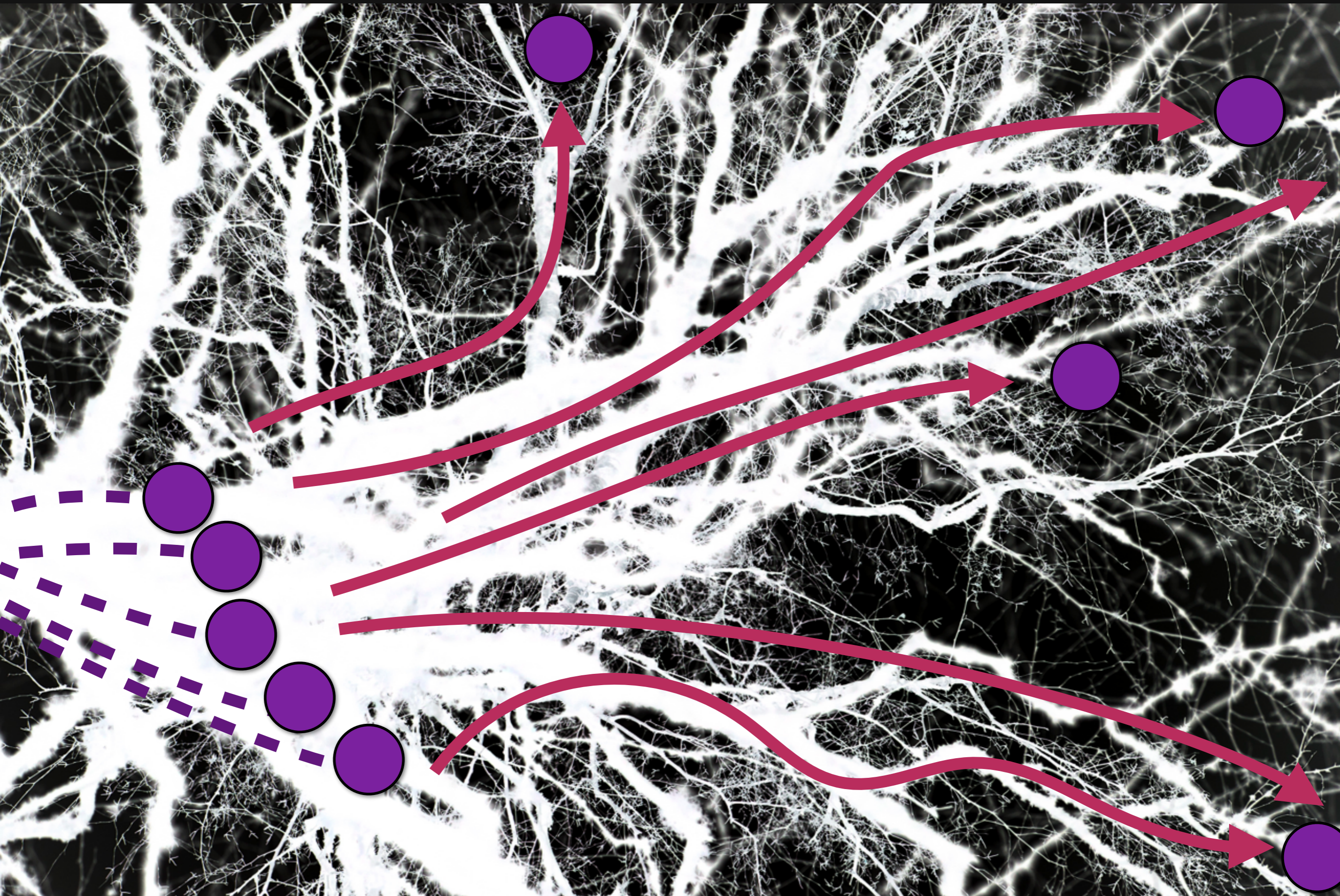
Reality: if you just focus on what's possible today, you suffer from serious myopia

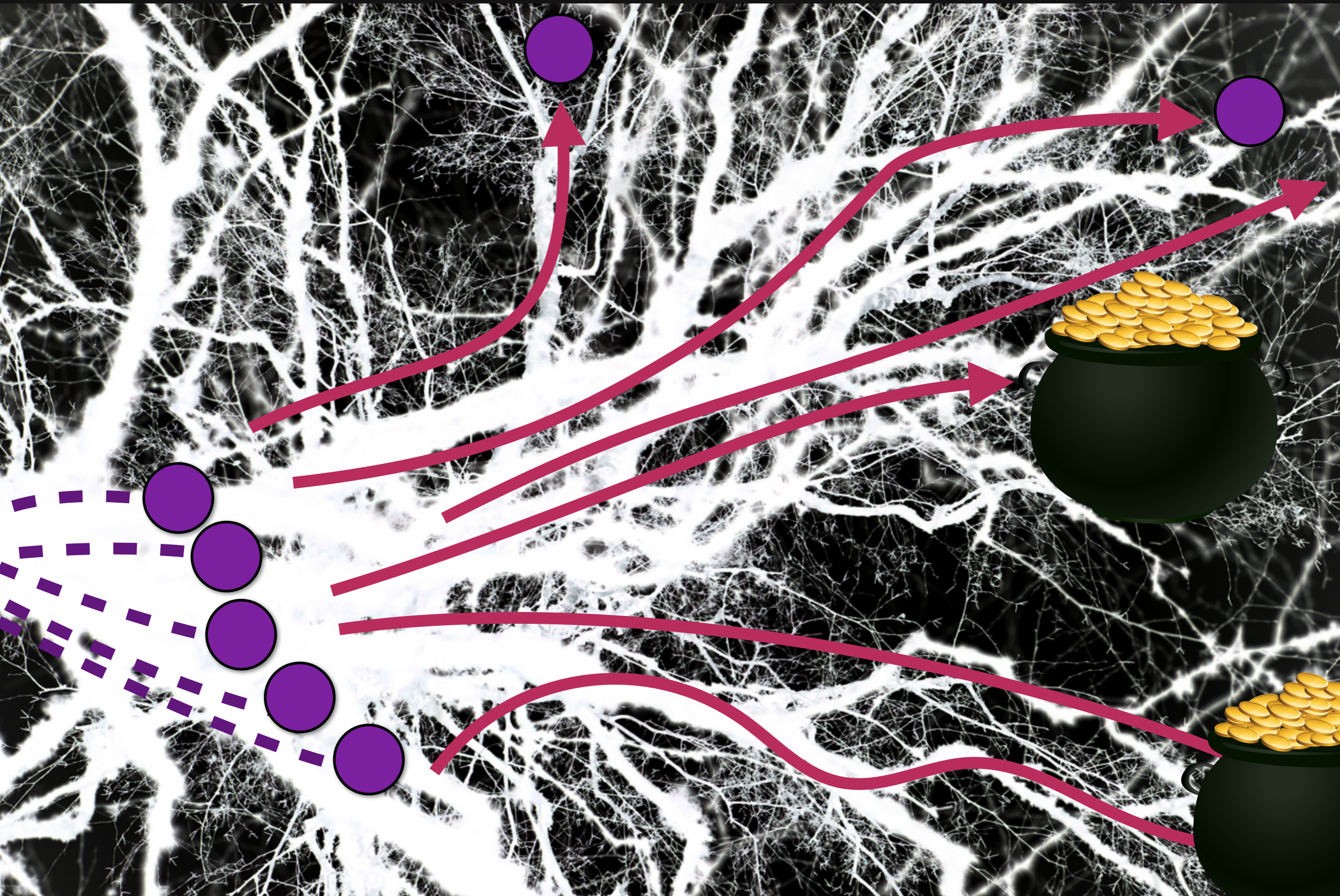


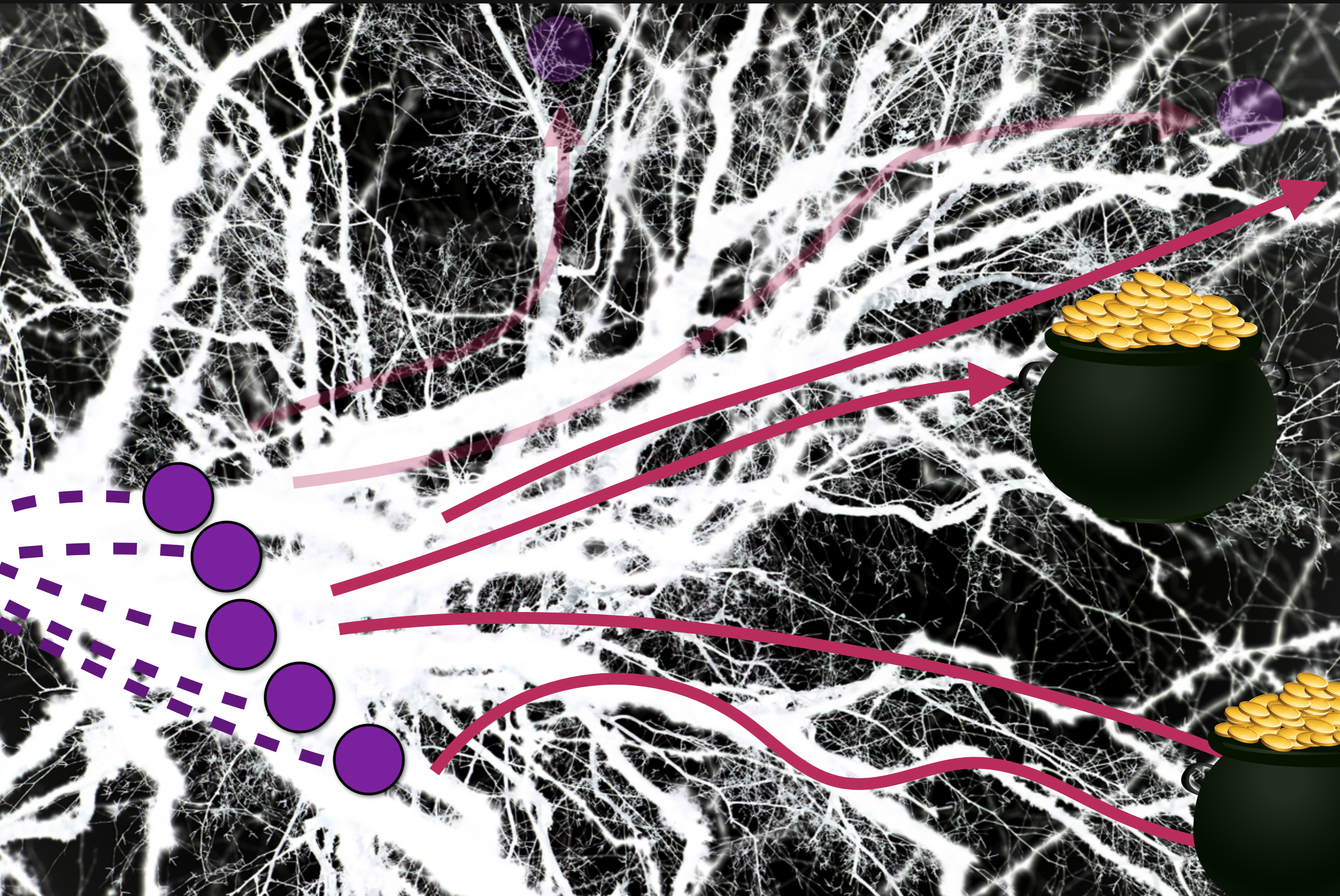
Reality: if you just focus on what's possible today, you suffer from serious myopia

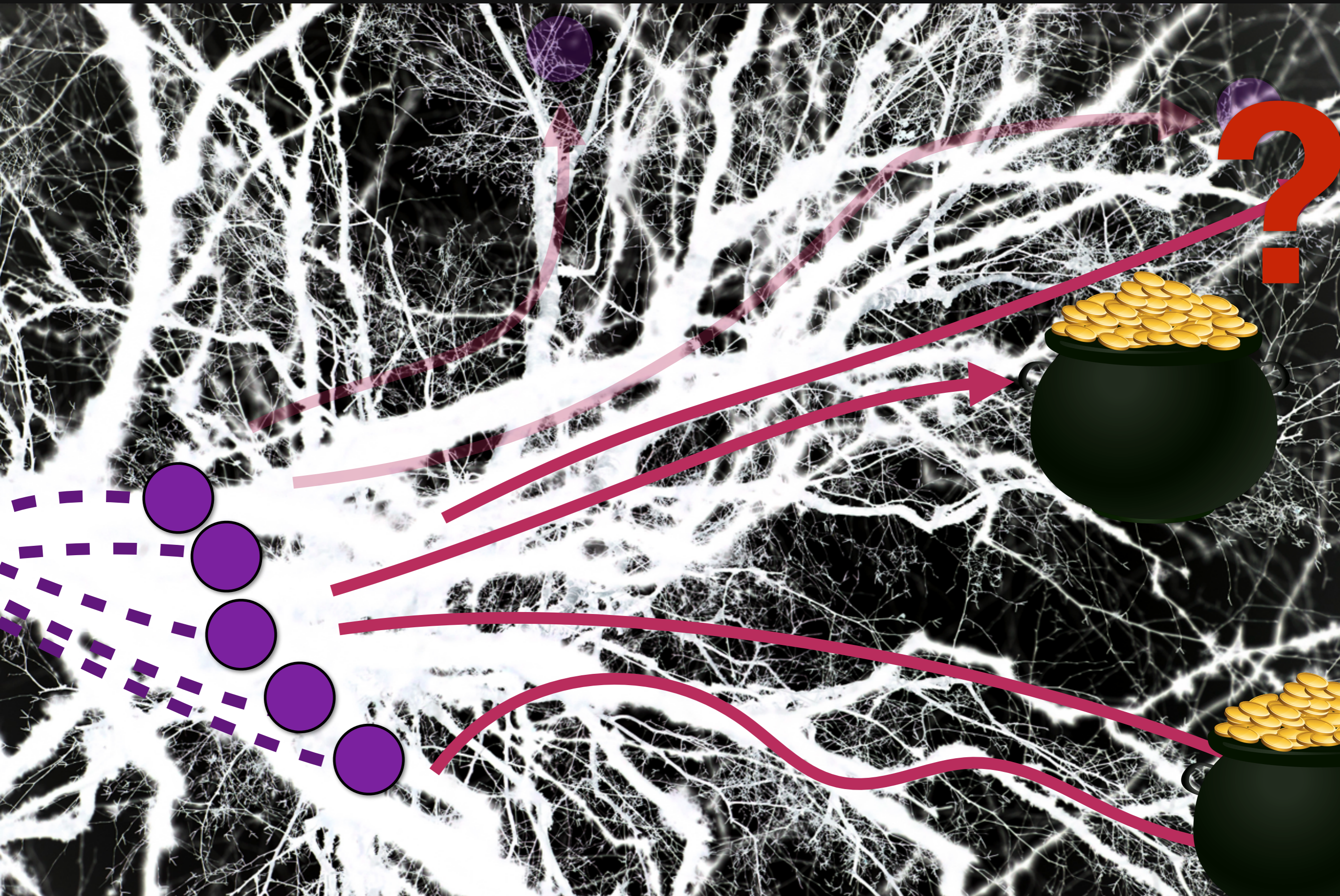
The role of research is...











Almost everything you learn in university
is a product of peer-reviewed, published academic research.

(In CS, also the industry participates in the academic forum.)



You can google ahead for
“rendering equation”, “radiance”

Journey Starts Next Wednesday