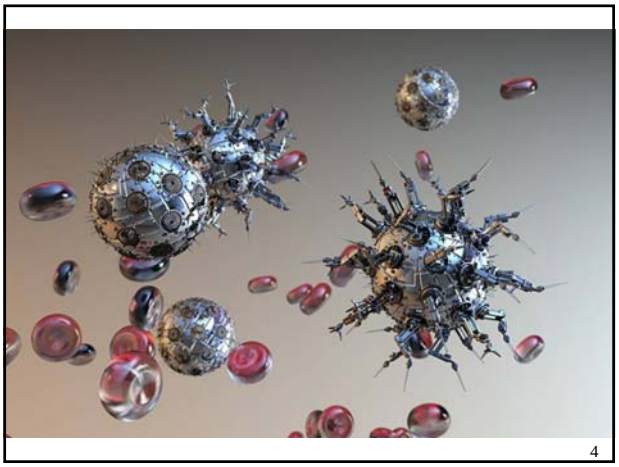
Ray Tracing

MIT EECS 6.837
Most slides are taken from Frédo Durand and Barb Cutler
Some slides courtesy of Leonard McMillan

1

3



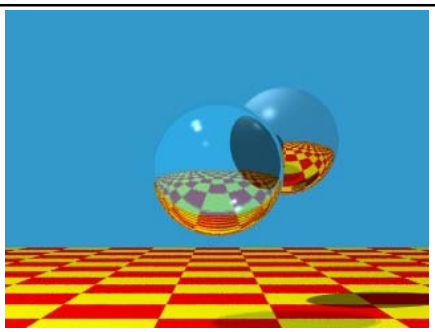
4

Ray Tracing

- Ray Tracing kills two birds with one stone:
  - Solves the Hidden Surface Removal problem
  - Evaluates an improved global illumination model
    - shadows
    - ideal specular reflections
    - ideal specular refractions
  - Enables direct rendering of a large variety of geometric primitives
- Book: A. Glassner, An Introduction to Ray Tracing
- Web: `http://www.cs.cf.ac.uk/Ray.Tracing`

5

Ray Tracing



Recursive ray tracing: Turner Whitted, 1980

6

Backward Tracing

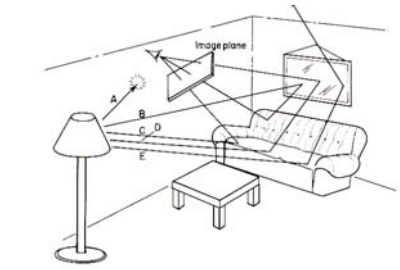
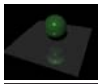


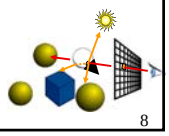
Fig. 5. Some light rays (like A and E) never reach the image plane at all. Others follow simple or complicated routes.

7

Overview of today

- Shadows
- Reflection
- Refraction
- Recursive Ray Tracing

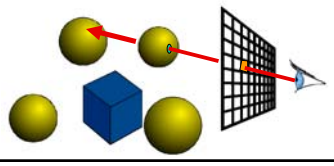


8

Ray Casting (a.k.a. Ray Shooting)

```
For every pixel (x,y)
  Construct a ray from the eye
  color[x,y]=castRay(ray)
```

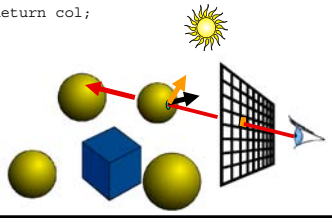
- Complexity?
  - O(n * m)
  - n: number of objects, m: number of pixels



9

Ray Casting with diffuse shading

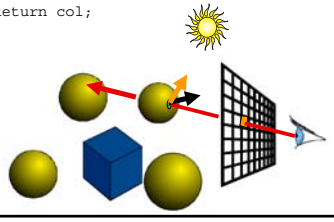
```
Color castRay(ray)
  Hit hit();
  For every object ob
      ob->intersect(ray, hit, tmin);
   Color col=ambient*hit->getColor();
   For every light L
      col=col+hit->getColorL()*L->getColor*
        L->getDir()->Dot3( hit->getNormal() );
   Return col;
```



10

Encapsulating shading

```
Color castRay(ray)
  Hit hit();
  For every object ob
      ob->intersect(ray, hit, tmin);
   Color col=ambient*hit->getMaterial()->getDiffuse();
   For every light L
      col=col+hit->getMaterial()->shade
        (ray, hit, L->getDir(), L->getColor());
   Return col;
```



11

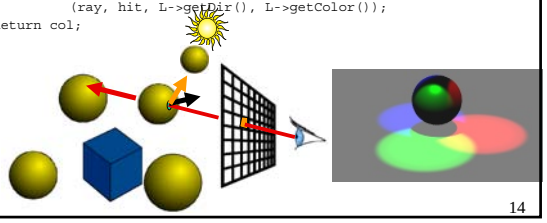
Questions?

- Image computed using the RADIANCE system by Greg Ward

12

Shadows

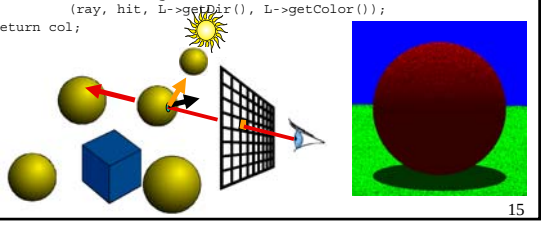
```
Color castRay(ray)
    Hit hit();
    For every object ob
        ob->intersect(ray, hit, tmin);
    Color col=ambient*hit->getMaterial()->getDiffuse();
    For every light L
        Ray ray2(hitPoint, L->getDir()); Hit hit2(L->getDist(),,)
        For every object ob
            ob->intersect(ray2, hit2, 0);
        If (hit->getT> L->getDist())
            col=col+hit->getMaterial()->shade
              (ray, hit, L->getDir(), L->getColor());
    Return col;
```



14

Shadows – problem?

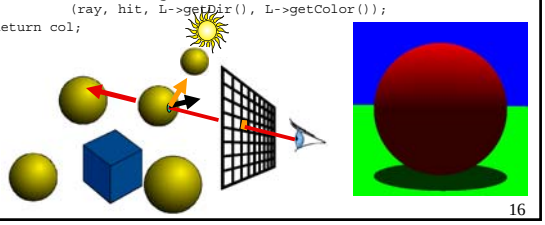
```
Color castRay(ray)
    Hit hit();
    For every object ob
        ob->intersect(ray, hit, tmin);
    Color col=ambient*hit->getMaterial()->getDiffuse();
    For every light L
        Ray ray2(hitPoint, L->getDir()); Hit hit2(L->getDist(),,)
        For every object ob
            ob->intersect(ray2, hit2, 0);
        If (hit->getT> L->getDist())
            col=col+hit->getMaterial()->shade
              (ray, hit, L->getDir(), L->getColor());
    Return col;
```



15

Avoiding self shadowing

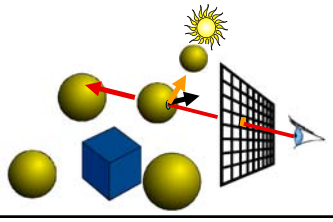
```
Color castRay(ray)
    Hit hit();
    For every object ob
        ob->intersect(ray, hit, tmin);
    Color col=ambient*hit->getMaterial()->getDiffuse();
    For every light L
        Ray ray2(hitPoint, L->getDir()); Hit hit2(L->getDist(),,)
        For every object ob
            ob->intersect(ray2, hit2, epsilon);
        If (hit->getT> L->getDist())
            col=col+hit->getMaterial()->shade
              (ray, hit, L->getDir(), L->getColor());
    Return col;
```



16

Shadow optimization

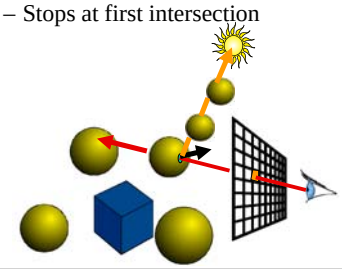
- Shadow rays are special
- How can we accelerate our code?



17

Shadow optimization

- We only want to know whether there is an intersection, not which one is closest
- Special routine `Object3D::intersectShadowRay()`
  - Stops at first intersection



18

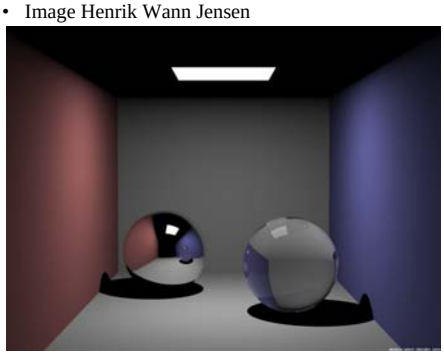
Shadow ray casting history

- Due to Appel [1968]
- First shadow method in graphics
- Not really used until the 80s

19

Questions?

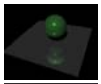
- Image Henrik Wann Jensen

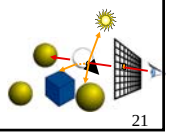


20

Overview of today

- Shadows
- Reflection
- Refraction
- Recursive Ray Tracing

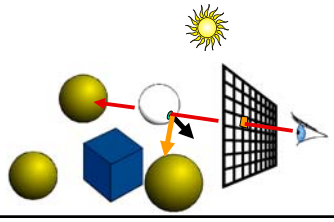
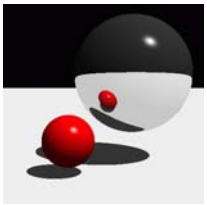





21

Mirror Reflection

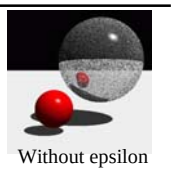
- Compute mirror contribution
- Cast ray
  - In direction symmetric wrt normal
- Multiply by reflection coefficient (color)

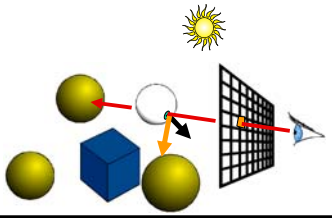
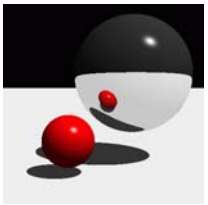



22

Mirror Reflection

- Cast ray
  - In direction symmetric w.r.t normal
- Don't forget to add epsilon to the ray

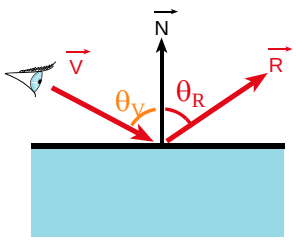

Without epsilon



With epsilon

23

Reflection

- Reflection angle = view angle

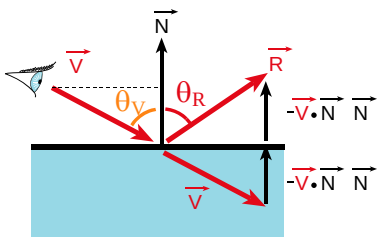


24

Reflection

- Reflection angle = view angle

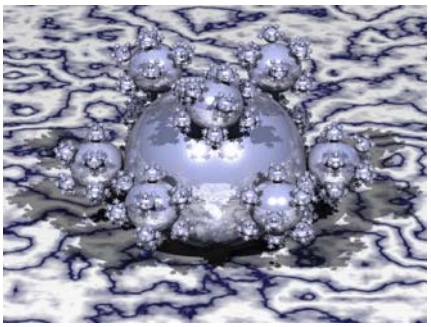
$$\vec{R} = \vec{V} - 2(\vec{V} \bullet \vec{N})\vec{N}$$



25

Questions?

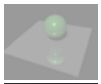
- Image by Henrik Wann Jensen

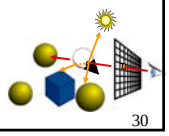


29

Overview of today

- Shadows
- Reflection
- Refraction
- Recursive Ray Tracing

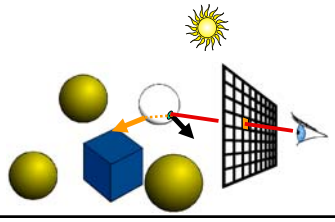





30

Transparency

- Compute transmitted contribution
- Cast ray
  - In refracted direction
- Multiply by transparency coefficient (color)



31

Qualitative refraction

- From “Color and Light in Nature” by Lynch and Livingston

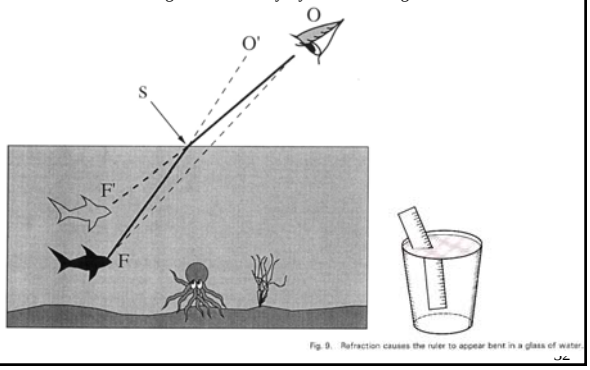
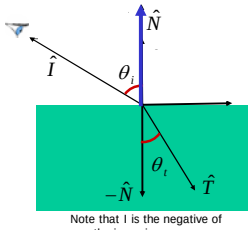


Fig. 9. Refraction causes the ruler to appear bent in a glass of water.

32

Refraction

Snell-Descartes Law $\frac{\sin\theta_i}{\sin\theta_t} = \frac{\eta_t}{\eta_i} = \eta_r$



Note that I is the negative of the incoming ray

34

Total internal reflection

- From “Color and Light in Nature” by Lynch and Livingstone


Fig. 3.7A The optical manhole. From under water, the entire celestial hemisphere is compressed into a circle only 97.2° across. The dark boundary defining the edges of the manhole is not sharp due to surface waves. The rays are analogous to the crepuscular type seen in hazy air. Section 1.9. (Photo by D. Granger)

Fig. 3.7B The optical manhole. Light from the horizon (angle of incidence = 90°) is refracted downward at an angle of 48.6°. This compresses the sky into a circle with a diameter of 97.2° instead of its usual 180°.

35

Wavelength

- Refraction is wavelength-dependent
- Newton's experiment
- Usually ignored in graphics

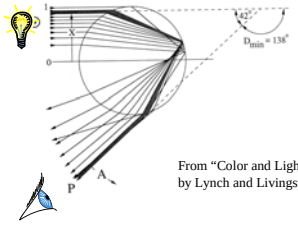
Pink Floyd, *The Dark Side of the Moon*

Pittoni, 1725, Allegory to Newton

Rainbow

- Refraction depends on wavelength
- Rainbow is caused by refraction+internal reflection+refraction
- Maximum for angle around 42 degrees

Digression



From "Color and Light in Nature" by Lynch and Livingstone

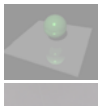
41

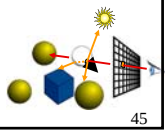
43

44

Overview of today

- Shadows
- Reflection
- Refraction
- Recursive Ray Tracing

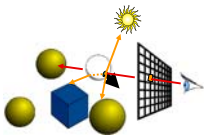




45

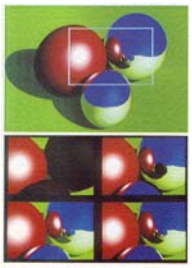
Recap: Ray Tracing

```
traceRay
   Intersect all objects
   Ambient shading
   For every light
      Shadow ray
      shading
   If mirror
      Trace reflected ray
   If transparent
      Trace transmitted ray
```



46

The depth of reflection



Avoiding infinite recursion

Stopping criteria:

- Recursion depth
  - Stop after a number of bounces
- Ray contribution
  - Stop if transparency/transmitted attenuation becomes too small

Usually do both

```
Color traceRay(ray)
    For every object ob
        ob->intersect(ray, hit, tmin);
    Color col=ambient*hit->getMaterial()->getDiffuse();
    For every light L
        If ( not castShadowRay( hit->getPoint(), L->getDir())
                col=col*hit->getMaterial()->shade
                    (ray, hit, L->getDir(), L->getColor());
    If (hit->getMaterial()->isMirror())
        Ray rayMirror (hit->getPoint(),
            getMirrorDir(ray->getDirection(), hit->getNormal()));
        Color=col+hit->getMaterial->getMirrorColor()
            *traceRay(rayMirror);
    If (hit->getMaterial()->isTransparent()
        Ray rayTransmitted(hit->getPoint(),
            getRefracDir(ray, hit->getNormal(),
            currentRefractionIndex, hit->Material-
            >getRefractionIndex());
        Color=col+hit->getMaterial->getTransmittedColor()
            *traceRay(rayTransmitted);
    Return col;
```


Recursion for reflection

0 recursion | 1 recursion | 2 recursions

Ray-Surface Intersection

- Implicit surfaces: $f(x, y, z) = 0$
  - Use a parametric representation for the ray:

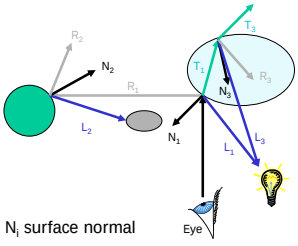
$$R(t) = O + tD$$
$$R_x(t) = O_x + tD_x$$
$$R_y(t) = O_y + tD_y$$
$$R_z(t) = O_z + tD_z$$

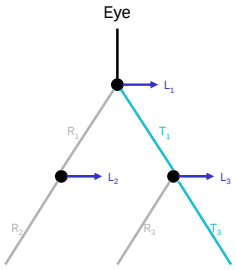
  - Substitute into the implicit equation:

$$f(O_x + tD_x, O_y + tD_y, O_z + tD_z) = 0$$

  - Solve the resulting equation
  - Examples: plane, sphere

The Ray Tree





N_i surface normal

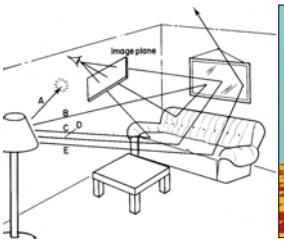
R_i reflected ray

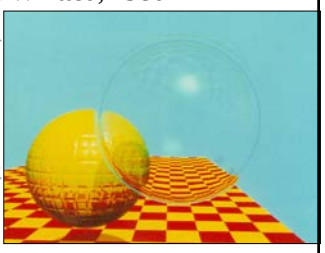
L_i shadow ray

T_i transmitted (refracted) ray

Ray Tracing History

- Ray Casting: Appel, 1968
- CSG and quadrics: Goldstein & Nagel 1971
- Recursive ray tracing: Whitted, 1980



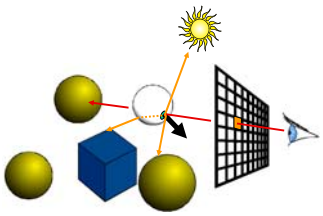


Mirror morphine This scene is composed entirely of spheres: the 40-sphere morphine molecule is tucked in the corner between two large mirrored balls and the yellow ground ball. The image was calculated at a resolution of 2048 × 2048 with 10 levels of reflections, 3 × 3 supersampling, and analytic penumbra calculations (not probabilistic methods), in 8 days of VAX 11/780 time. For a discussion of shadows and penumbrae, see Section 5.1. (Copyright © Paul Heckbert, NYIT, 1983)

57

Does Ray Tracing simulate physics?

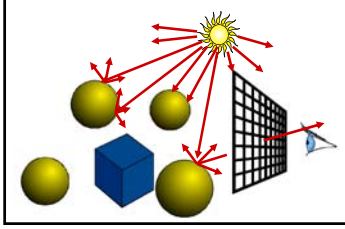
- Photons go from the light to the eye, not the other way
- What we do is backward ray tracing



58

Forward ray tracing

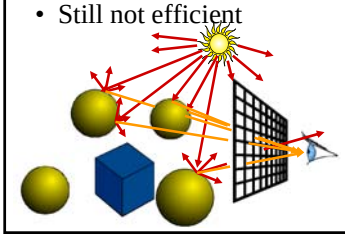
- Start from the light source
- But low probability to reach the eye
  - What can we do about it?



59

Forward ray tracing

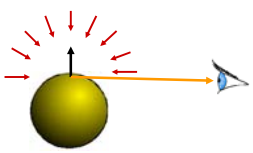
- Start from the light source
- But low probability to reach the eye
  - What can we do about it?
  - Always send a ray to the eye
- Still not efficient



60

The Rendering equation

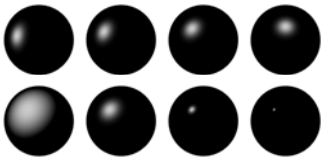
- Clean mathematical framework for light-transport simulation
- At each point, outgoing light in one direction is the integral of incoming light in all directions multiplied by reflectance property

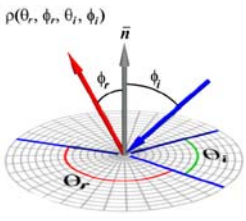


63

BRDF

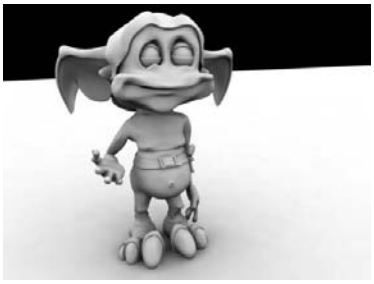
- Reflectance properties, shading and BRDF



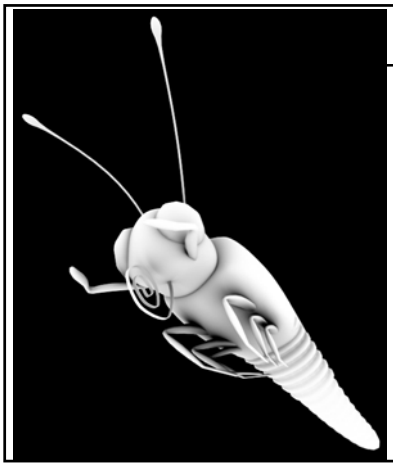


64

Ambient Occlusion

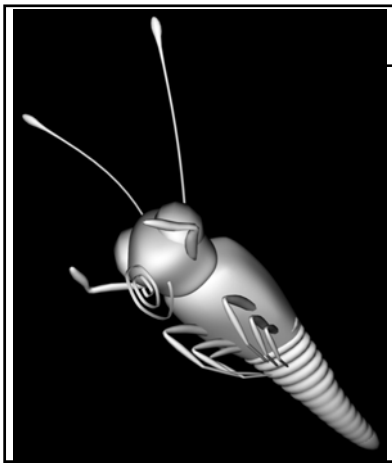


65



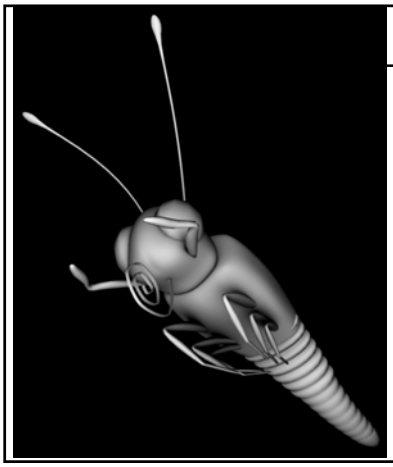
Ambient Occlusion

66



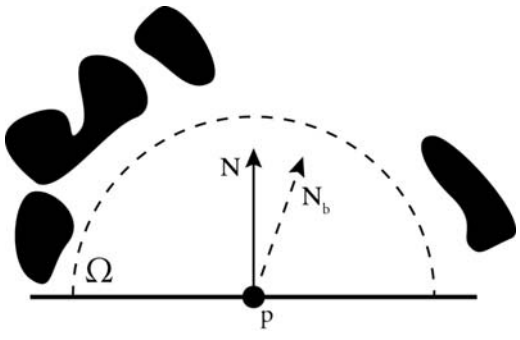
Diffuse Only

67



Diffuse and Ambient

68



69

70

71

72

73