CS 563 Advanced Topics in Computer Graphics

by Emmanuel Agu

Outline

- Overview: about me, about class
- What is photorealistic rendering?
- Raytracing introduction

Professor Background

- Dr. Emmanuel Agu (professor, “Emmanuel”)
- Research areas
  - Computer Graphics (photorealistic rendering, etc)
  - Mobile Computing (mobile graphics, cell, iPhone, etc)
  - Wireless networking
- This class: creating computer-generated photorealistic images
    - Ray tracing
    - Humans (face, skin)
    - Nature (water, trees, seashells)
    - Animals (feathers)… etc
- Research opportunities
  - Independent Study Project
  - MQP
  - MS theses
  - PhD theses

Student Background

- Name
- Class (undergrad (seniors), masters, PhD ...)
- Full and Part-time student
- Programming experience (C, C++, java)
- Systems experience (Unix, windows,...)
- Helpful background
  - At least one graphics class taken
  - Solid math skills....
  - Other (Physics, computer vision, image science, ???)
- Students intro themselves!
- Important: fill in above info, say what you want from this class

Course Prerequisites

- No official prerequisite
- However, will assume you
  - Can program in C/C++
  - Have probably taken at least 1 graphics course (OpenGL?), based on raster graphics?
  - You are fearless. can quickly pick up graphics and image processing algorithms and techniques, (lectures will briefly cover them in class as needed)
  - have background in calculus, linear algebra
  - Can read/understand text, research articles, fill in gaps
  - Can program in C, learn rendering package, tools
- Still have questions? See me

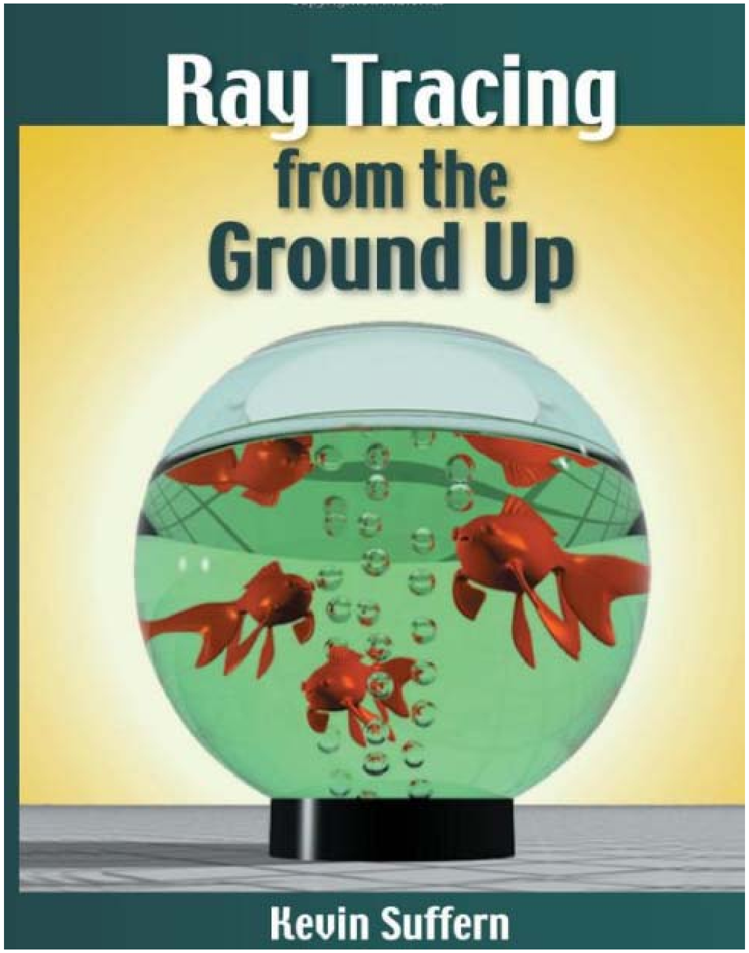
Syllabus

- http://www.cs.wpi.edu/~emmanuel/courses/cs563/S10/
- Office hours:
  - Wednesdays: 4:30 - 5:30
  - Note: Please use office hours or book appointments
- Questions of general interest, post on myWPI
- Email me if you have specific questions
- Text Book: Ray tracing from the ground up ***plus*** selected papers
- Note: Most lectures will be based on the text. But student can supplement from papers, web

Textbook

Ray Tracing from the ground up

by Kevin Suffern



- Author has experience in ray tracing
- Grew out of his classes
- Text Condenses lots of state-of-the art theory + code + explanation of code
- Working code, more concrete

Course Structure

- Grading
  - No exams
  - About 4 presentations each (40%)
  - Class participation: discussions, answer questions (10%)
  - One final rendering project, chosen by you (50%)
  - Students will score other projects at the end. Part of grade determined by your peers

Why This Class?

- WPI graduate course requirements
  - Masters, PhD, grad course requirements
- WPI research requirements
  - Want to do research in graphics (MS, PhD theses)
- Work in graphics
  - Rendering for movie studio, architectural firm, etc
  - Animation, etc.
- Hobbyist
  - Want to build cooler stuff
  - Understand more how visual effects, etc happen

Course Objectives

- Understand state-of-the-art techniques for photorealistic rendering
- Become conversant with cutting edge graphics literature
- Hands-on exploration of one (or more) of the techniques encountered (a project).
- Learning and using raytracing to generate amazing pictures.
- Possibly extend one of the studied techniques, implement new ones

Class Time

- Two halves with 15 minutes break
- Each half
  - 50 minute presentation *followed by*
  - 20 minute discussion of topic(s) and questions

- Commons presentation mistakes
  - Avoid: putting too much on a slide (talk!!)
  - Too many slides for alloted time (2-3 mins/slide)
  - 50 mins: about 20 – 25 slides

- First two student presentations in two weeks time

Presentations

- I will try to guide you on how to present effectively
- I will be strict with time if you go too long
- Get right to the point (core), offer motivation & insights
- Communicate basic ideas to fellow students
- Offer a ‘roadmap’ for studying the paper
- Look over reading list & let me know which topics you want to present
- **Note:** can use additional resources to build your talk. Must give credit. If not.. **Cheating**!!!
- Don’t just summarize! Find authors websites, videos, images, supplementary cool stuff

Final Project

- Implement one of the rendering techniques discussed in class, use ray tracer from text
- May also use high end package to create models
  - Maya
  - Renderman
  - Blender
  - PovRay, etc
- Must submit your final project proposal by March 31st, 2010
- Can get ambitious: Implement new photorealistic technique from a paper
- Ideas?? See Stanford rendering competition
- *http://graphics.stanford.edu/courses/cs348b-competition/*

Class resources

- Where to do the projects:
  - On your home computer, download ray tracer
  - On campus computer labs
- Class text
- Supplementary books:
  - Physically-based rendering by Pharr and Humphreys
  - Computer graphics using OpenGL by F.S. Hill and Kelley, 3rd edition, Chapter 12
  - Other books I place on reserve in CS 563 folder in library

Outline

- Overview: about me, about class
- What is photorealistic rendering?
- Raytracing introduction

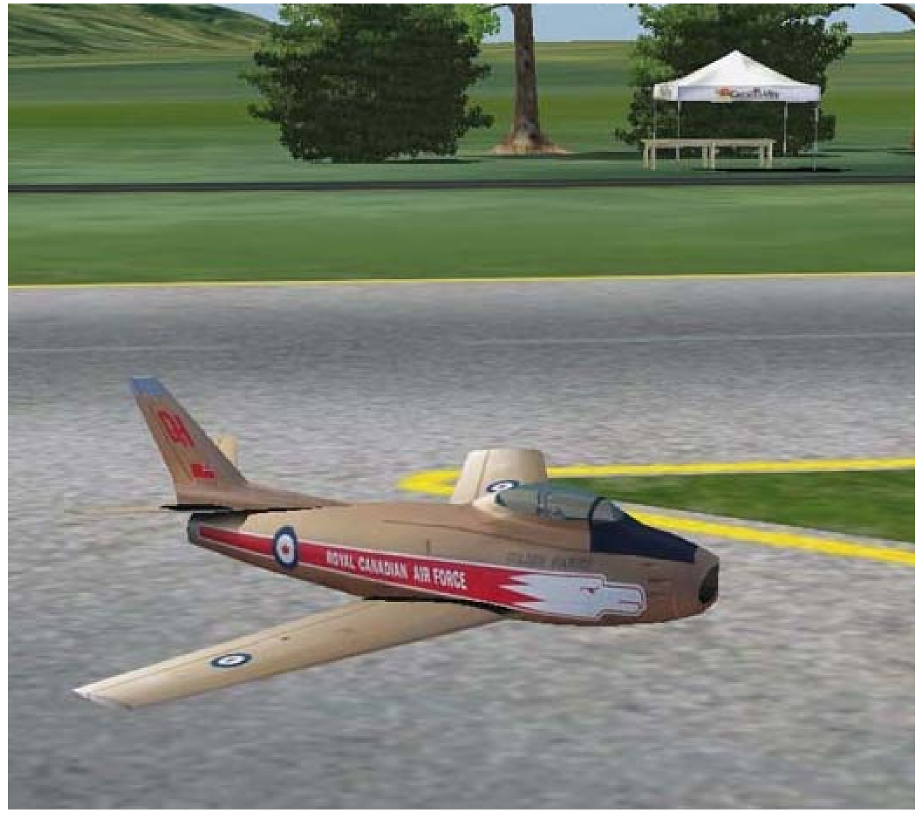
What is Rendering?

- Create a 2D picture of a 3D world
- Photorealistic: Indistinguishable from photo

Applications

- Movies
- Industrial design
- Architecture
- Demo products
- Virtual reality (games)

Photorealistic vs real-time rendering



- **Photorealistic rendering (E.g. Ray tracing)**
- Can take days to render
- Used in: movies, adverts

- **Raster graphics:**
  (E.g. OpenGL, DirectX)
  fast: milliseconds to render
  poor image quality

Used in: games, simulators

Photorealistic Rendering

- Ingredients: Require good models for
  - **Geometry** (Realistic shapes, meshes)
  - **Light source** (sky, light bulb, flourescent)
  - **Volume** through which light travels (smoke, fog, mist, water)
  - **Materials:** Reflection/refraction at object surfaces (velvet, wood, polished, rough, smooth)
  - **Cameras:** Lens and film
- Old approach: Fudge it! (E.g. Phong's shading)
- New approach:
  - study light physics
  - derive models, adapt equations from physics papers
  - Use physically-based models for rendering
  - **Capture:** Place cameras/equipment around real objects/phenomena and collect data
  - **Measure:** phenomena

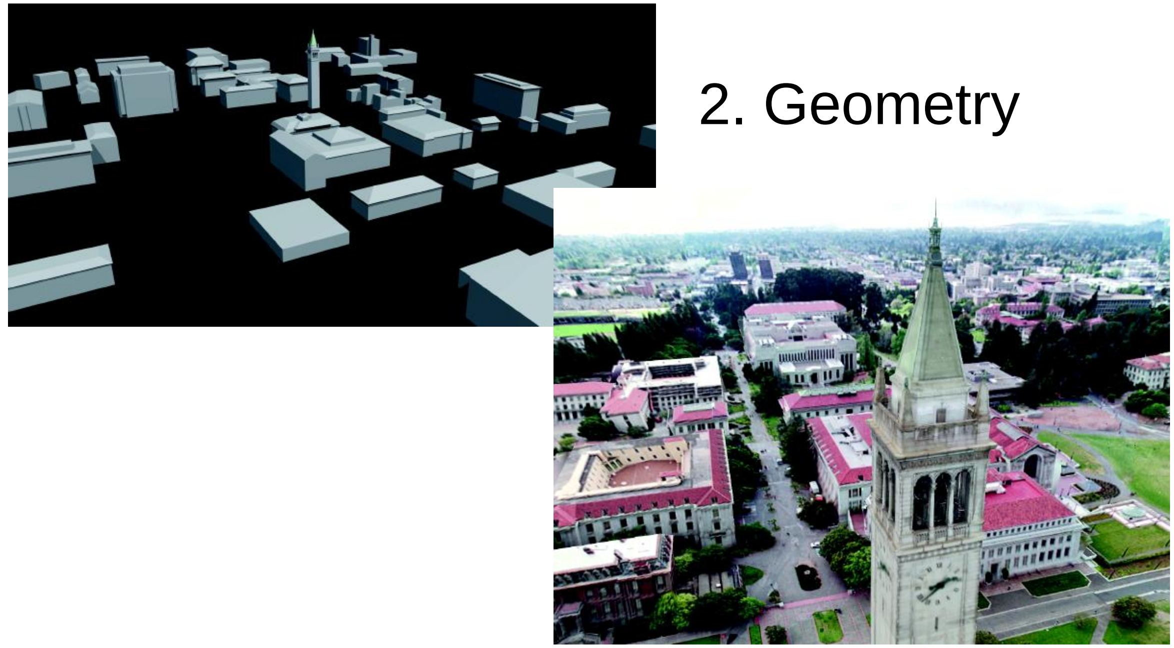
Physically-based rendering

uses physics to simulate the interaction between matter and light, realism is primary goal

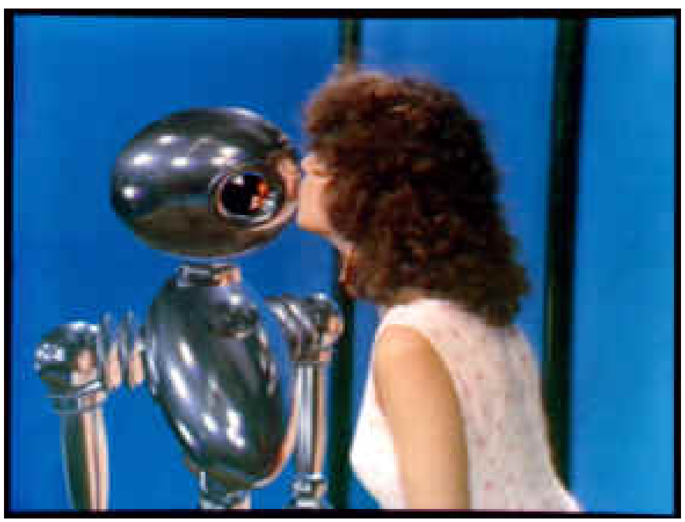
Exactly What Can We Capture?

1. Appearance

2. Geometry



3. Reflectance & Illumination



4. Motion

Scanning 3D geometry

- Quest for greater realism: Trend in Computer Graphics towards very large polygonal models
  - Projects on precise 3D scanning (Stanford, IBM,etc)

Model: David, **2 billion polygons**

Courtesy: Stanford Michael Angelo 3D scanning project

What can we model?

Physically-based Appearance Models

- Why?
  - Sky **appears** blue?
  - Wet sand **appears** darker than dry sand?
  - Iridescent surfaces (CD-ROM, butterflies, hummingbird) wings) **appear** to have different colors when viewed in different directions ?
  - Old and weathered surfaces **appear** different from new ones?
  - Rusted surfaces **appear** different from un-rusted ones?
- **Appearance models** in computer graphics and vision try to answer these questions
  - Using physics-based appearance models to render:
    - Humans (face, skin)
    - Nature (water, trees, seashells)
    - Animals (feathers, butterflies)

History: Geometric Aspects First

- Transformation/clipping and the graphics pipeline
  - Evans and Sutherland
- Hidden line and surface algorithms
  - Sutherland, Sproull, Shumacker

History: Simple Shading

- Simple shading and texturing
  - Gouraud $\Rightarrow$ interpolating colors
  - Phong $\Rightarrow$ interpolating normals
  - Blinn, Catmull, Williams $\Rightarrow$ texturing

History: Optical Aspects Second

- Reflection and texture models
  - Cook and Torrance $\Rightarrow$ BRDF
  - Perlin $\Rightarrow$ Procedural textures
  - Cook, Perlin $\Rightarrow$ Shading languages
- Illumination algorithms
  - Whitted $\Rightarrow$ Ray tracing
  - Cohen, Goral, Wallace, Greenberg, Torrance Nishita, Nakamae $\Rightarrow$ Radiosity
  - Kajiya $\Rightarrow$ Rendering equation

Lighting

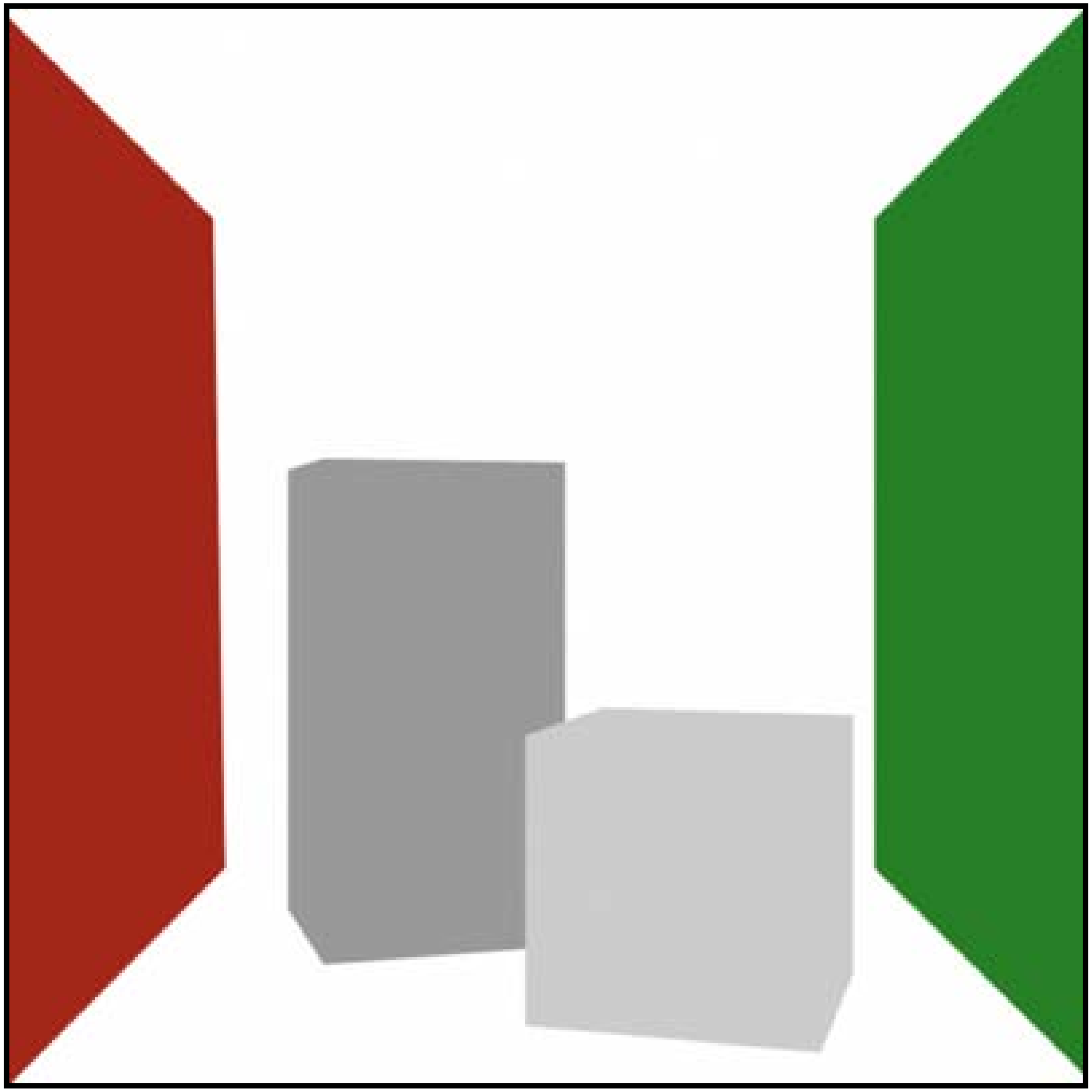
Lighting Simulation

- The Rendering Equation

  Given a scene consisting of geometric primitives with material properties and a set of light sources, compute the illumination at each point on each surface

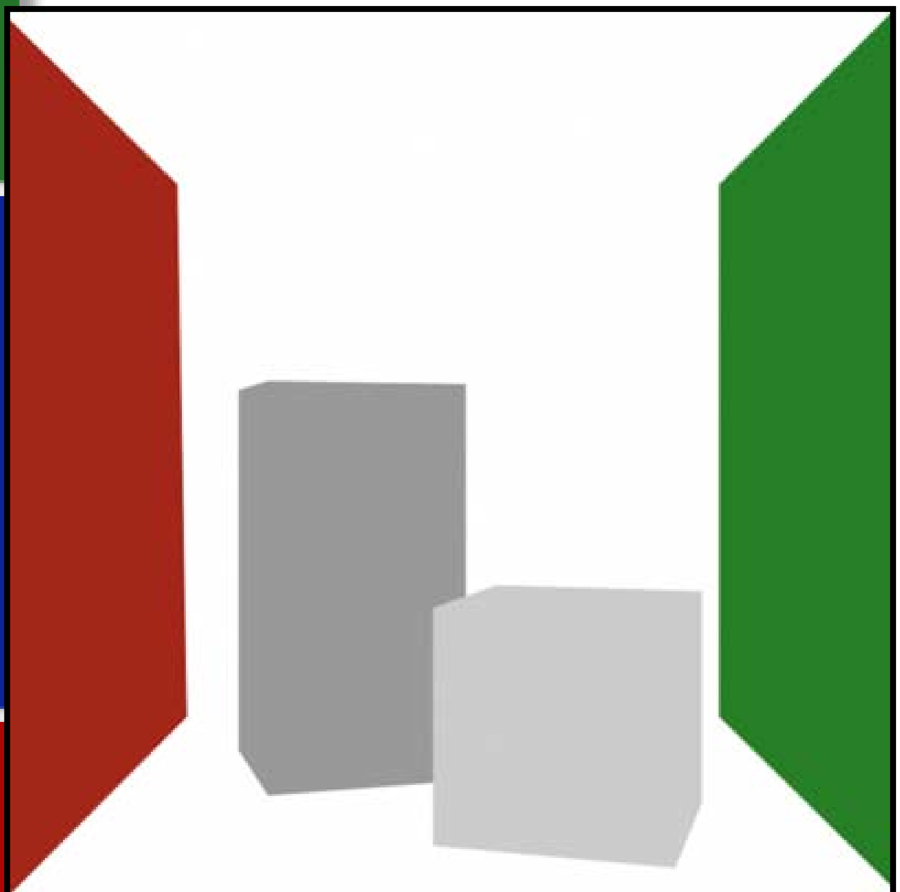
- Challenges
  - Primitives complex: lights, materials, shapes
  - Infinite number of light paths
- How to solve it?
  - Radiosity ⇒ Finite element
  - Ray tracing ⇒ Monte Carlo

Lighting Example: Cornell Box

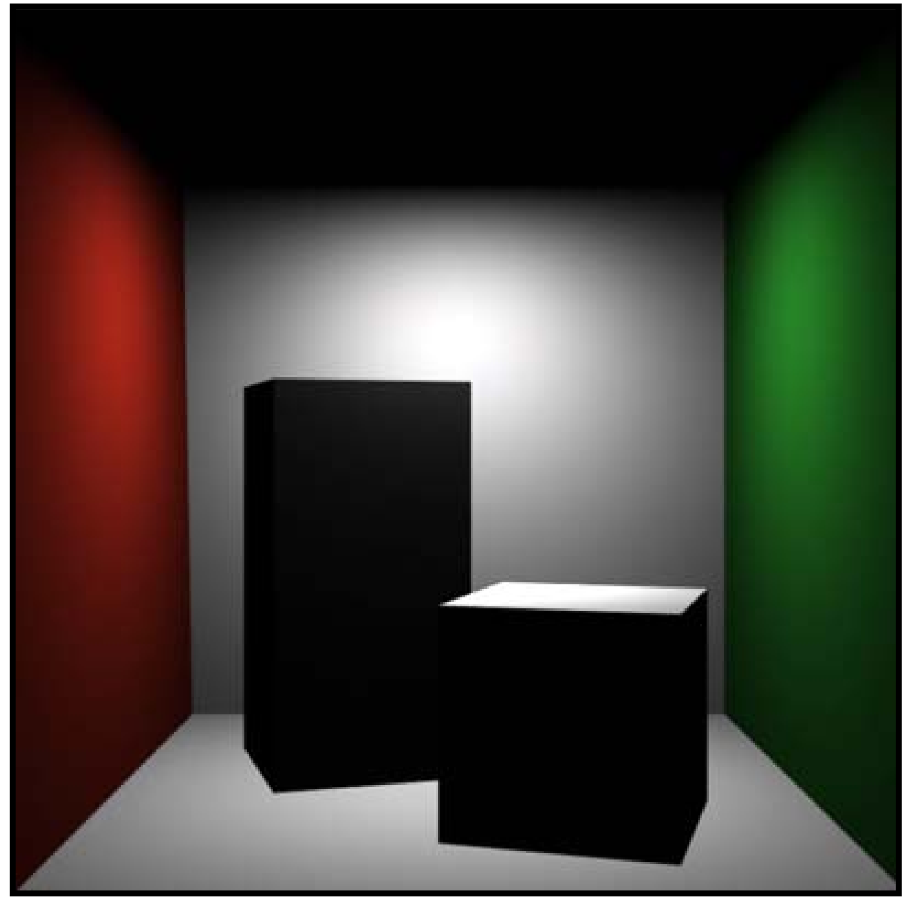


Surface Color

Lighting Example: Diffuse Reflection

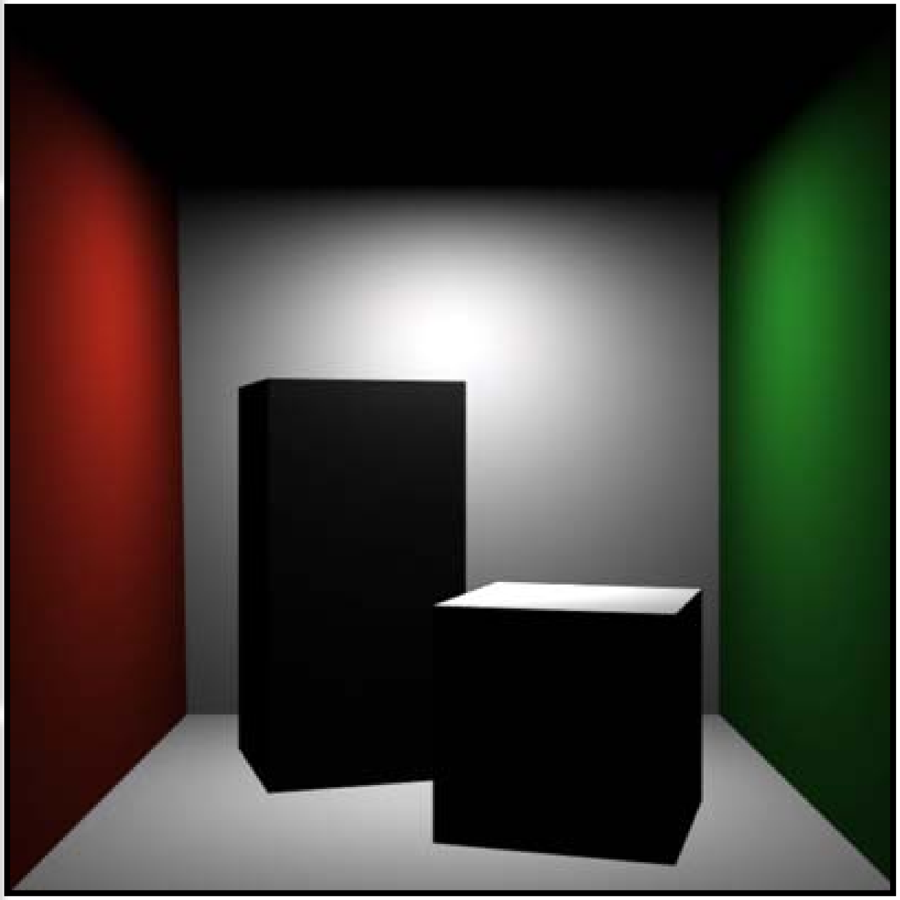


Surface Color

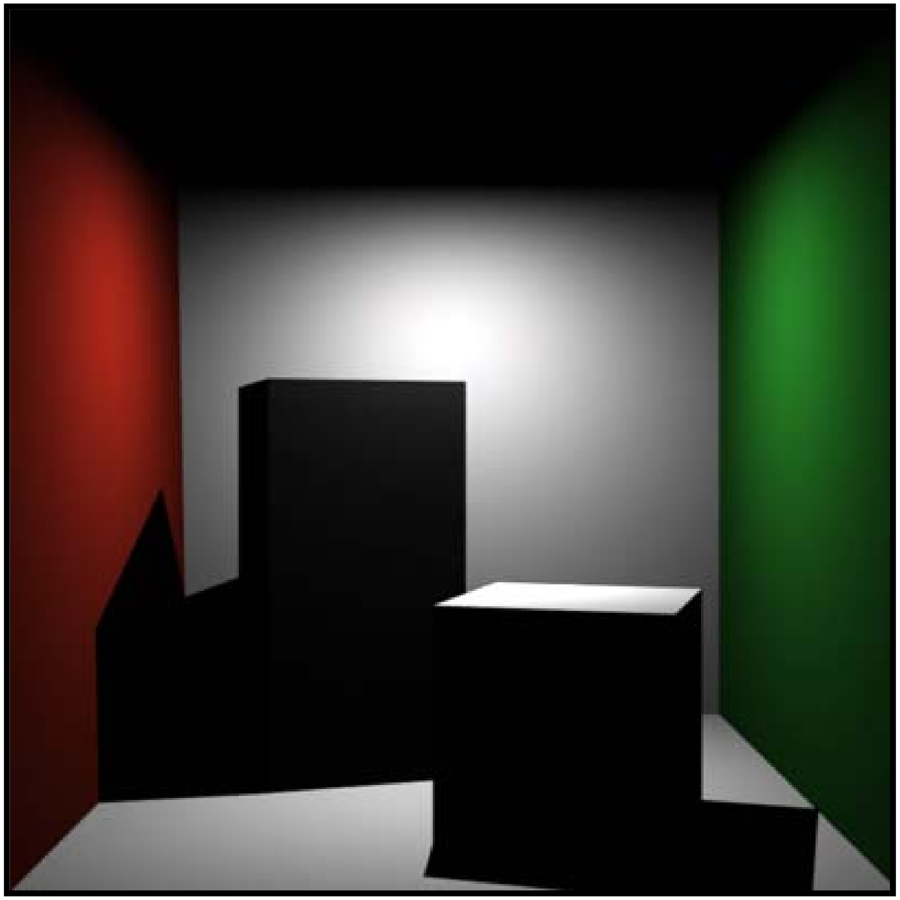


Diffuse Shading

Lighting Example: Shadows

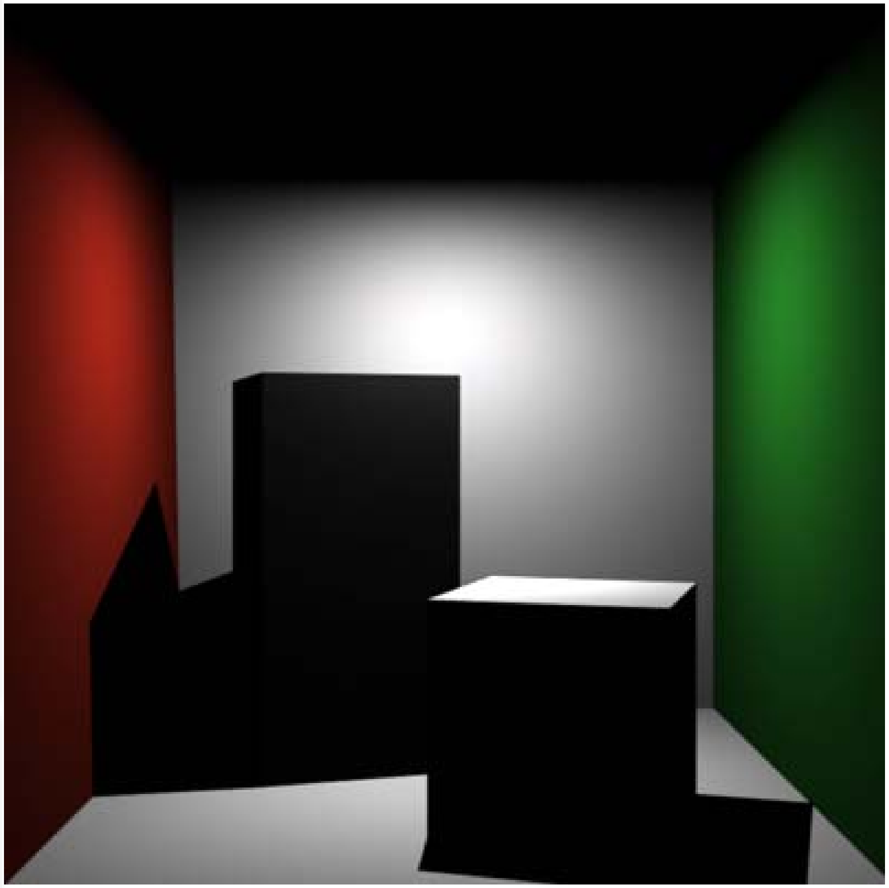


No Shadows

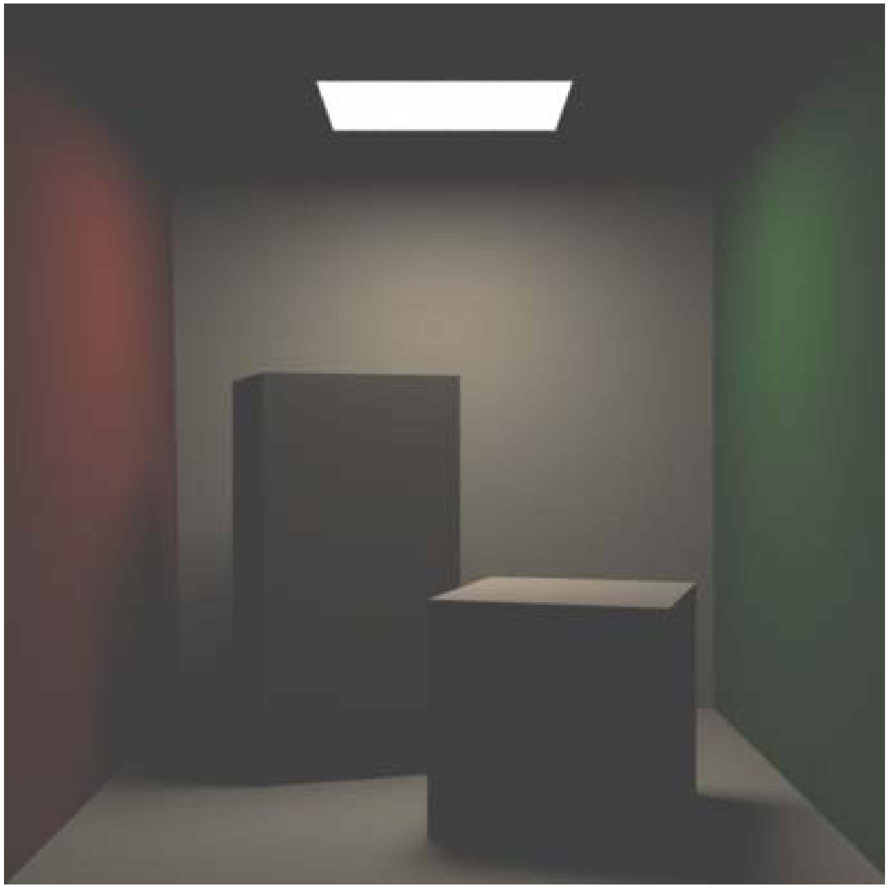


Shadows

Lighting Example: Soft Shadows

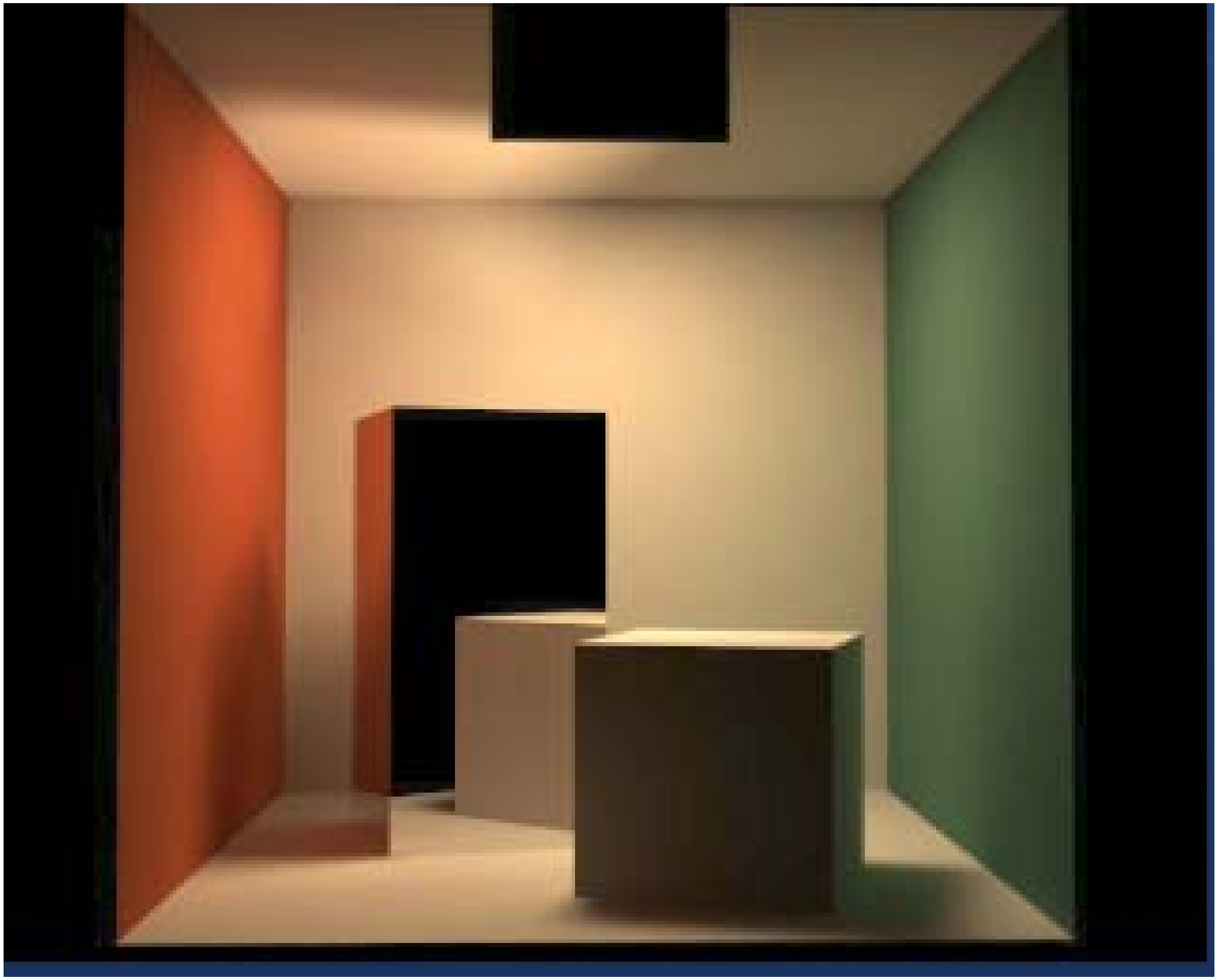


Hard Shadows
Point Light Source



Soft Shadows
Area Light Source

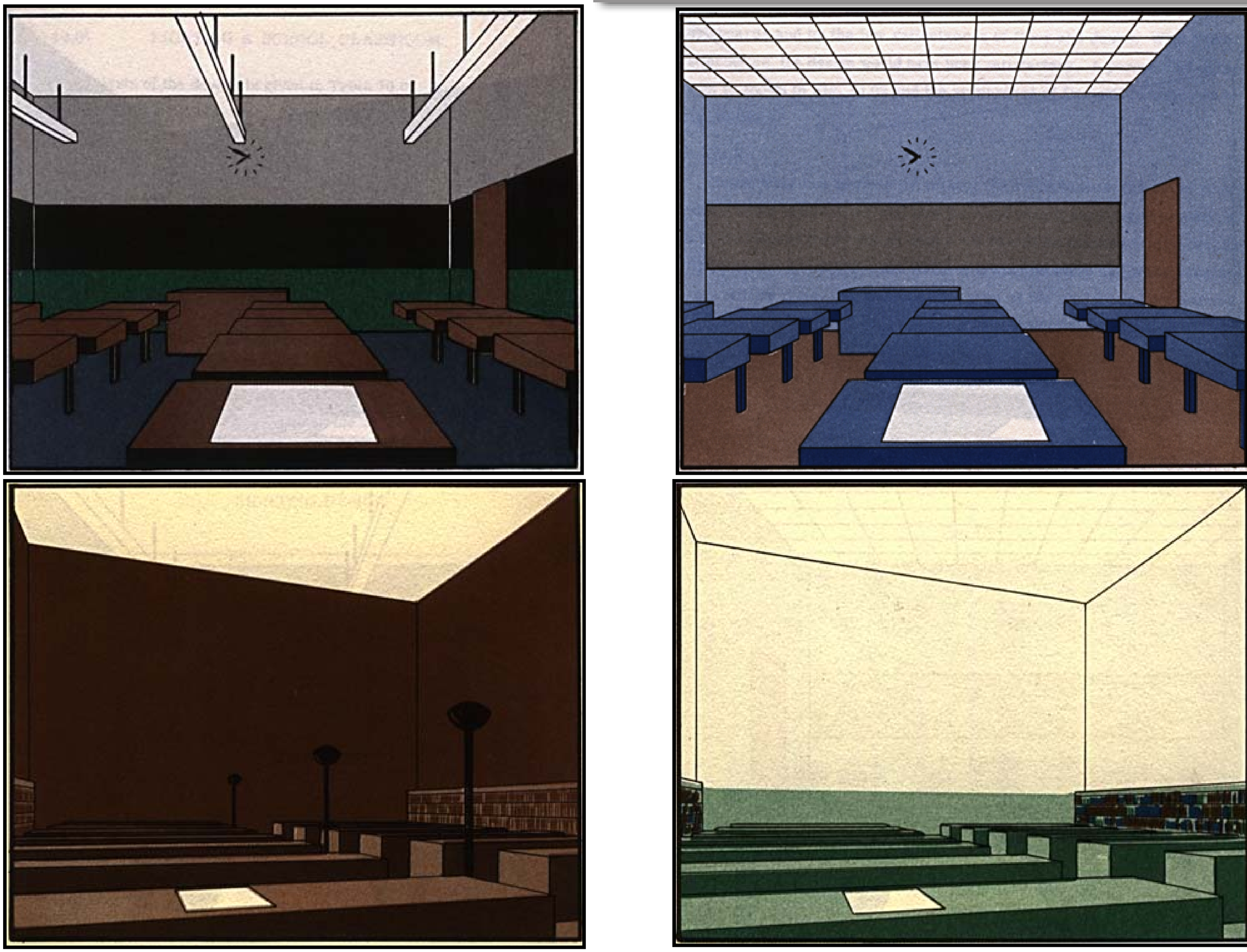
Radiosity: Indirect Illumination



Program of Computer Graphics
Cornell University

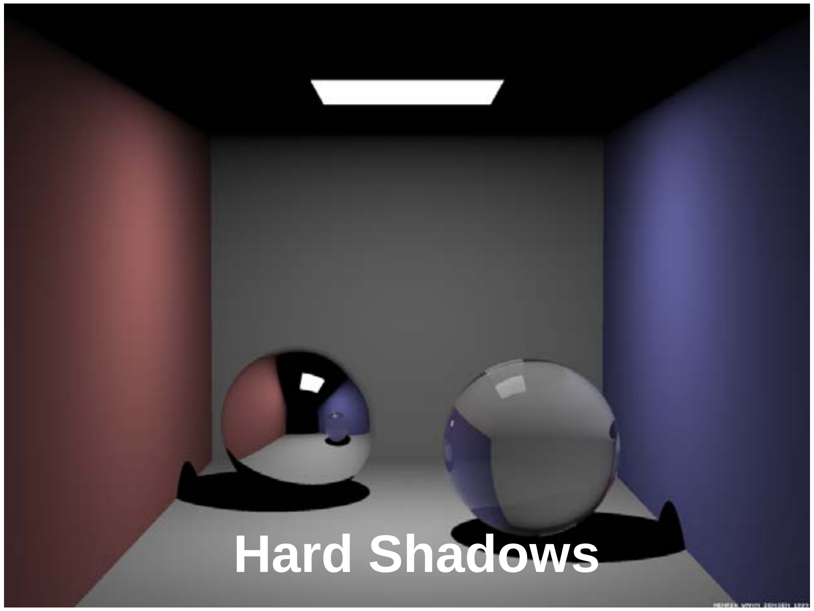
Early Radiosity

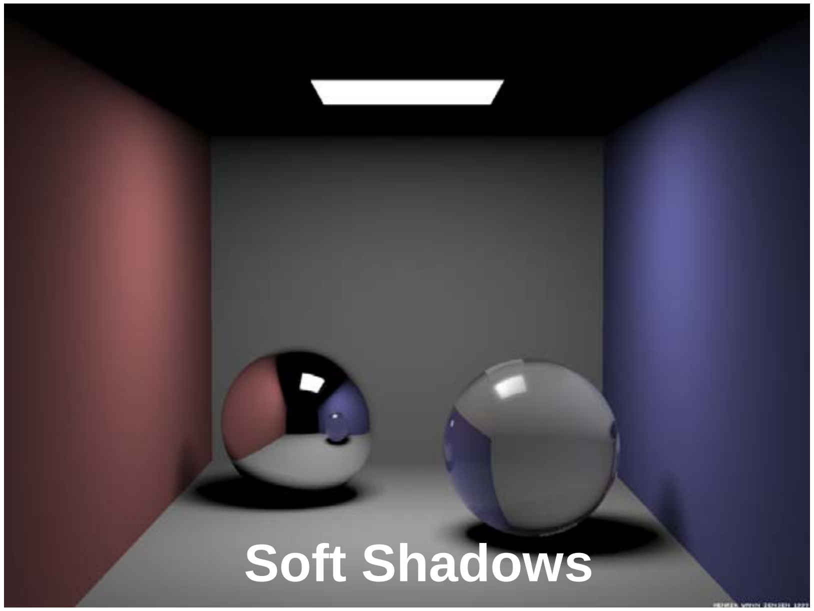
Early, Early Radiosity

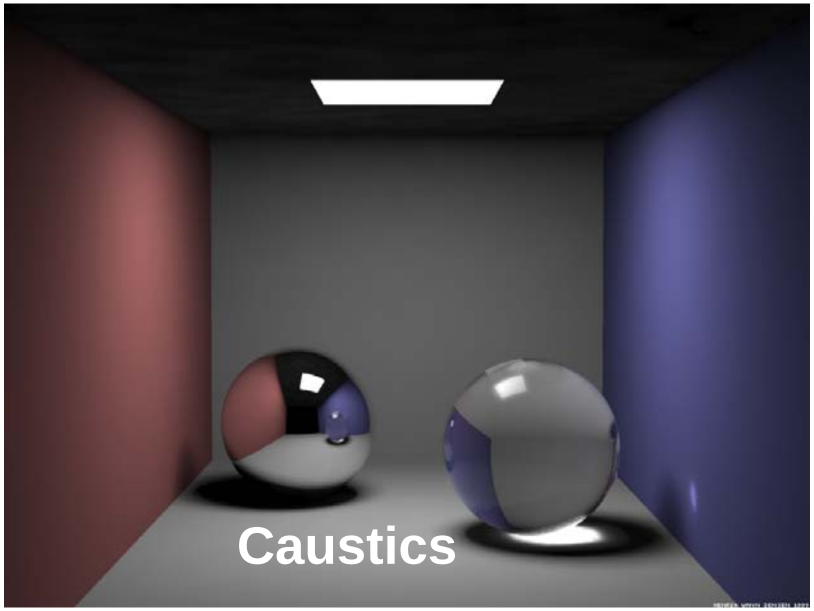


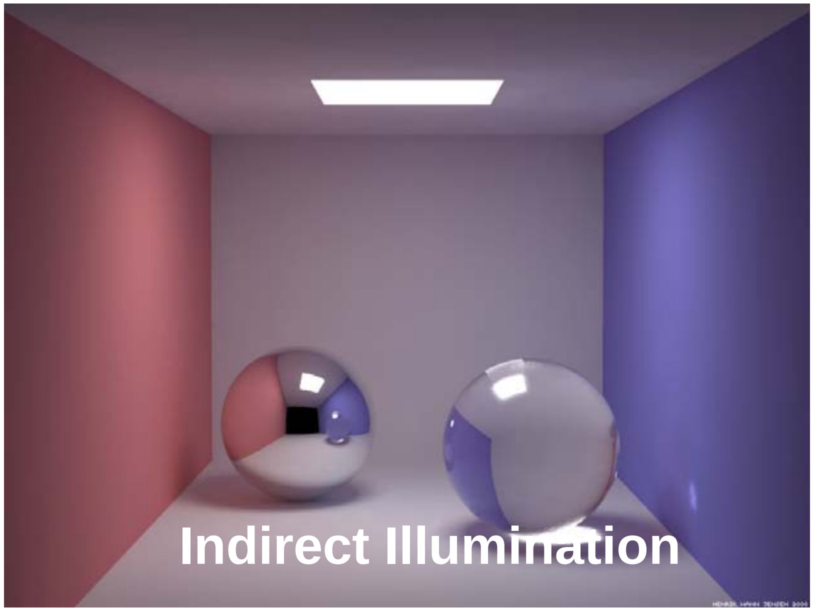
Parry Moon and Domina Spencer (MIT), Lighting Design, 1948

Lighting Effects: Glossy Materials

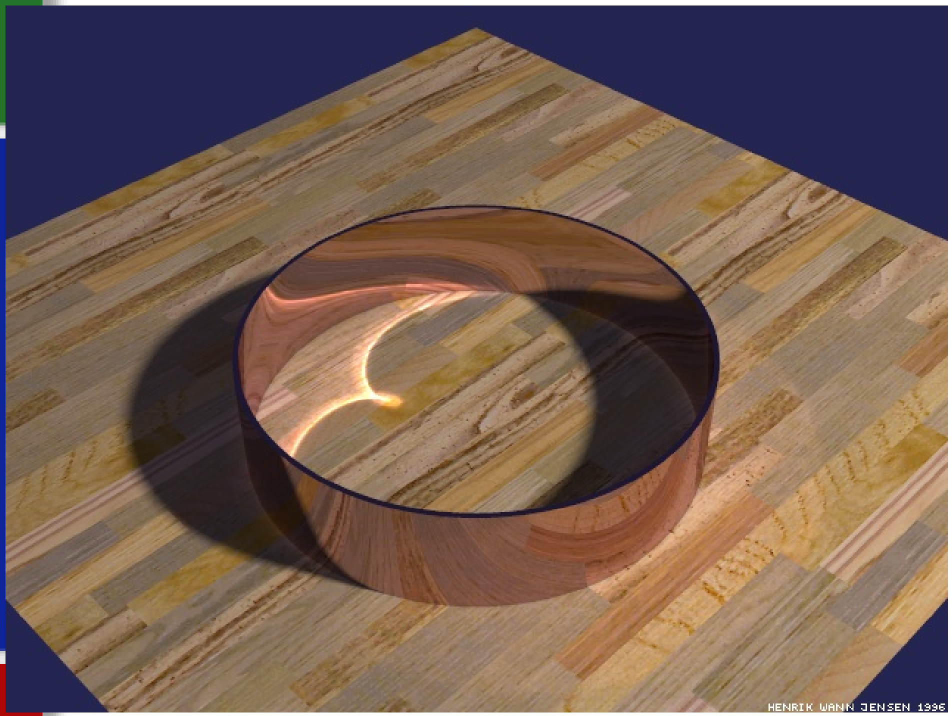








Caustics



Jensen 1995

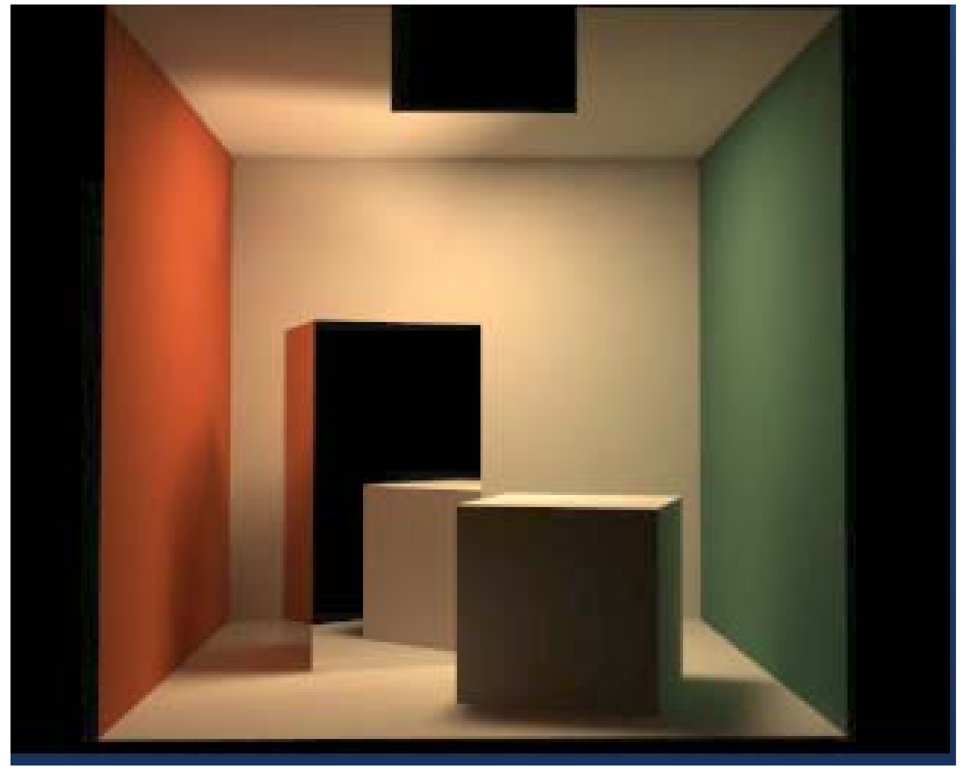
Complex lighting

Complex Indirect Illumination

Modeling: Stephen Duck; Rendering: Henrik Wann Jensen

Radiosity: “Turing Test”

Measured

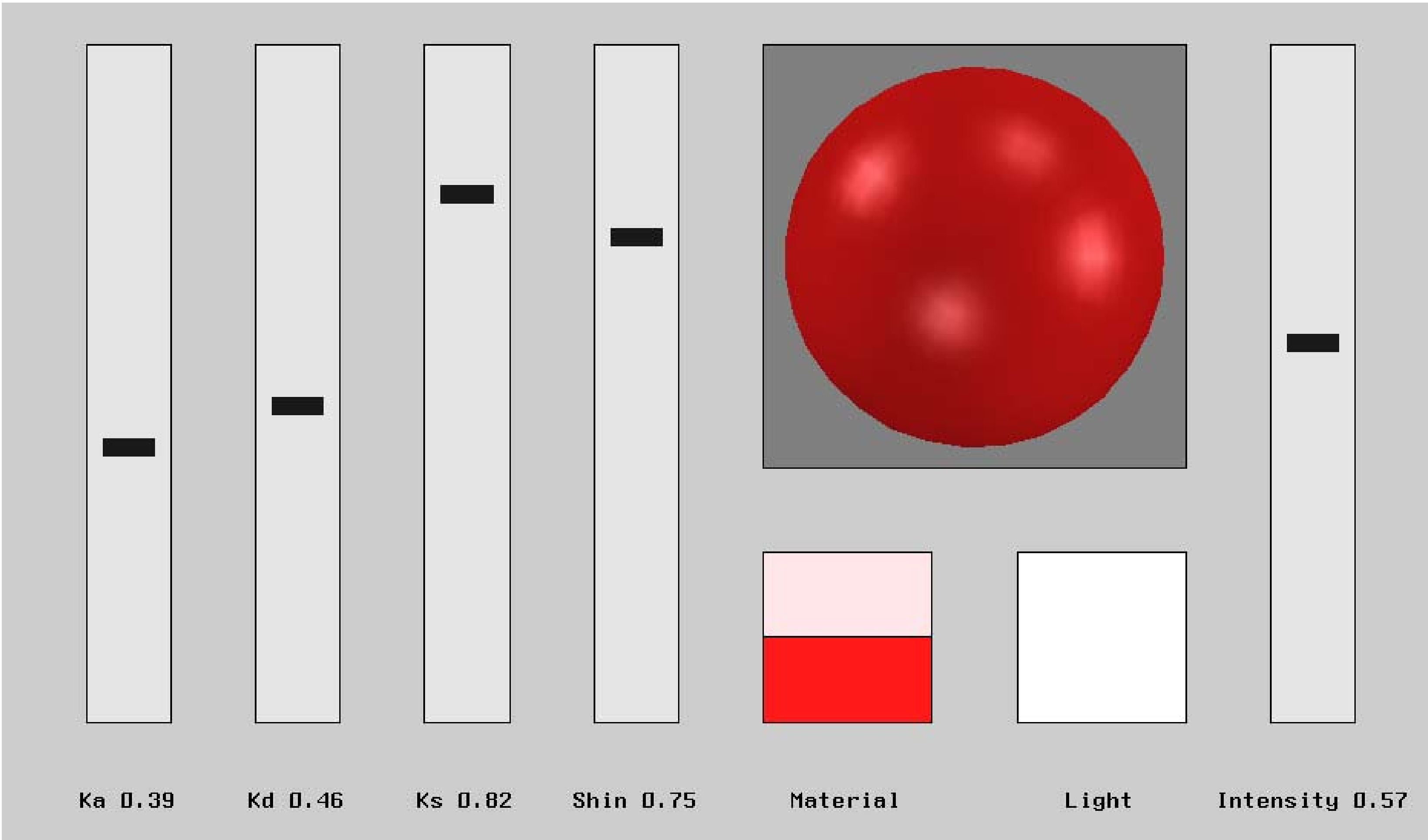


Simulated

Program of Computer Graphics
Cornell University

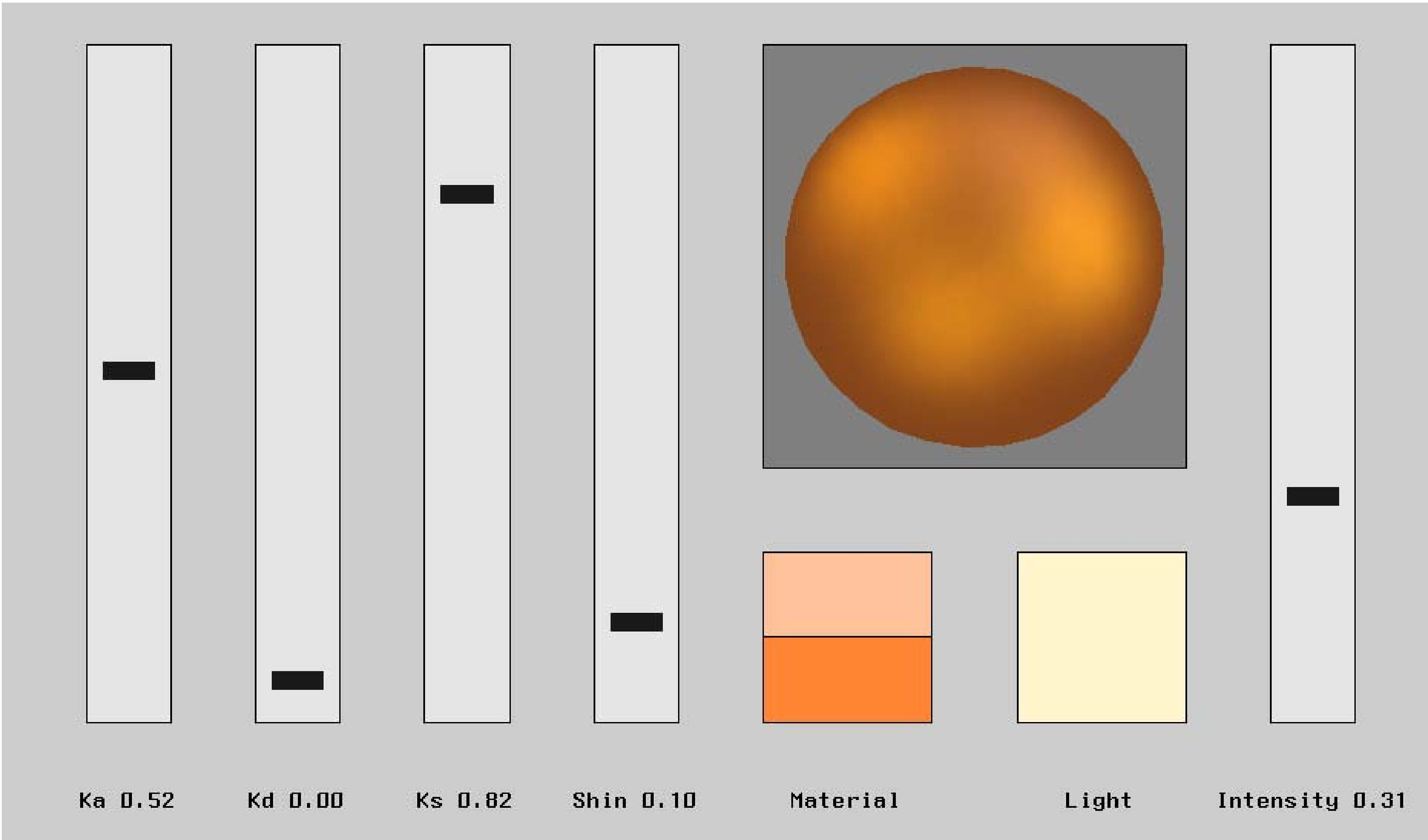
Materials

Classic Computer Graphics Model



Plastic

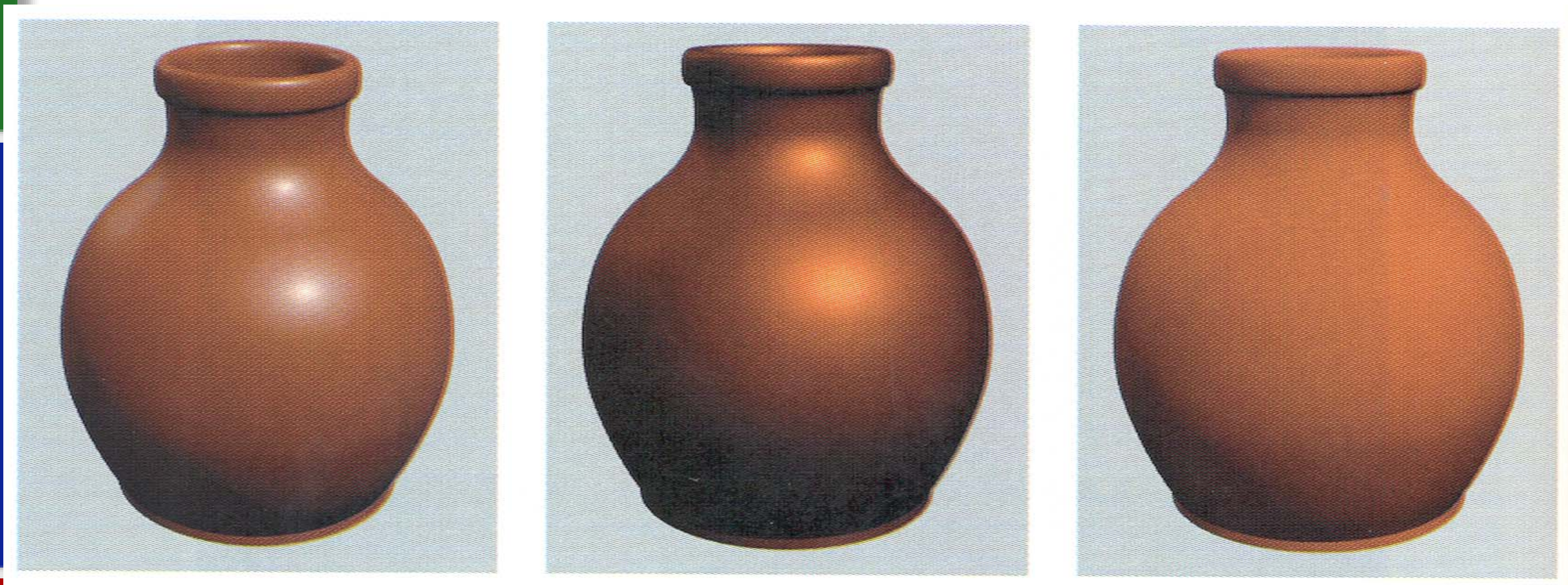
Classic Computer Graphics Model



Brushed Copper

Material Taxonomy

RenderMan



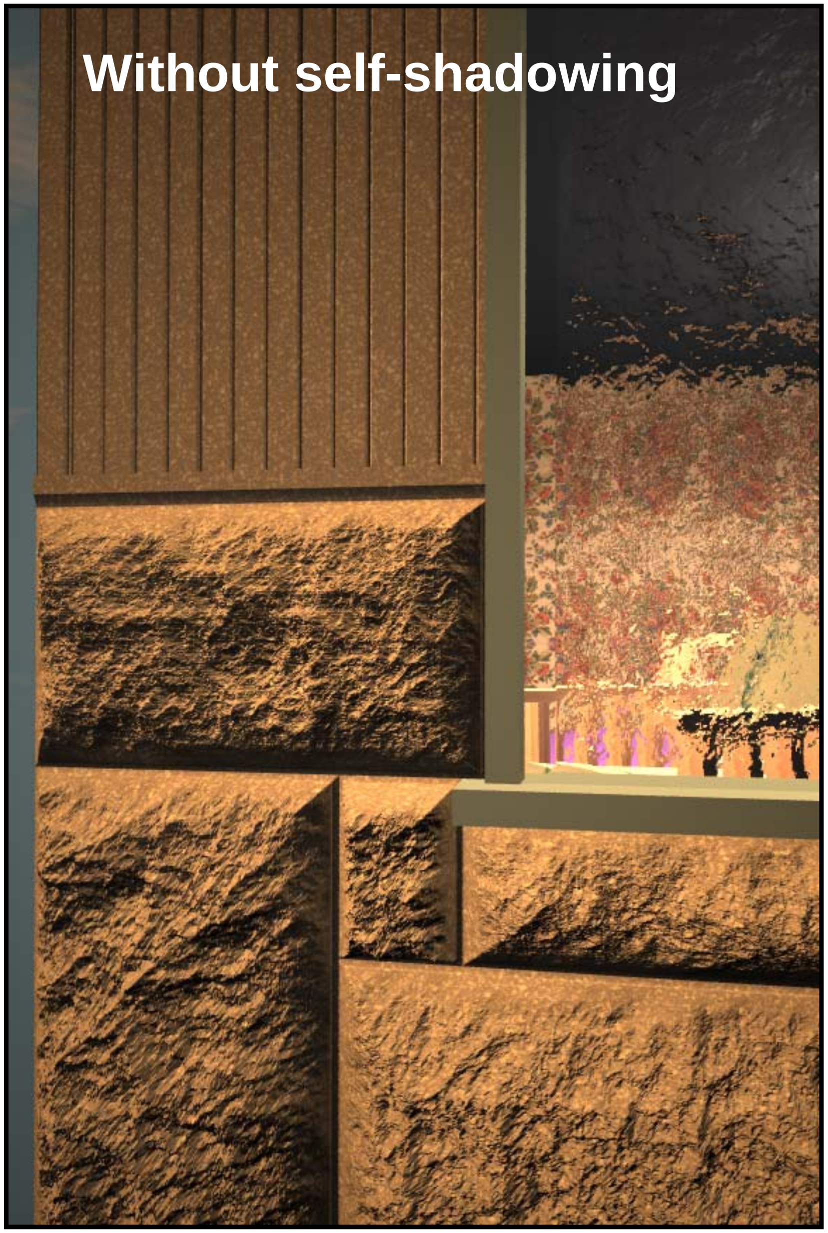
Plastic
Shiny Plastic

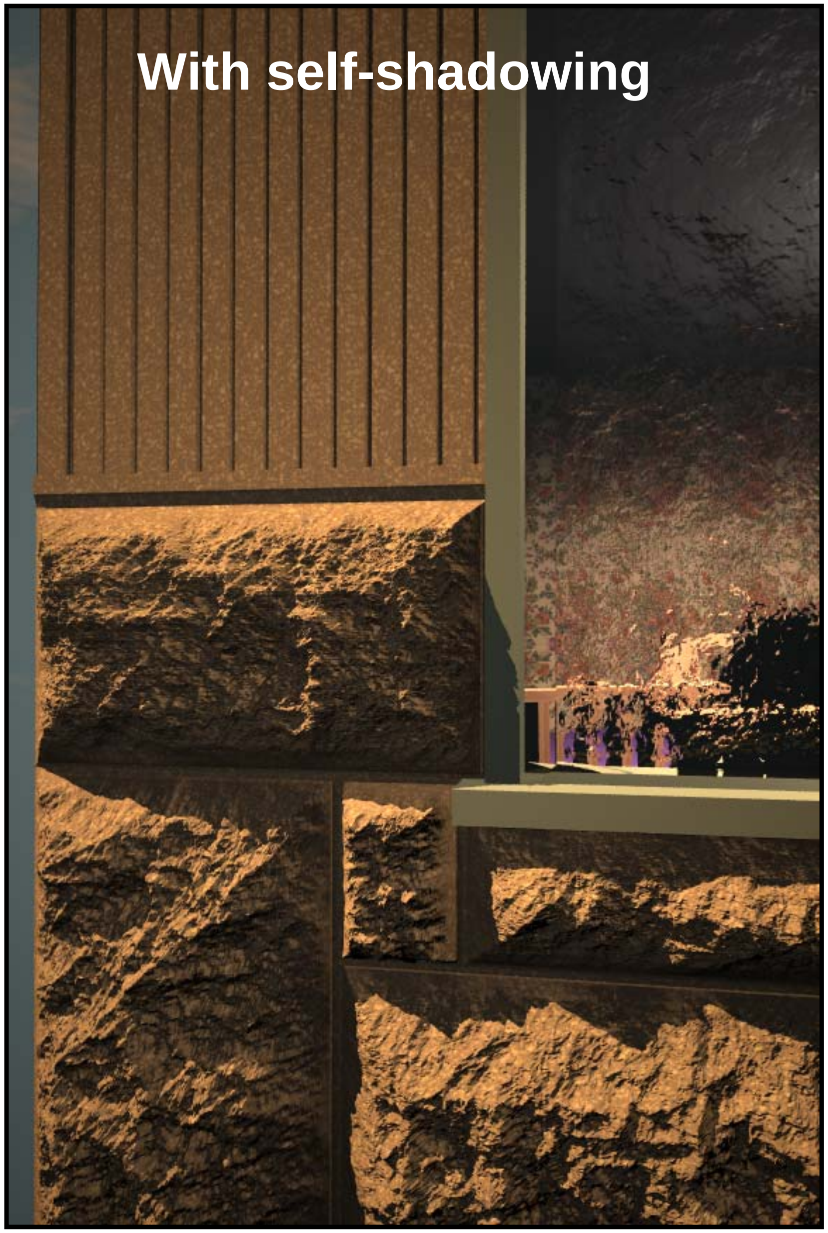
Rough Metal
Shiny Metal

Matte

From Apodaca and Gritz, *Advanced RenderMan*

Shadows on Rough Surfaces





Translucency

Surface Reflection

Subsurface Reflection

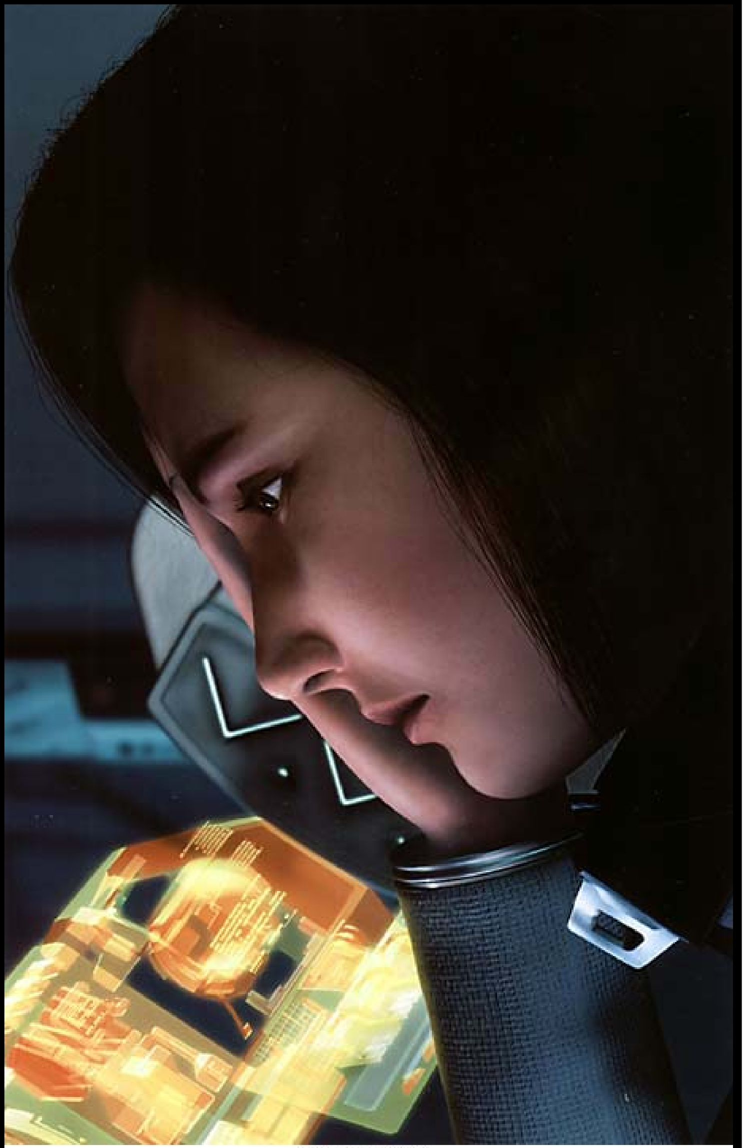
Translucent objects

Water Flows on the Venus

Patinas

A Sense of TIme

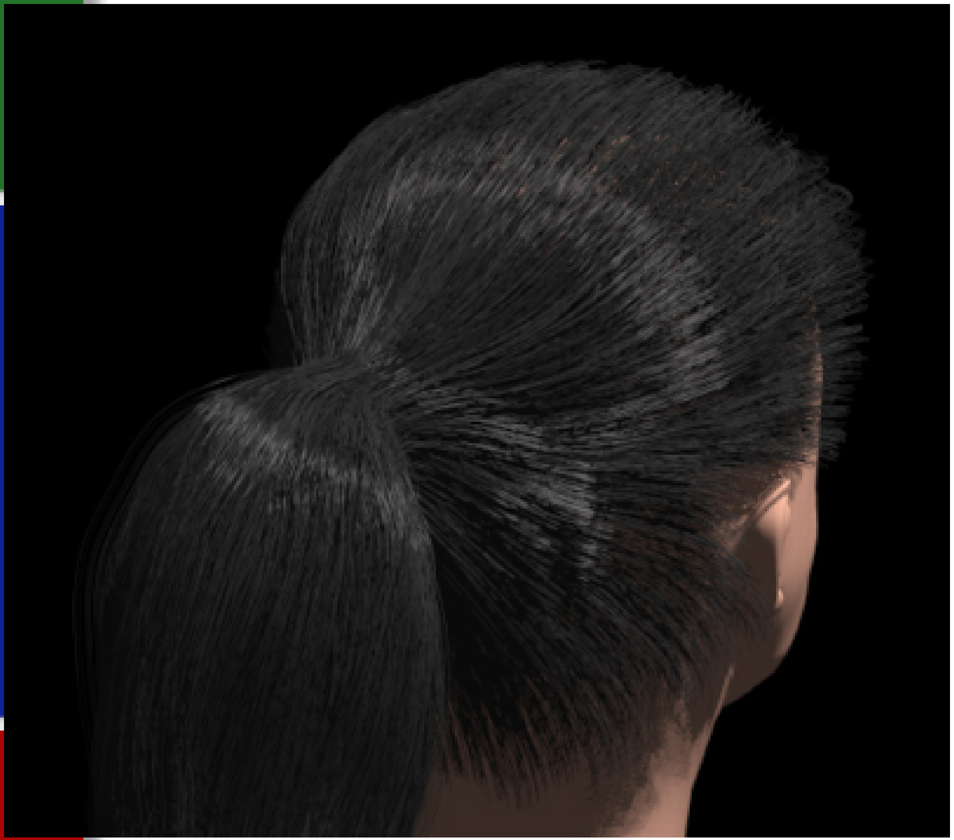
Virtual Actors: Faces


Square USA
The digital heroine of the Final Fantasy film.

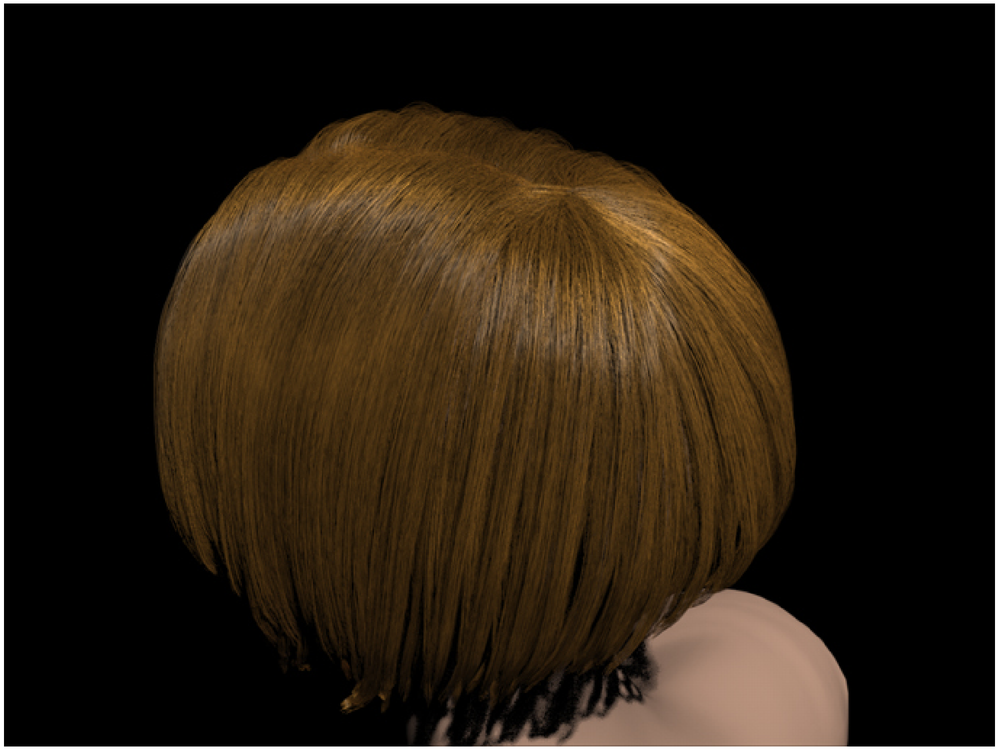
Final Fantasy
SquareUSA

Jensen,
Marschner,
Levoy,
Hanrahan

Virtual Actors: Hair



Black

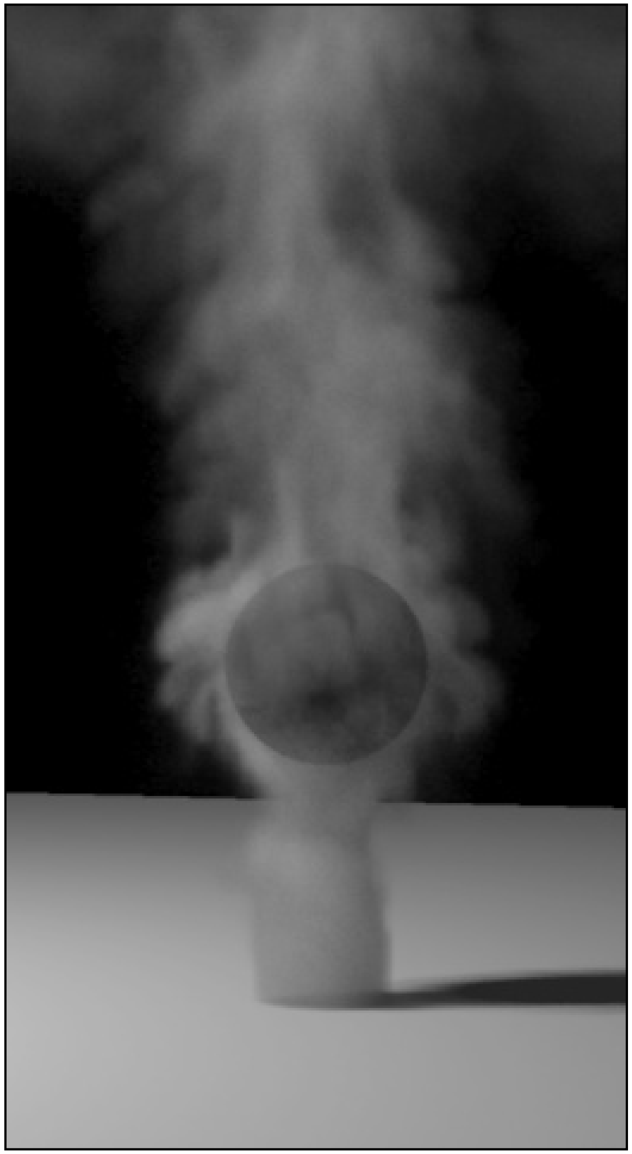


Brown

Refraction / dispersion

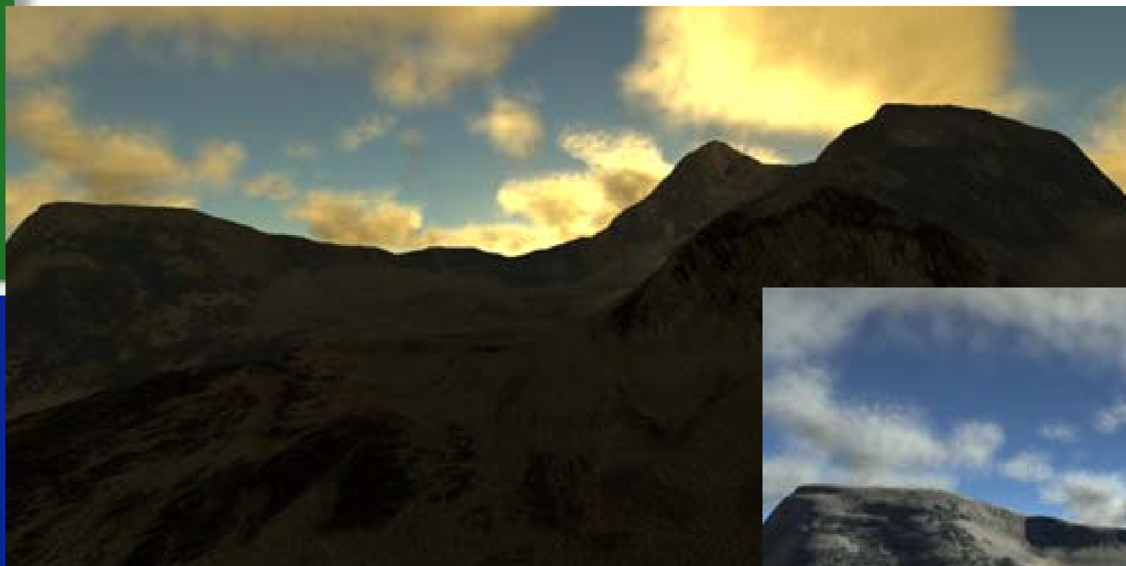
- Iridescent: Wavelength-dependent phenomena

Coupling Modeling & Rendering



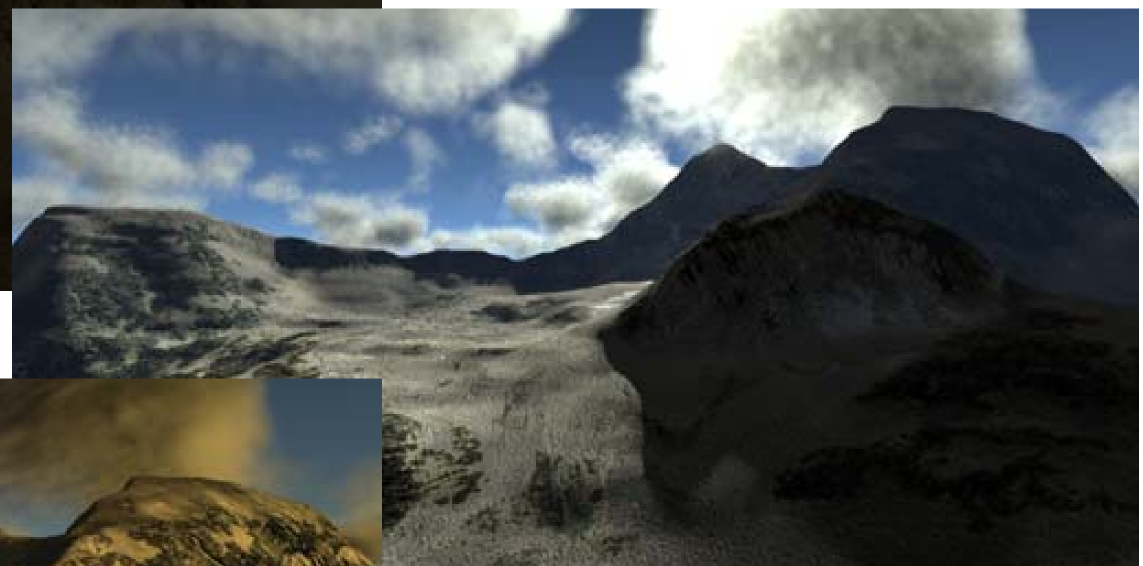
Fedkiw, Stam, Jensen 2001

Clouds and Atmospheric Phenomena

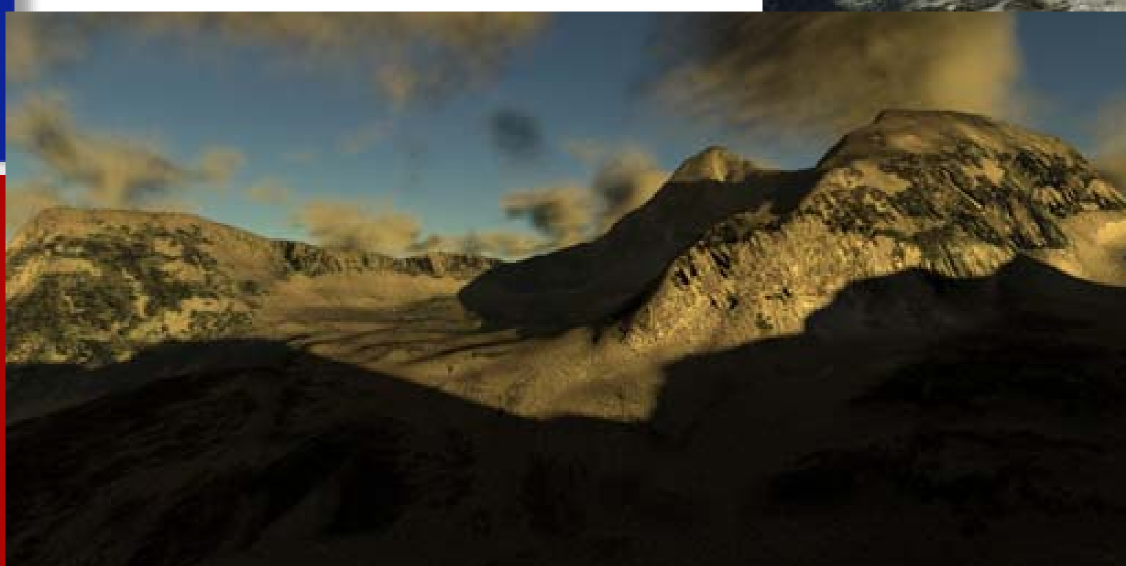


7am

Hogum Mountain
Sunrise and sunset



9am



6:30pm

Modeling:
Simon Premoze
William Thompson

Rendering:
Henrik Wann Jensen

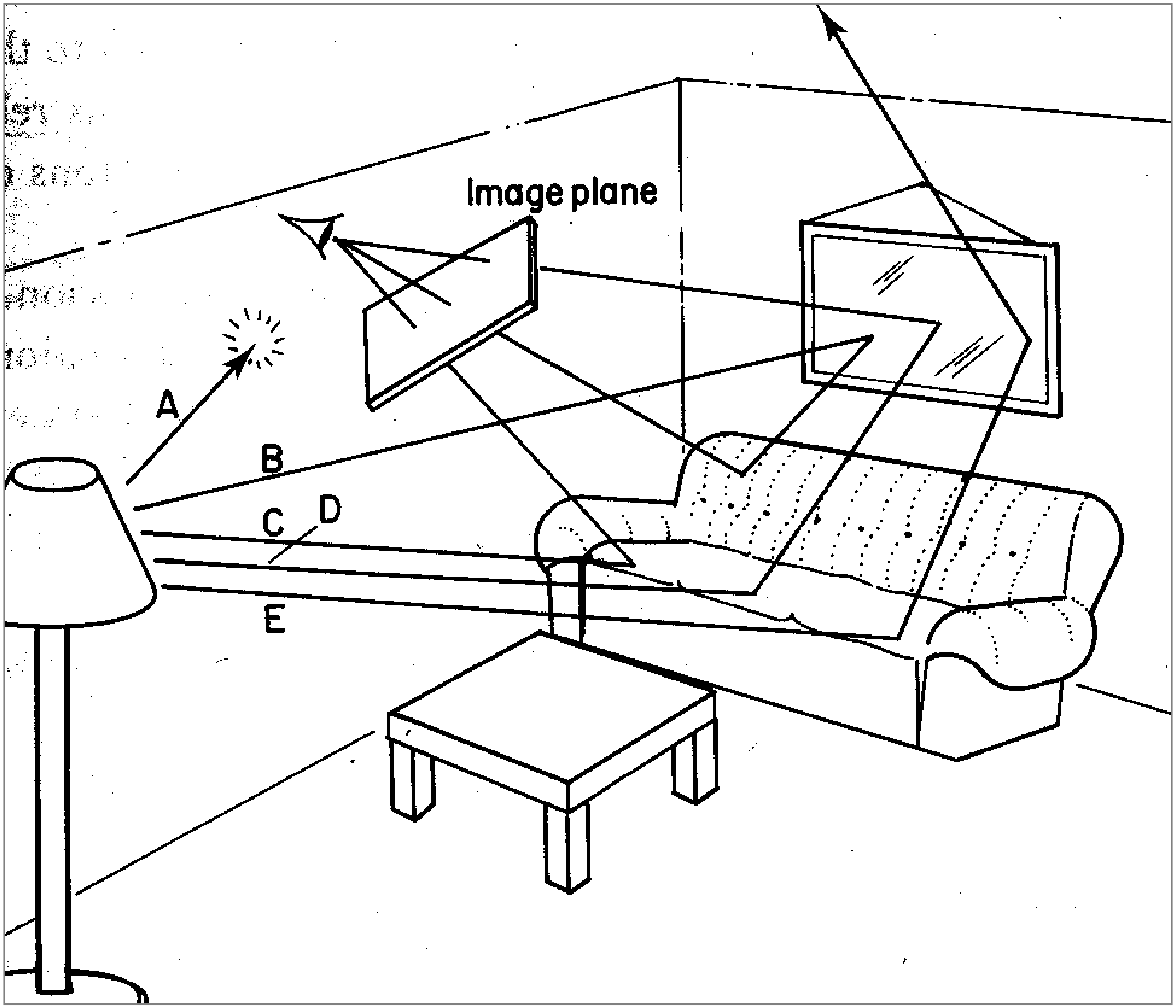
Vegetation

Texture and complex materials

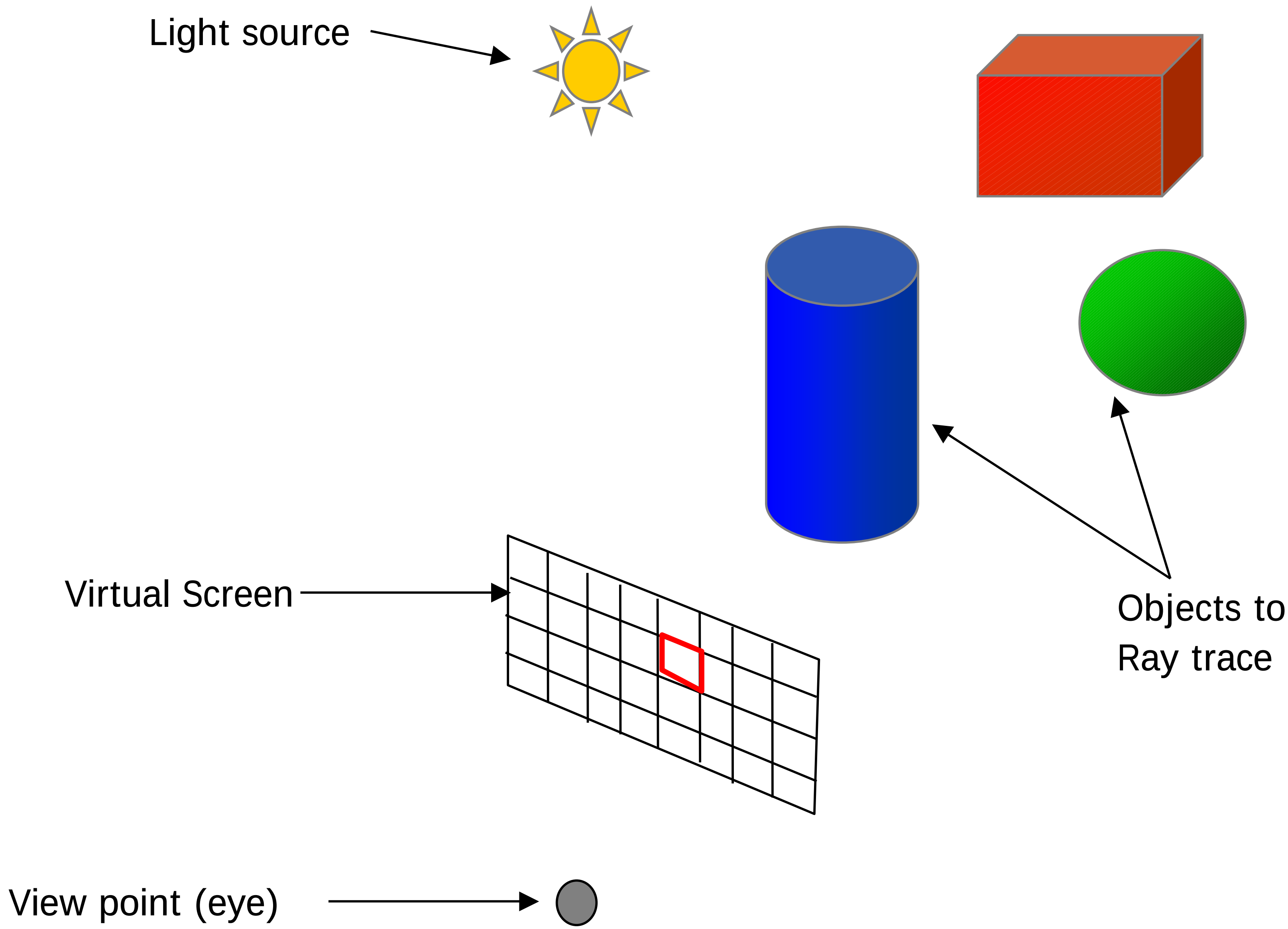
Outline

- Overview: about me, about class
- What is photorealistic rendering?
- Introduction to Raytracing

Introduction to ray tracing



Ray Casting (Appel, 1968)

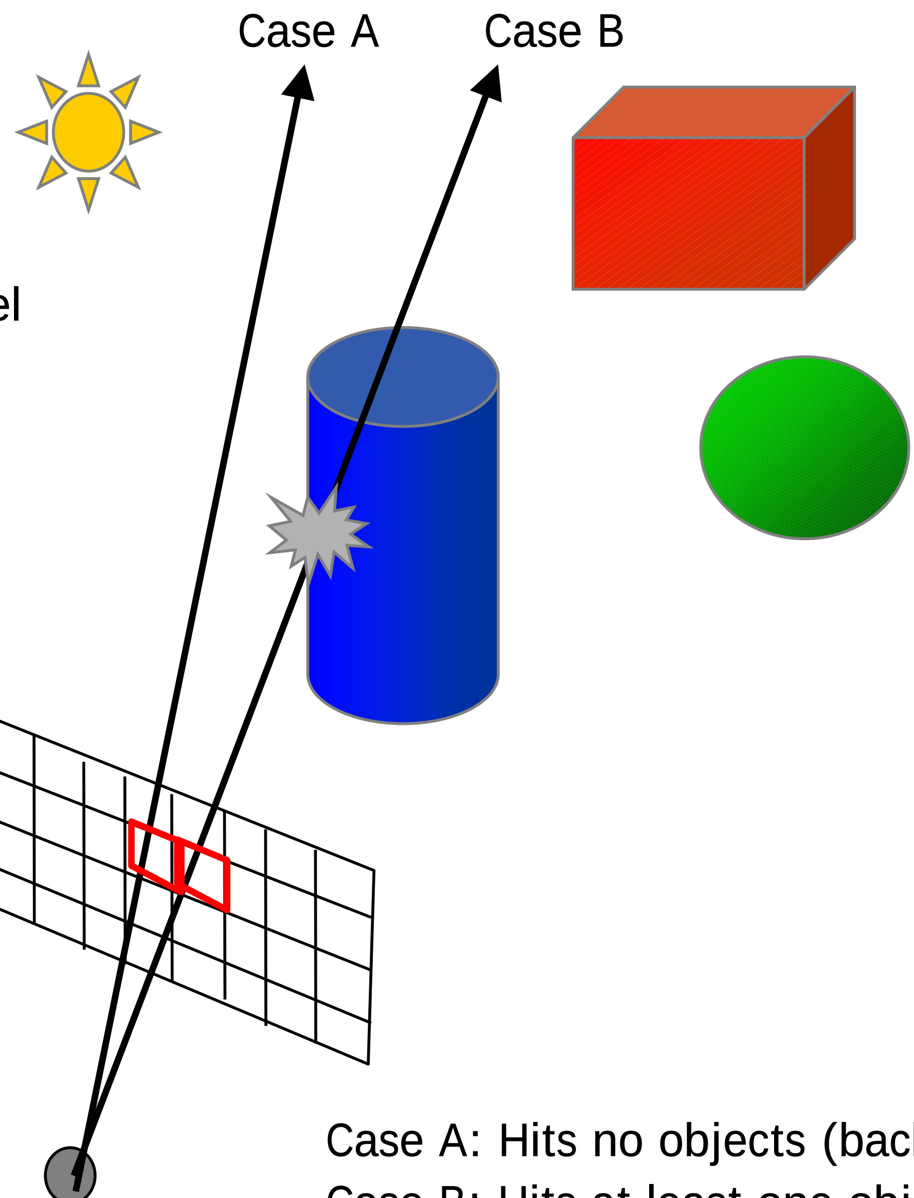


Ray Casting (Appel, 1968)

1. Build ray
2. Cast ray into scene through pixel

$$ray = origin + (direction)t$$

$$r(t) = eye + dir_{rc}t$$

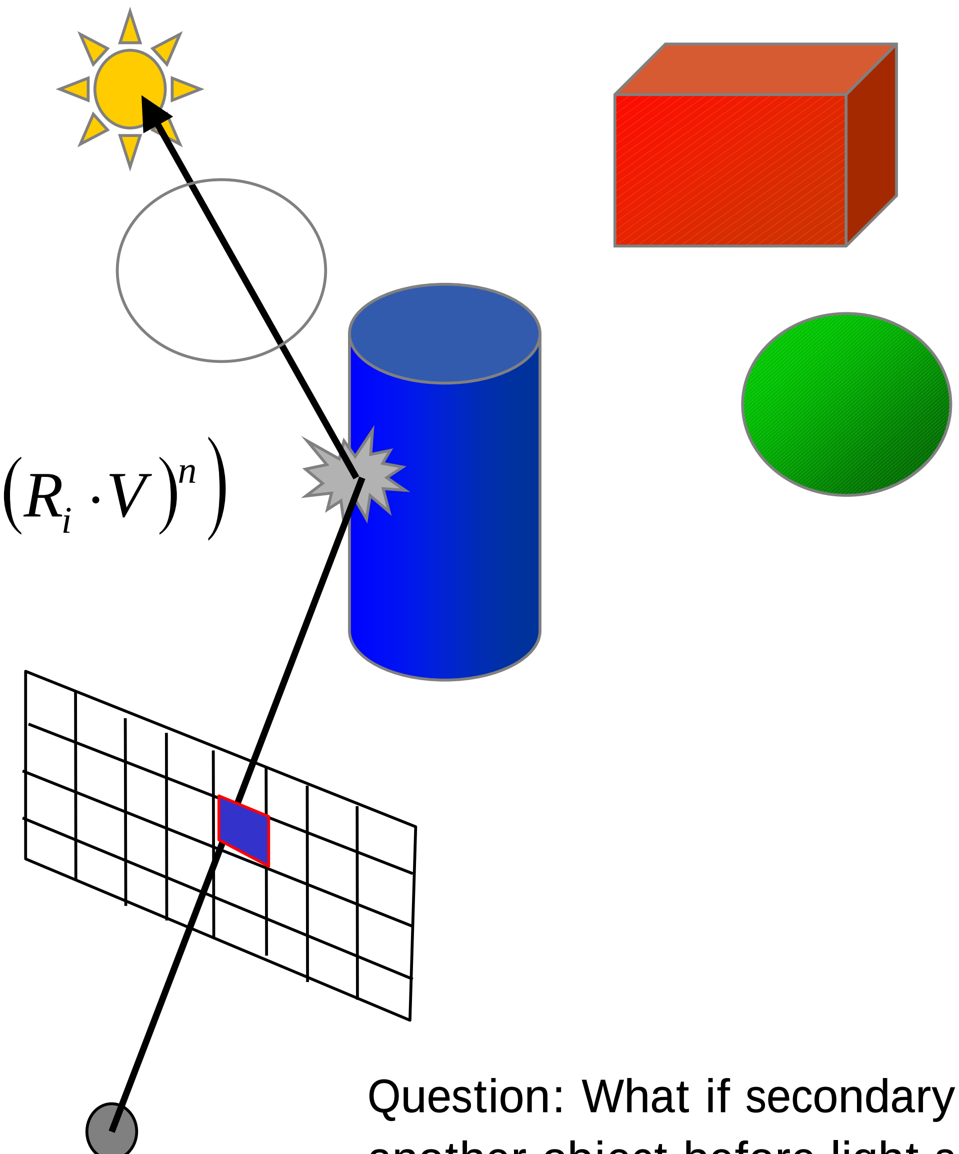


Case A: Hits no objects (background)
Case B: Hits at least one object

Ray Casting (Appel, 1968)

1. Ray hits object, build secondary ray to light source
2. Evaluate shading at hit point

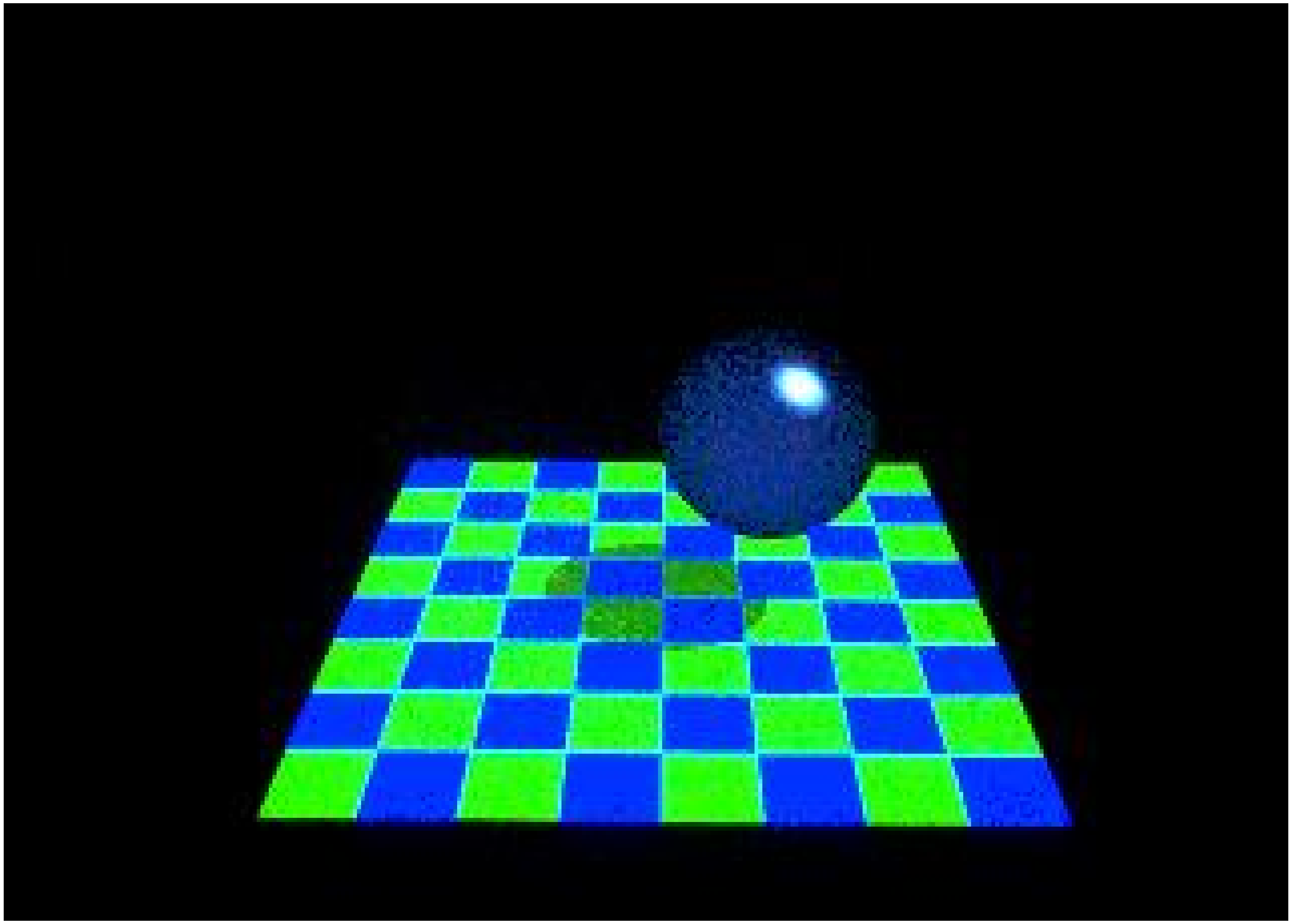
$$k_a I_a + \sum_{i=1}^{nls} I_i \left(k_d \left(L_i \cdot N \right) + k_s \left(R_i \cdot V \right)^n \right)$$



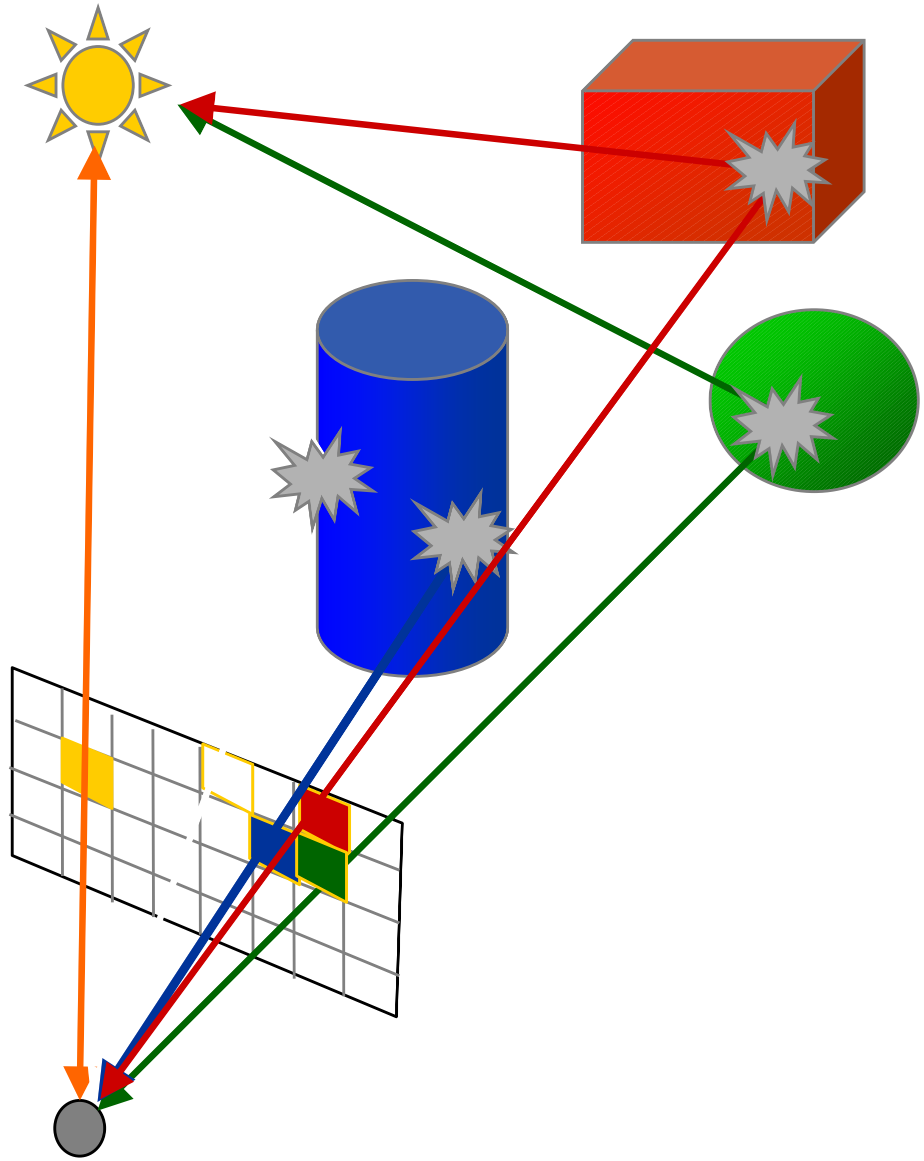
Question: What if secondary ray hits another object before light source?

Secondary Ray hits object

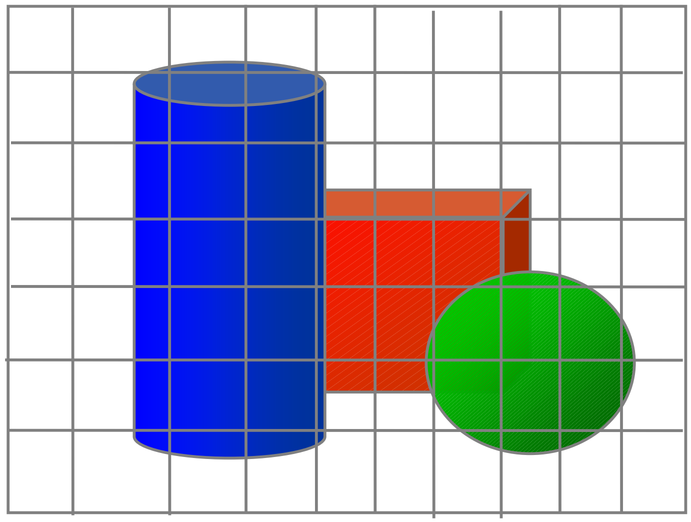
- First Intersection point in the shadow of the second object



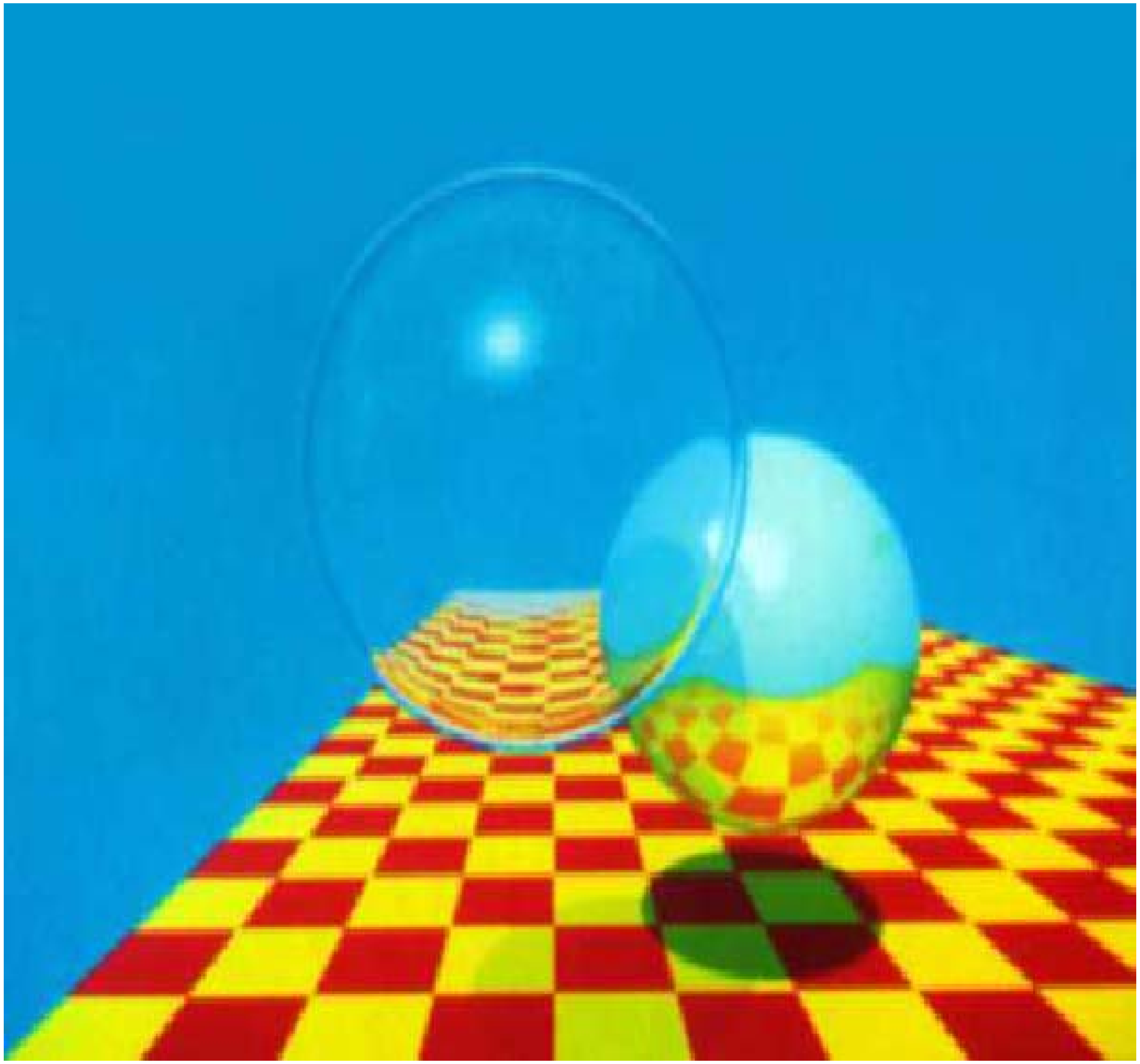
Ray Casting (Appel, 1968)



direct illumination

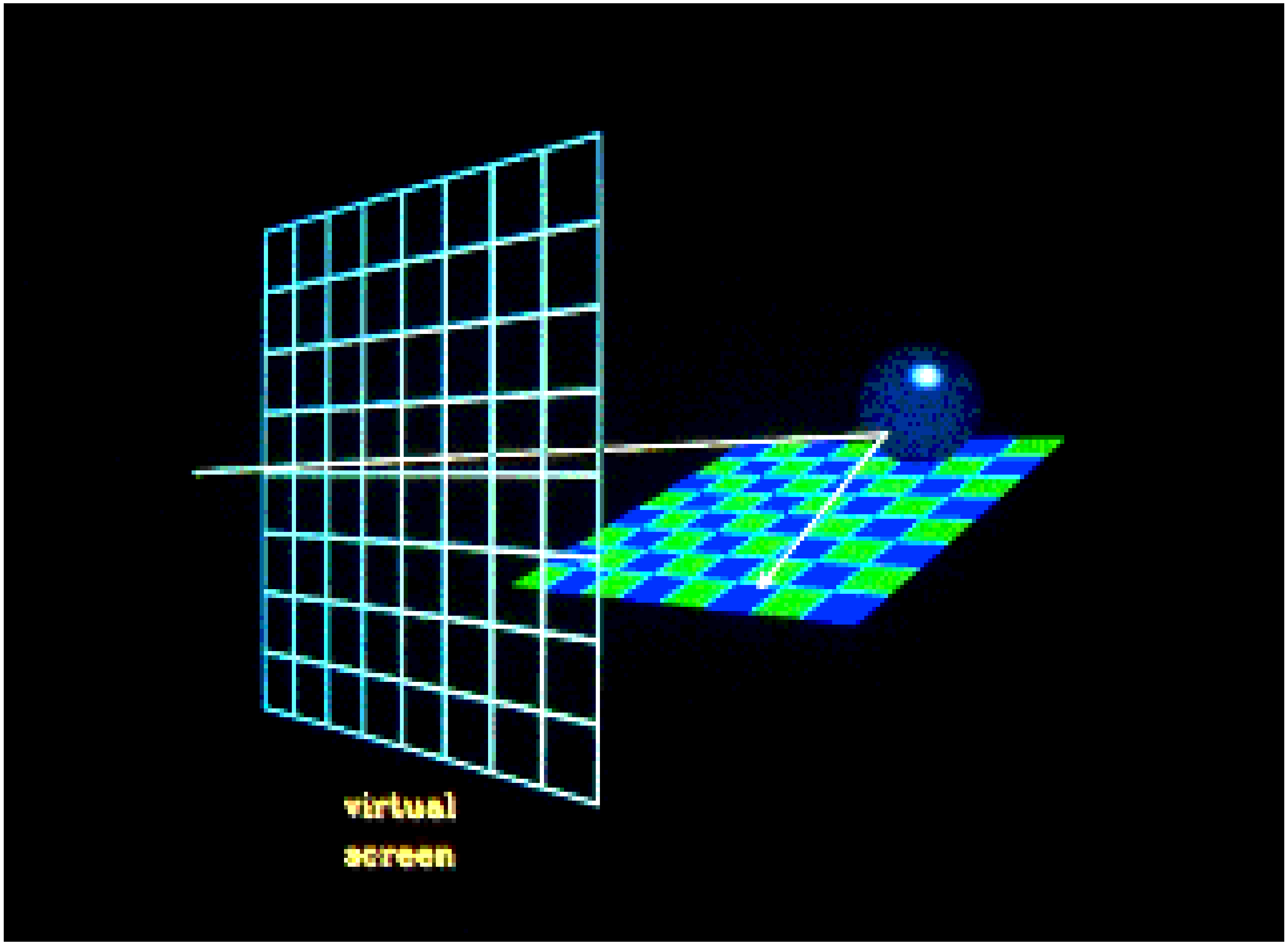


Recursive ray tracing (Whitted, 1980)

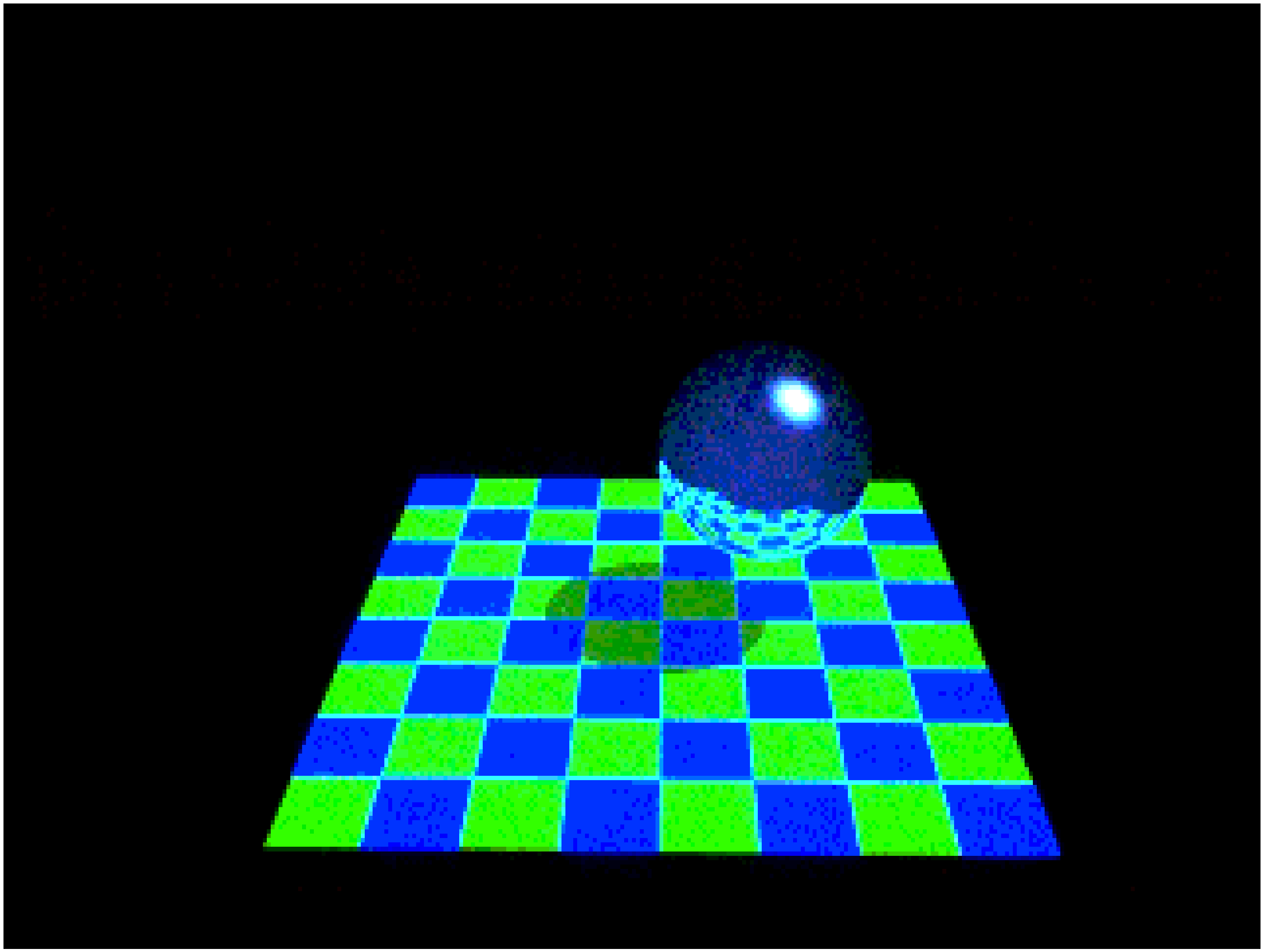


Reflected Ray

- When a ray hits an object, a reflected ray is generated which is tested against all of the objects in the scene.

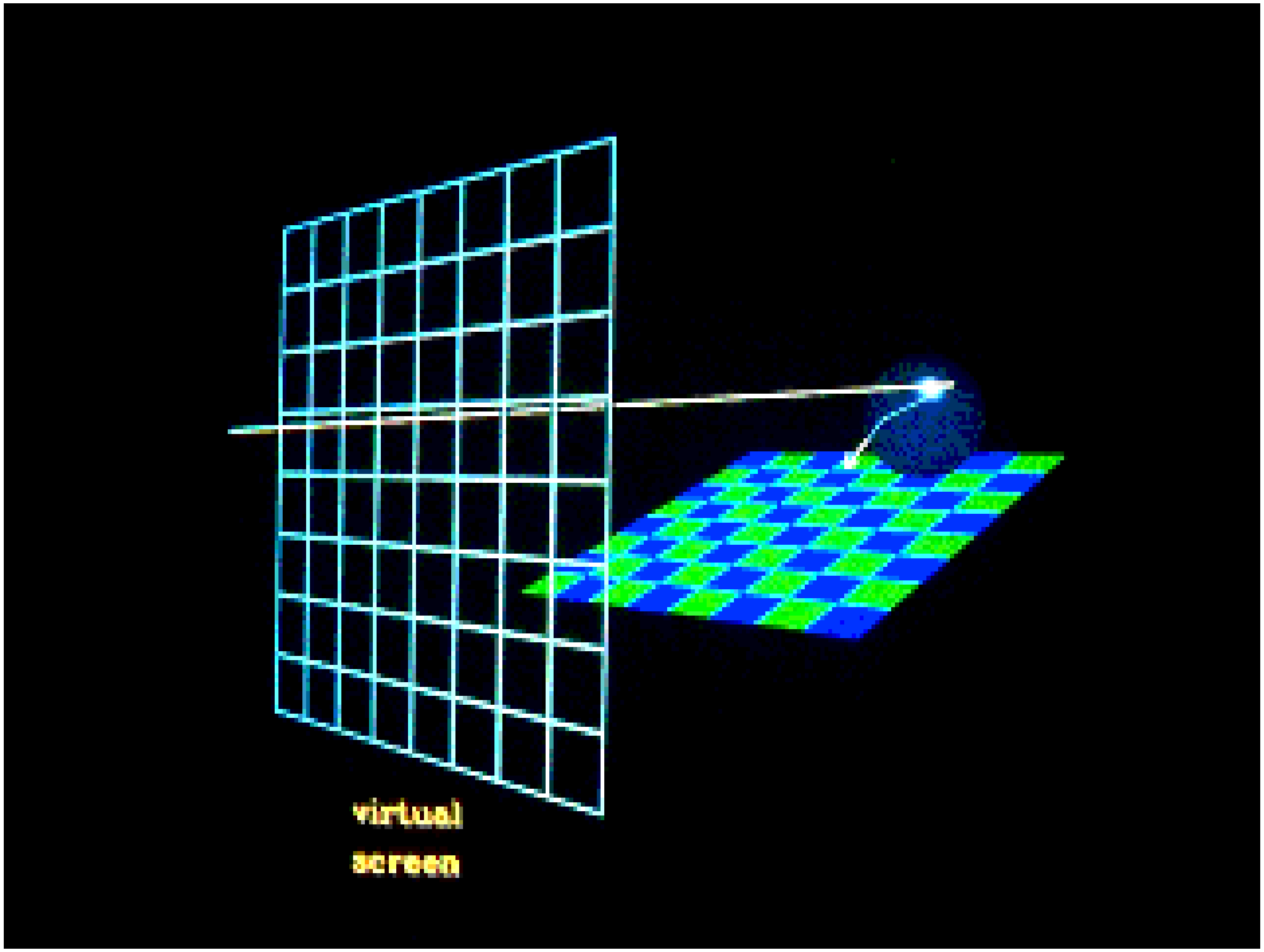


Reflection: Contribution from reflected ray

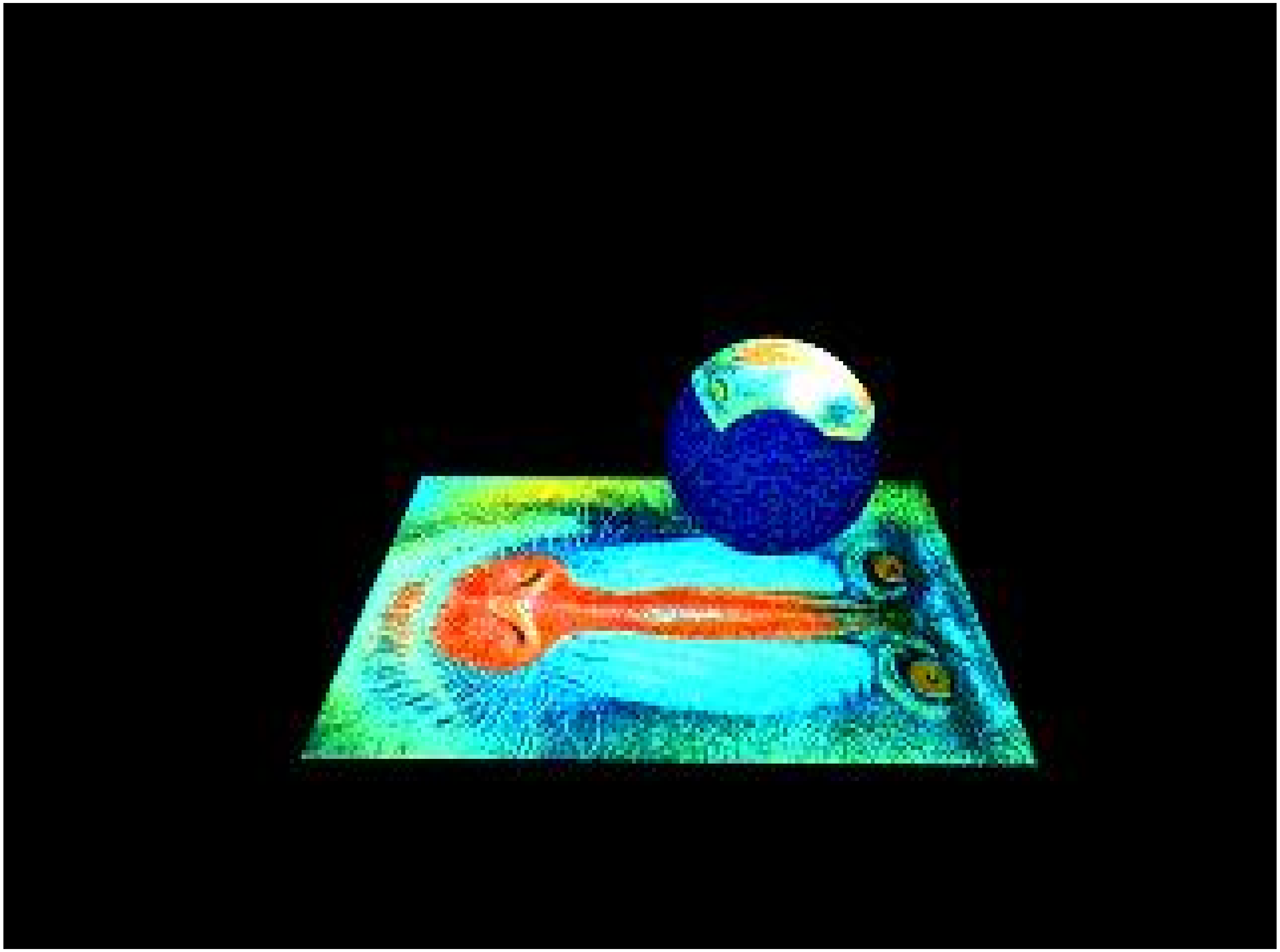


Transparency

- If intersected object is transparent, transmitted ray is generated and tested against all the objects in the scene.



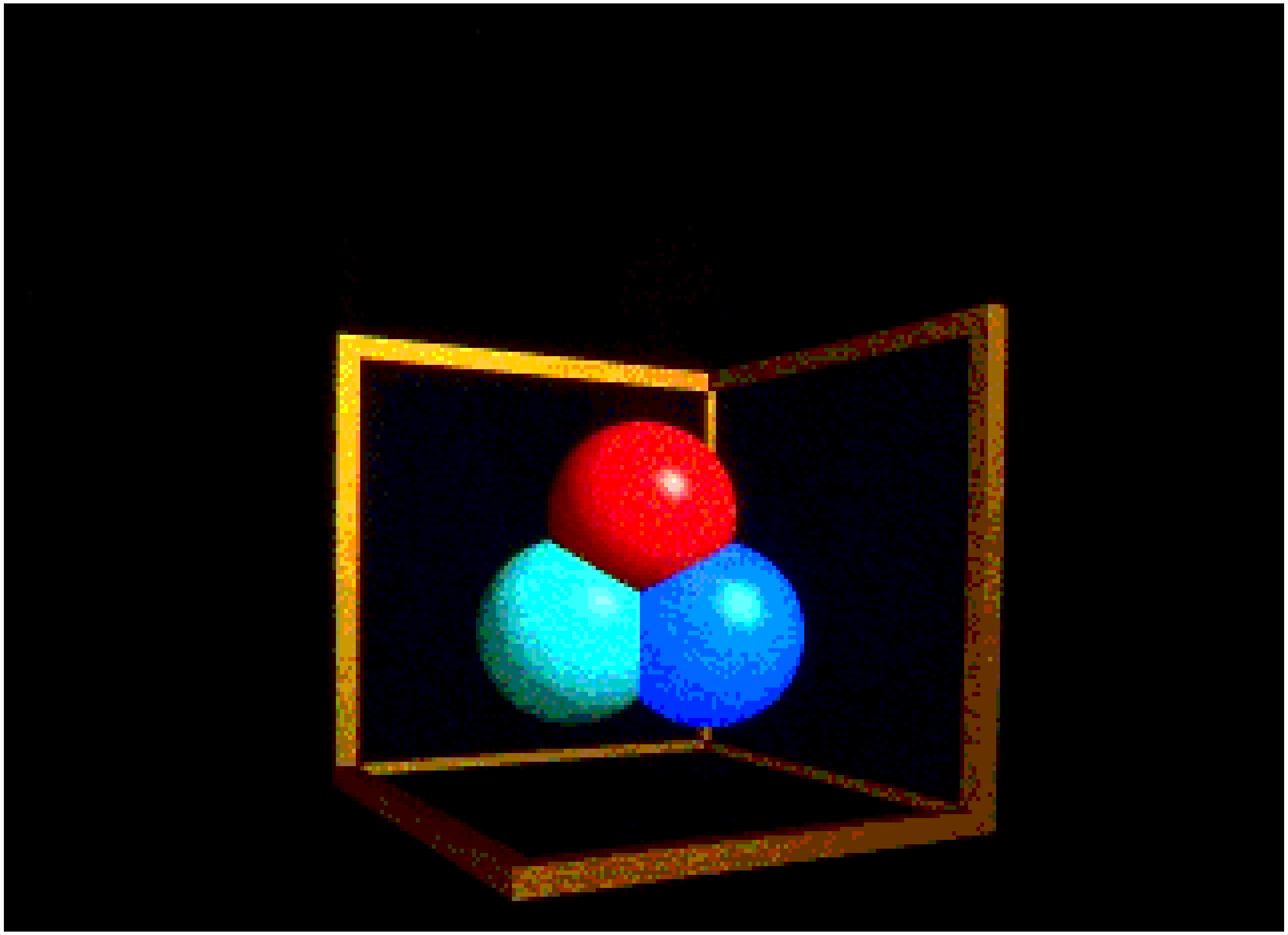
Transparency: Contribution from transmitted ray



Reflected Ray: Recursion

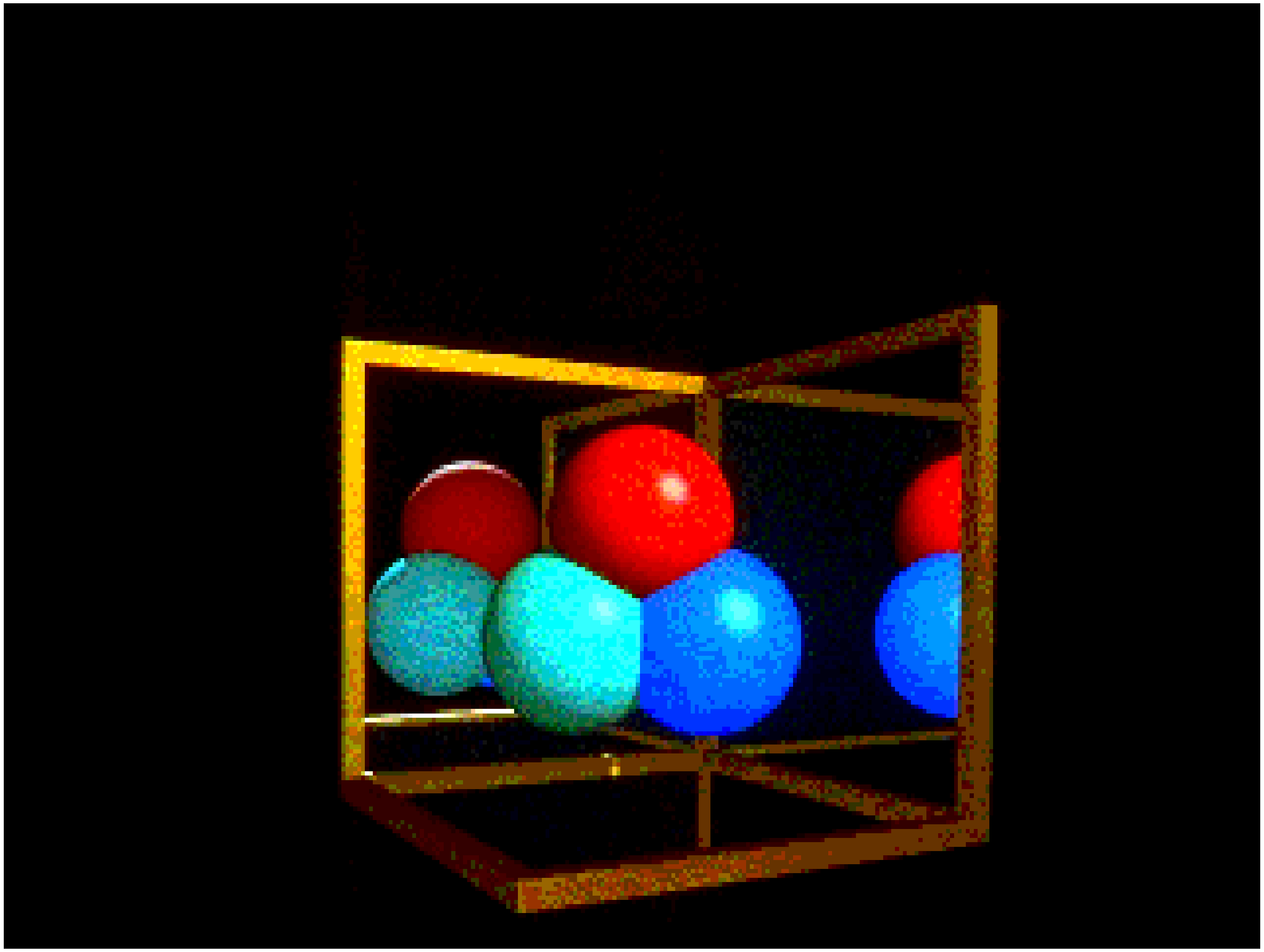
Reflected rays can generate other reflected rays that can generate other reflected rays, etc.

Case A: ***Scene with no reflection rays***



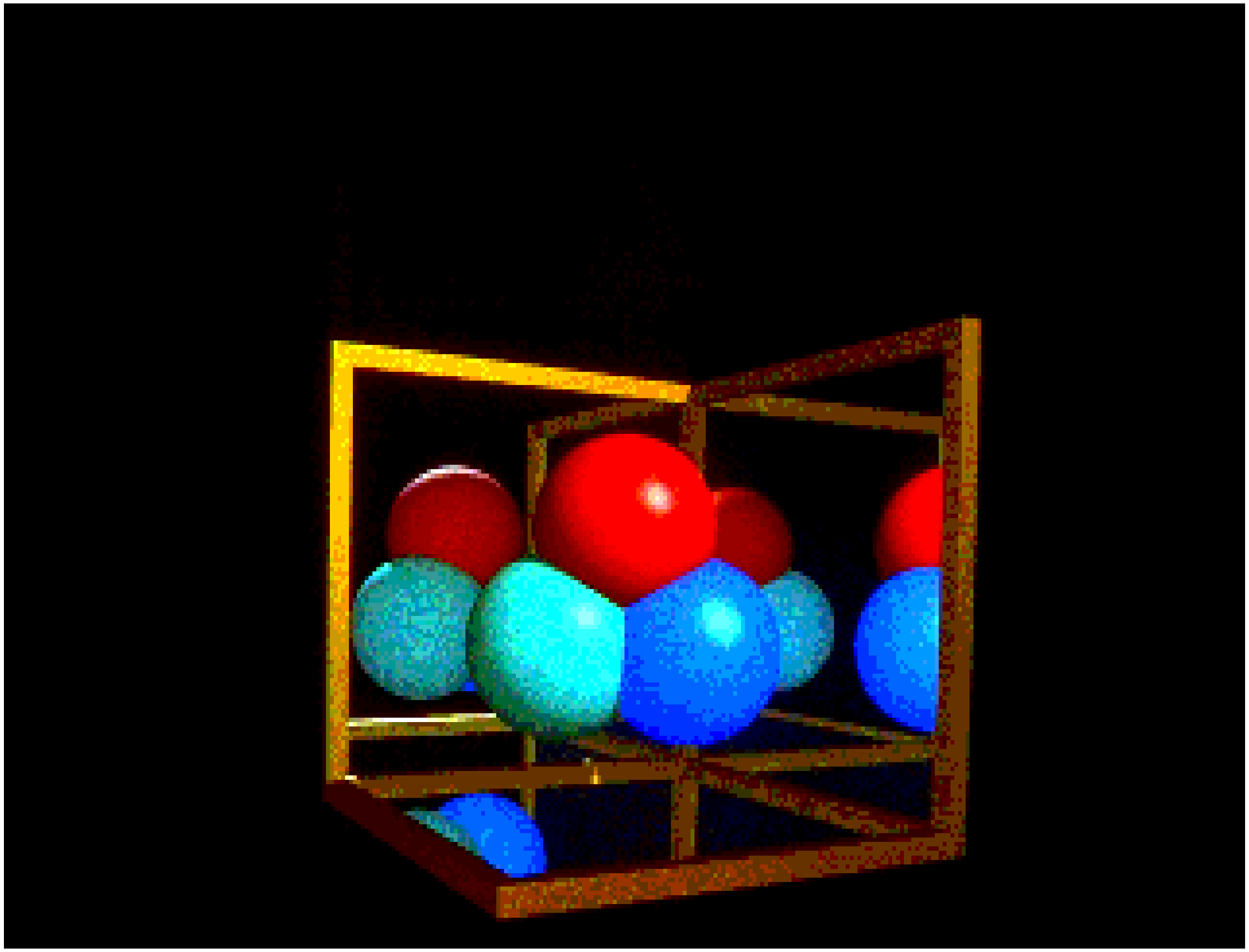
Reflected Ray: Recursion

Case B: Scene with one layer of reflection

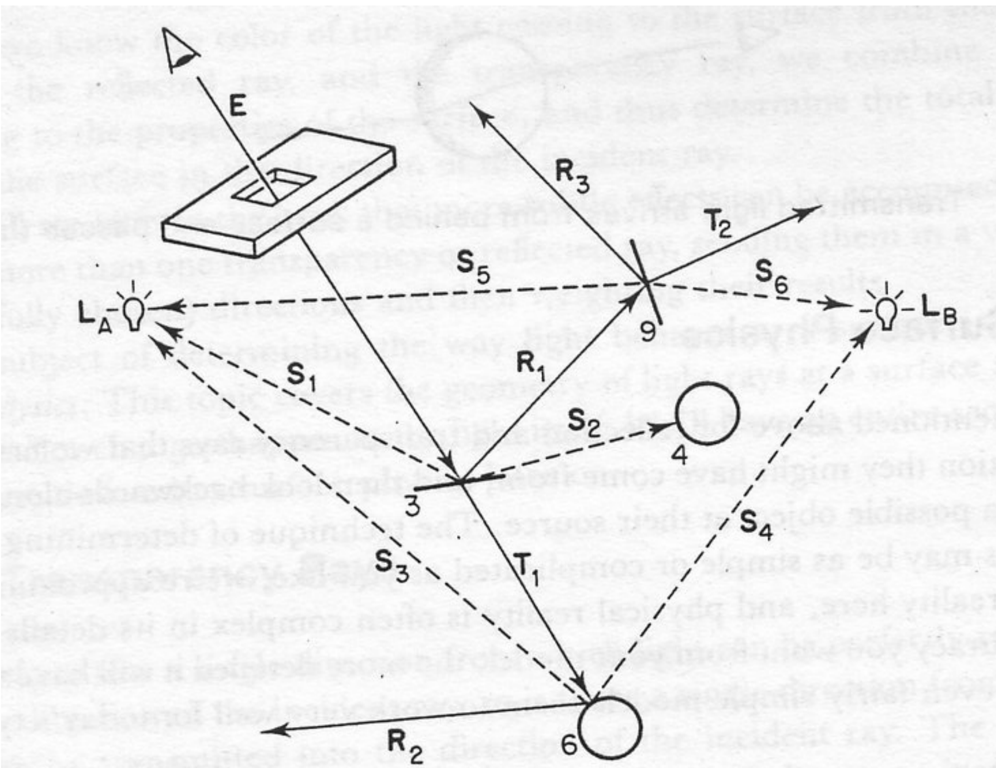


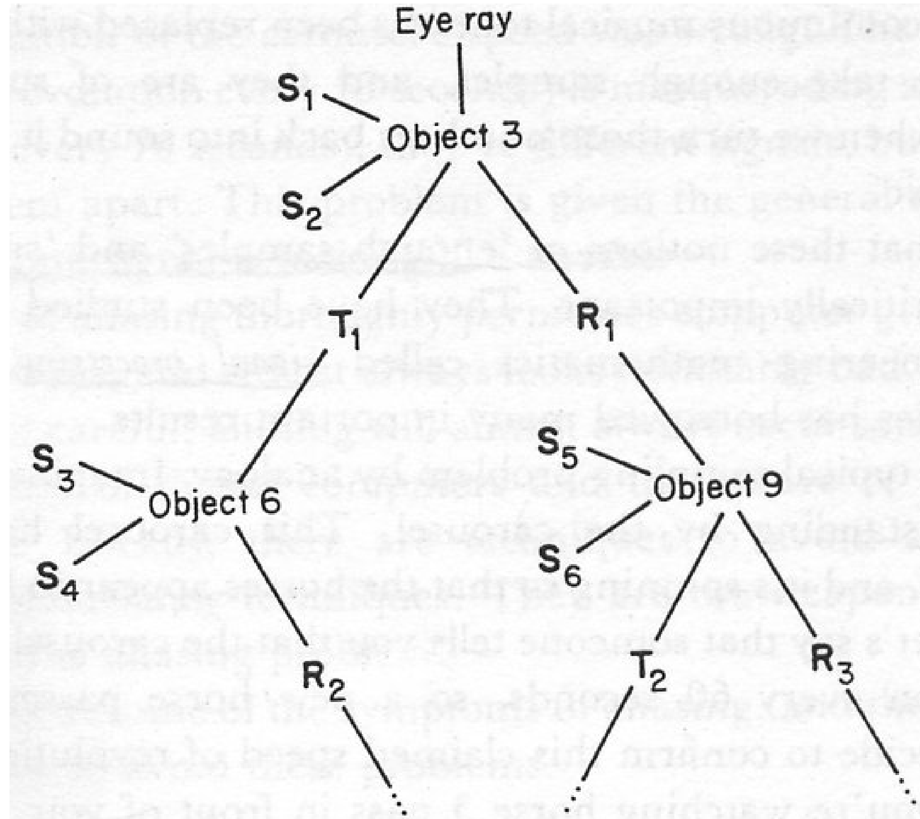
Reflected Ray: Recursion

Case C: Scene with two layers of reflection



- Recursive ray tracing creates tree of rays





- Reflective and/or transmitted rays are continually generated until ray leaves the scene without hitting any object or a preset recursion level has been reached.

Ray tracer components

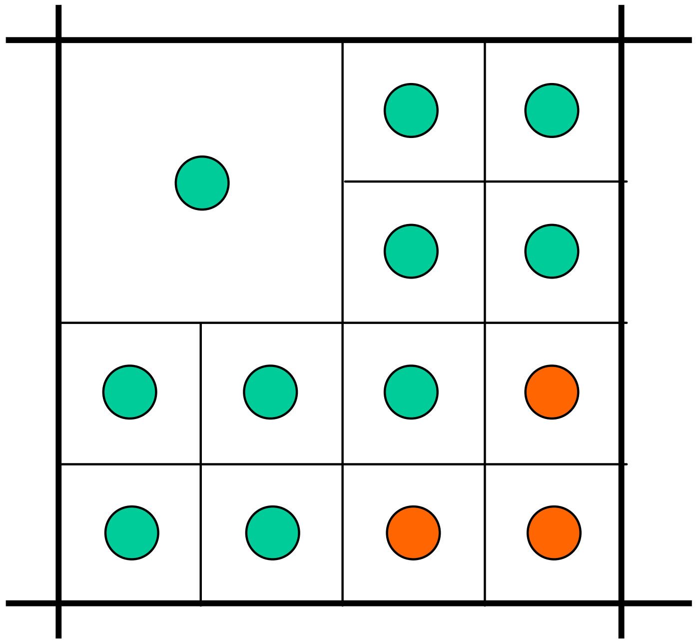
- Cameras
- Films
- Lights
- Ray-object intersection
- Visibility
- Surface scattering
- Recursive ray tracing

Why Ray Tracing Looks Fake / Effects

- Jagged edges
- Hard shadows
- Everything in focus
- Objects completely still
- Surfaces perfectly shiny
- Glass perfectly clear

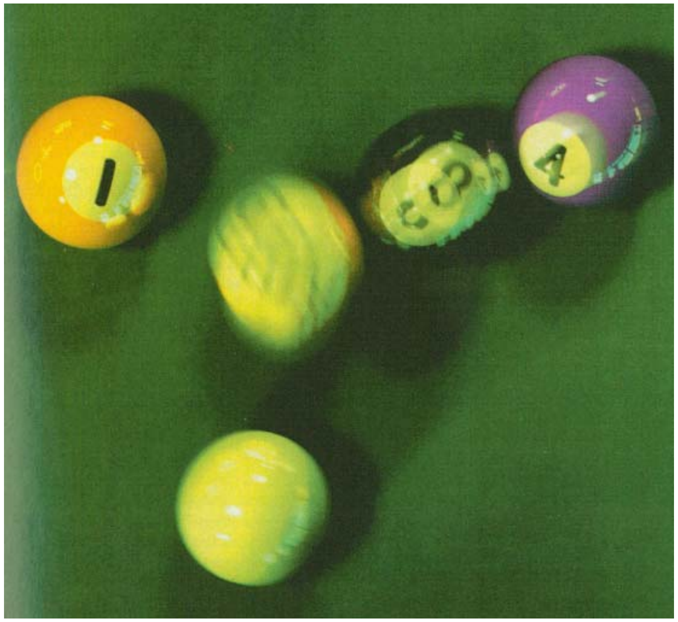
Why Ray Tracing Looks Fake

- Distributed Ray Tracing
  - Rob Cook, SIGGRAPH 84
  - Replace single ray with distribution of rays
  - Not just fat ray through pixel, but fat rays everywhere
  - Cast Multiple
    - Eye rays
    - Shadow rays
    - Reflection rays
    - Refraction rays
- Supersampling
  - Cast multiple rays from eye through different parts of same pixel

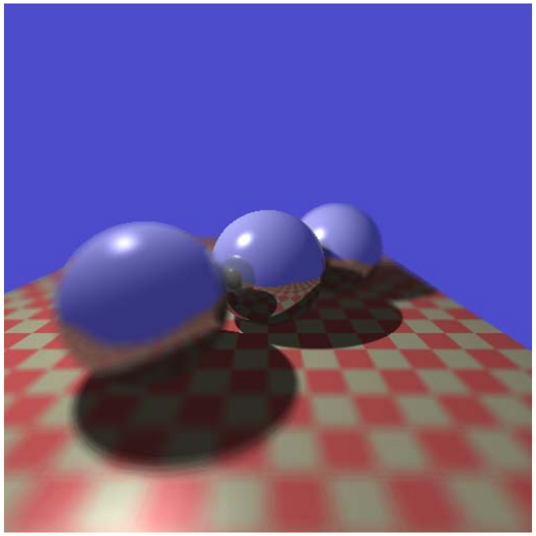


Why Ray Tracing Looks Fake

- Motion blur
  - Cast multiple rays from eye through same point in each pixel
  - Each of these rays intersects the scene at a different time
  - Reconstruction filter controls shutter speed, length

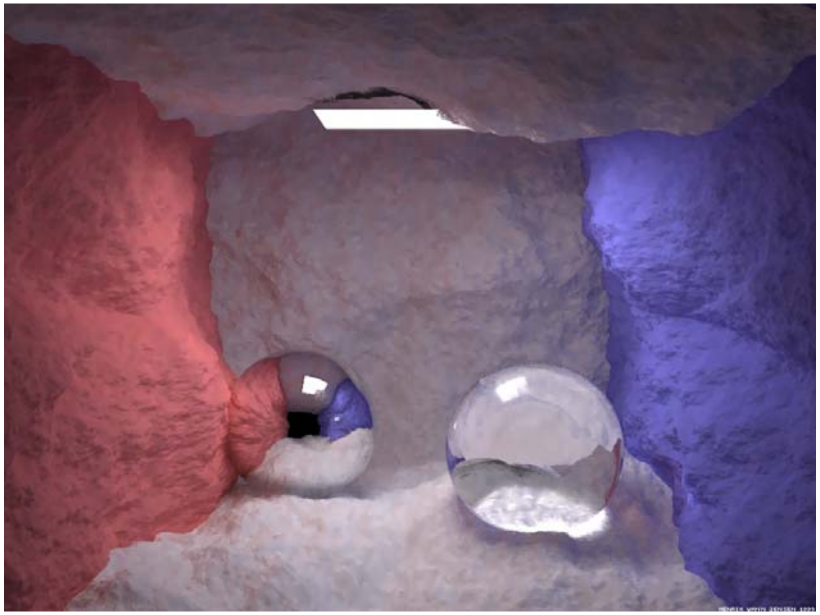


- Depth of Field
  - Better simulation of camera model
    - f-stop
    - focus
- Others (soft shadow, glossy, etc)



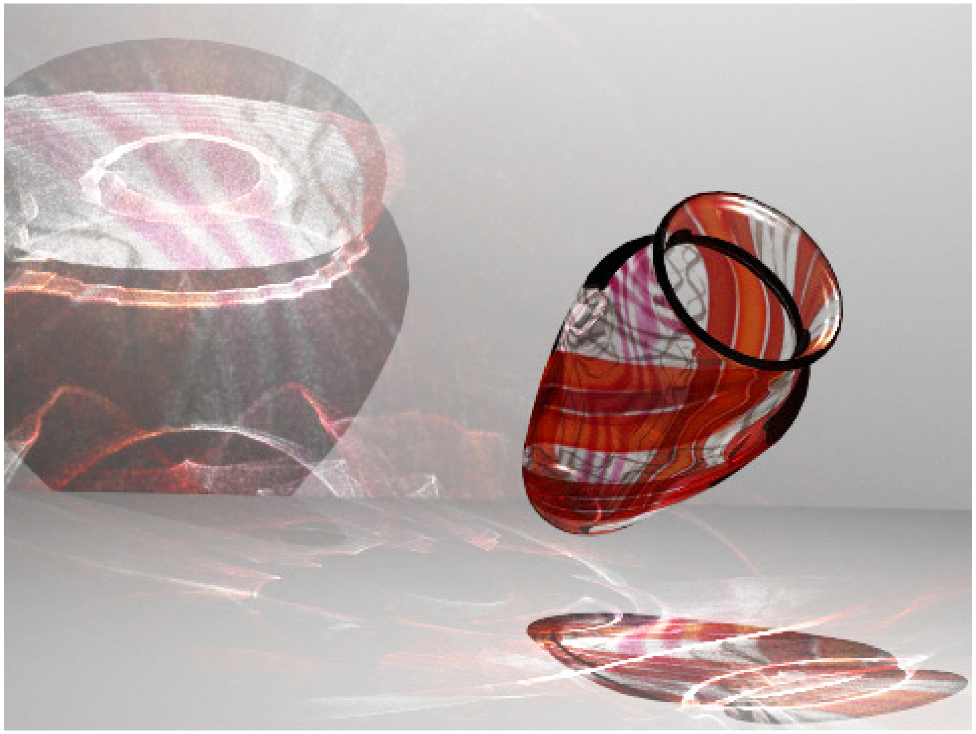
Photon Mapping

- Jensen EGRW 95, 96
- Simulates the transport of individual photons
- Two parts. First
  - Photons emitted from source
  - Photons deposited on surfaces
- Secondly:
  - Photons reflected from surfaces to other surfaces
  - Photons collected by rendering
- Good for:
  - Light through water
  - Cloud illumination
  - Marble



Rendering Techniques

- Photon mapping examples



Images: courtesy of Stanford rendering contest

Final words: To do

- Before next class
  - Read chapters 1 – 4 of text
  - Many concepts familiar to CS 543 students
  - If you did not take CS 543 with me, skim
    - Ray tracing chapter: F.S Hill, "Computer Graphics Using OpenGL", 2nd edition, Prentice Hall, 2000
- Homework 0
  - Download and install class ray tracer
  - Run several examples

References / Shamelessly stolen

- Pat Hanrahan, CS 348B, Spring 2005 class slides
- Yung-Yu Chuang, Image Synthesis, class slides, National Taiwan University, Fall 2005
- Kutulakos K, CSC 2530H: Visual Modeling, course slides
- UIUC CS 319, Advanced Computer Graphics Course slides
- http://www.siggraph.org/education/materials/HyperGraph/raytrace/rtrace0.htm