

Ray Tracing

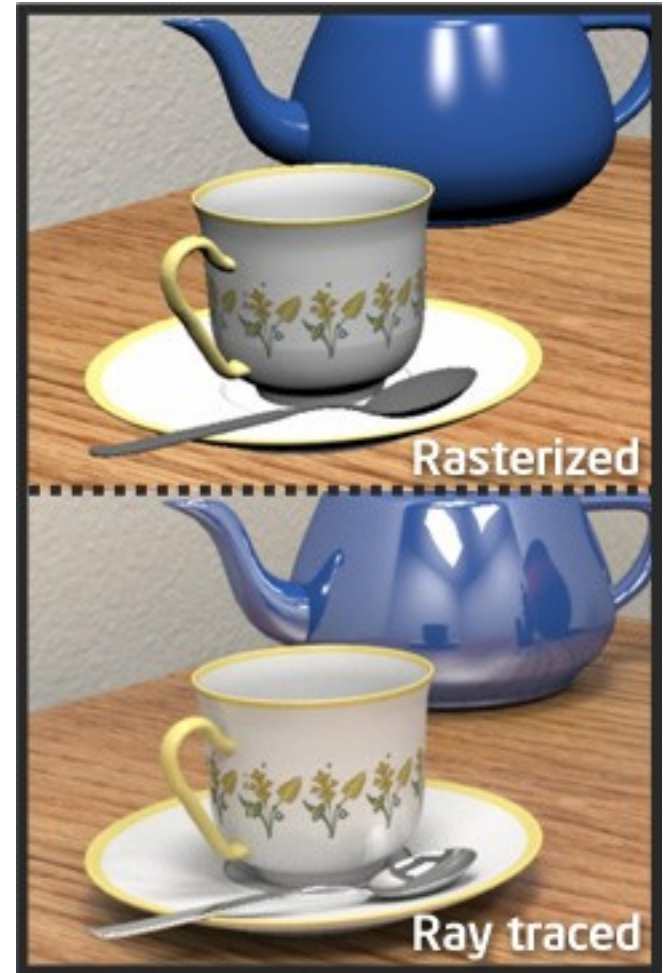
CSCI 4830/7000

Advanced Computer Graphics

Spring 2009

What is it?

- Method for rendering a scene using the concept of optical rays bouncing off objects
 - More realistic
 - Reflections
 - Shadows



How does it work?

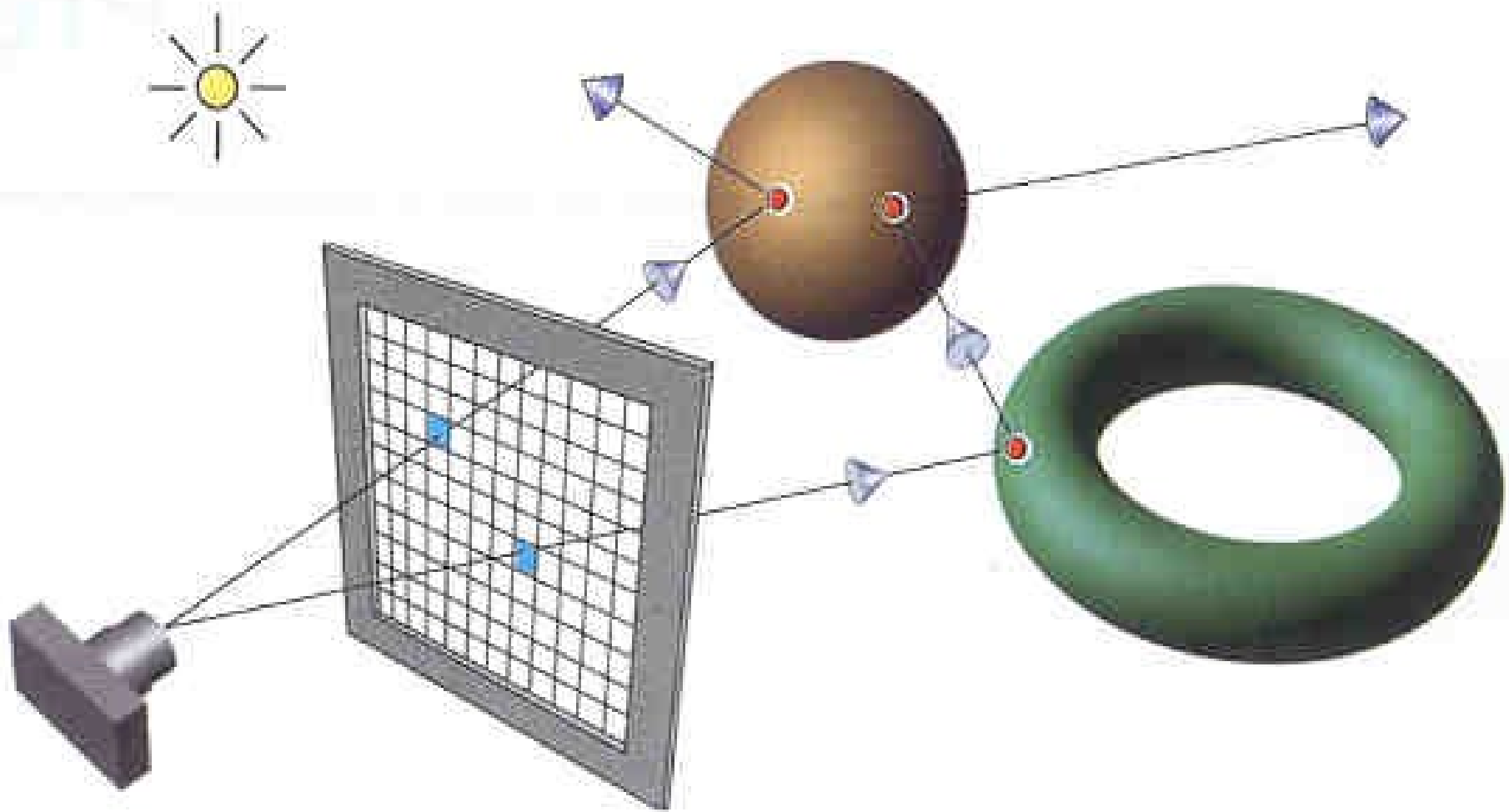


Figure 1. The ray-tracing process.

Sources

- Ray Tracing from the Ground Up
 - Kevin Suffern
 - Excellent tutorial
 - Some working examples
 - <http://www.raytracegroundup.com/>
- nVidia
- Intel
- Van Der Ploeg thesis

Interactive Ray Tracing

- True ray tracing is VERY compute intensive
- Global problem –scene complexity adds effort
- Generally there is no upper limit to computation
- Solutions are generally software based
 - Dedicated hardware may be near
 - <http://www.caustic.com/>



nVidia Quadra Plex
1920x1024@30fps



nVidia Quadra Plex
1920x1024@30fps

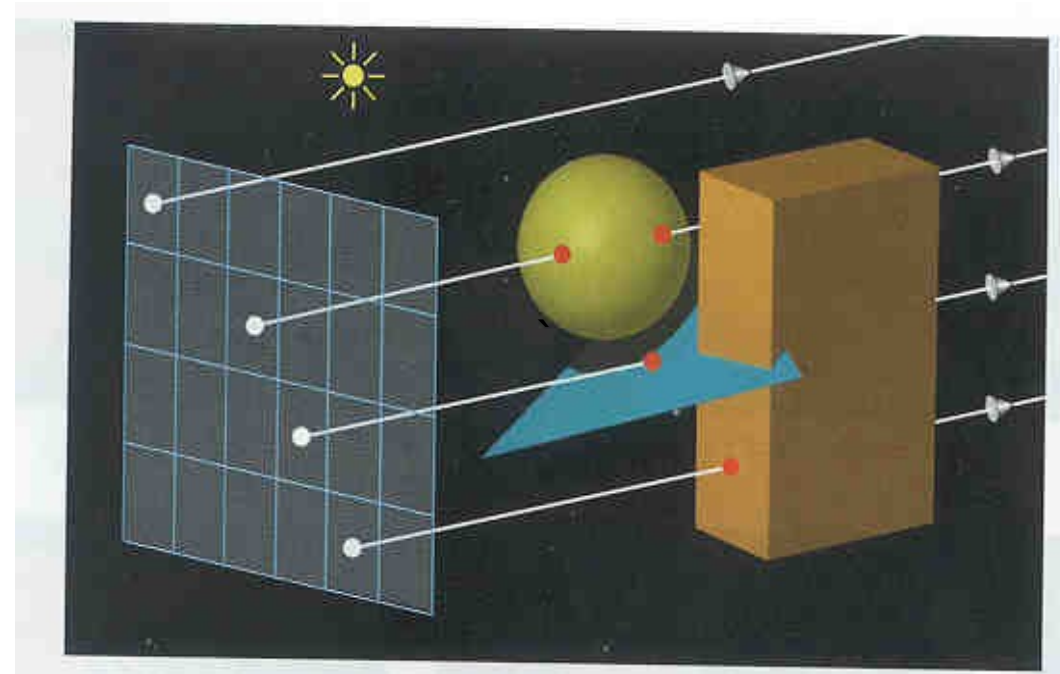


How is it Done?

- Scene Description Language
 - Defines objects in scene
 - Geometry and properties
 - Lights
 - Eye position
- Determine color of individual pixels using ray tracing algorithms
 - Very hard to do real time

How ray tracing works

- Define scene and view
 - objects
 - lights
 - eye
- For each pixel
 - Shoot ray from pixel
 - Find nearest hit
 - Use object properties and lights to calculate color, or set to black if no hits



True Global Ray Tracing

- Light can bounce many times
 - Color changes at each bounce
 - Each bounce attenuates light
 - Light scatters in complex ways
 - Objects block light
- This simple scene took 2 CPU years to render
 - Cornell Box
 - Area light and three boxes

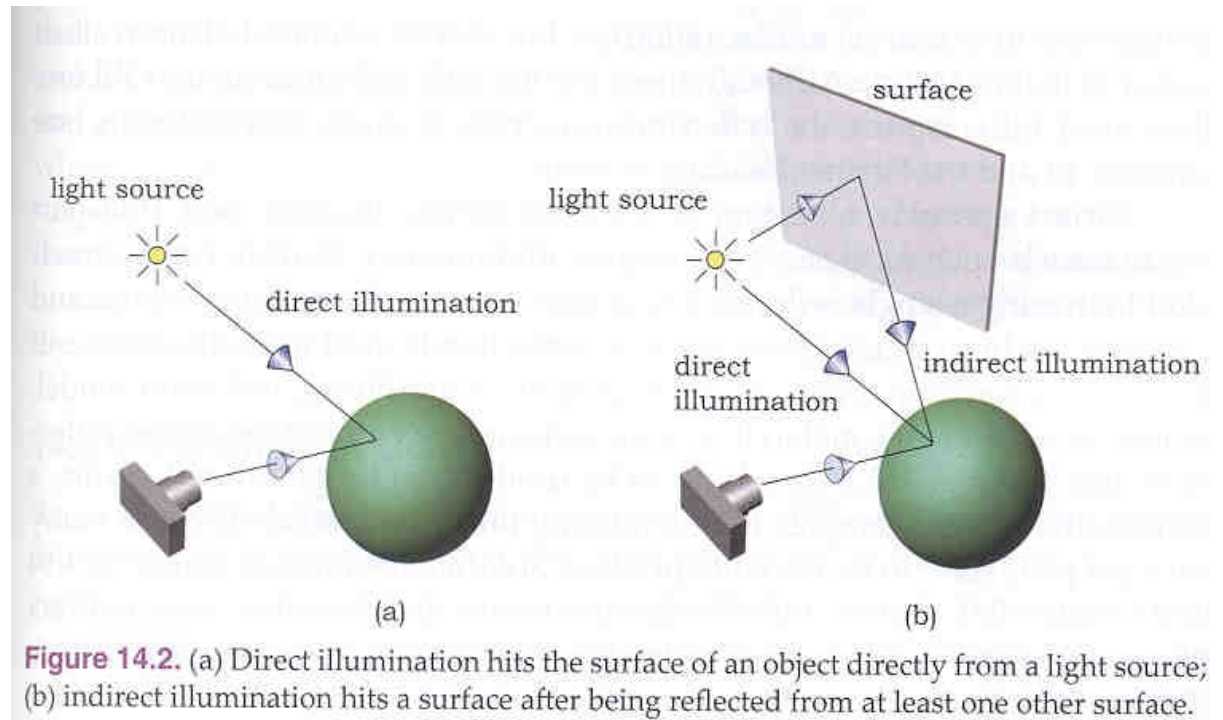


Efficiency and Complexity

- Most ray tracers written in C++
 - Object Oriented paradigm for objects, rays, colors
 - Good efficiency/readability trade-off
- Efficiency is a HUGE deal
 - Pushing the envelope of hardware
 - Algorithm is global by definition
- Recursion and complexity
 - Need clean interface on objects

What is a Ray?

- $\mathbf{p} = \mathbf{o} + t \mathbf{d}$
- Types of rays
 - Primary rays
 - Secondary rays
 - Shadow rays
 - Light rays
- Rays are one directional



Intersections

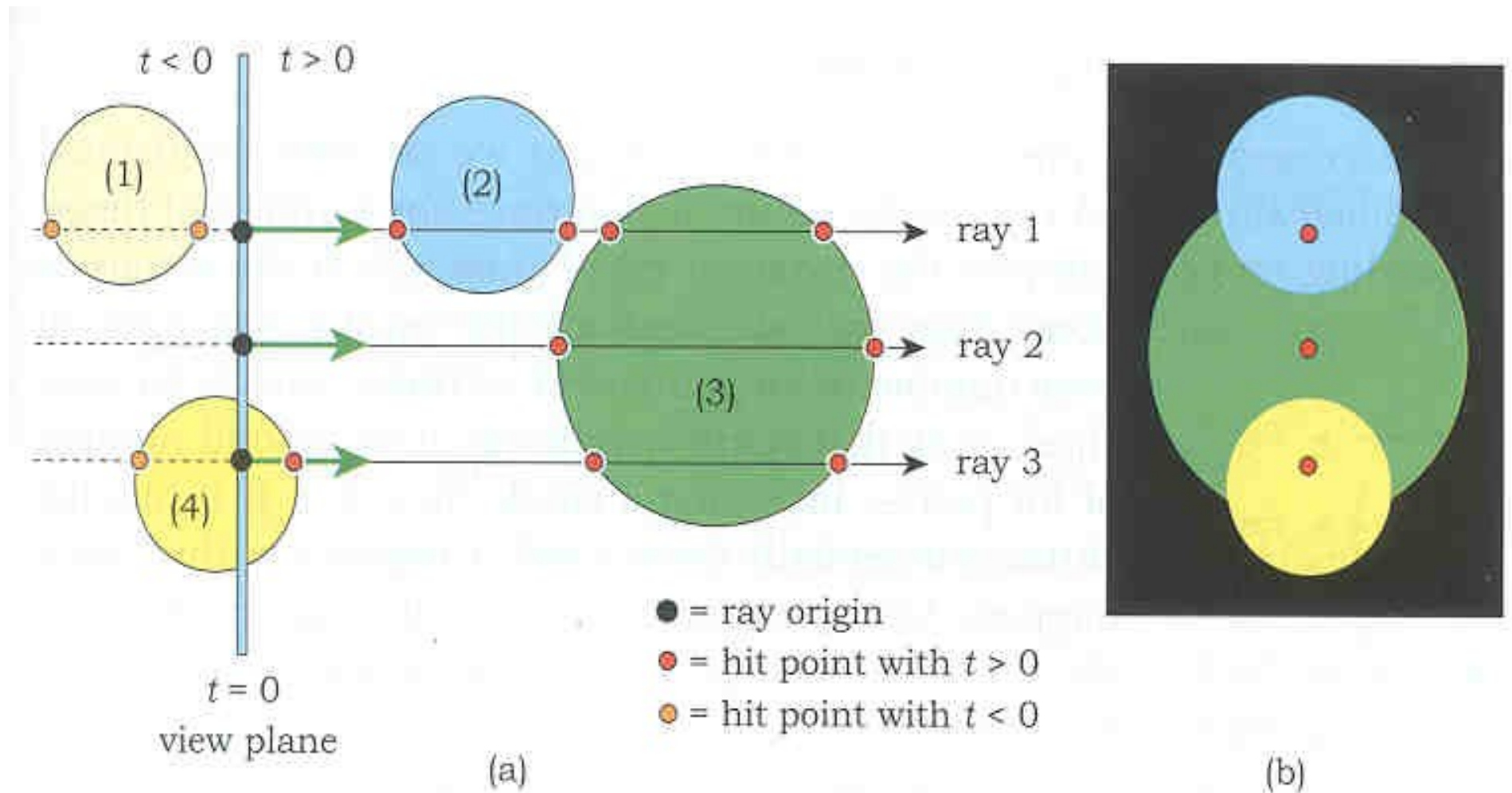


Figure 3.4. (a) Rays and their intersections with spheres; (b) ray-traced image of the spheres.

Intersecting a Sphere

- Simplest 3D object

- Center
- Radius

- Smooth normal

- Intersections

- none
- once
 - tangent
 - internal
- twice

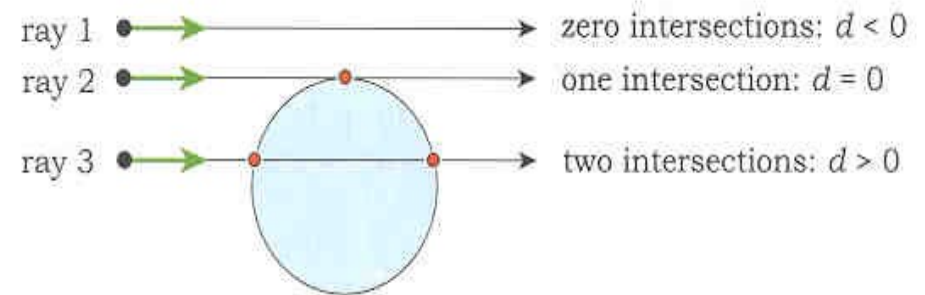


Figure 3.7. Ray-sphere intersections.

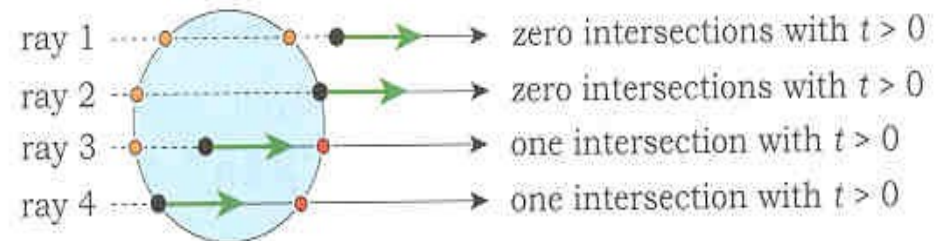


Figure 3.8. Further ray-sphere intersections.

Implicit Surfaces

- General
 - $f(x,y,z) = 0$
- Plane: Point **a** and Normal **n**
 - $(p-a) \bullet n = 0$
- Sphere
 - $(p-a) \bullet (p-a) - r^2 = 0$
- Triangle
 - Limit plane