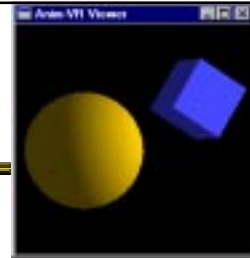


# Ray Tracing



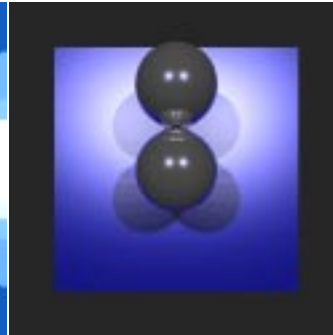
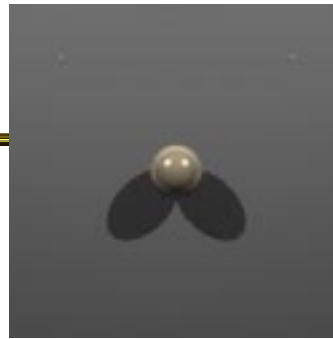
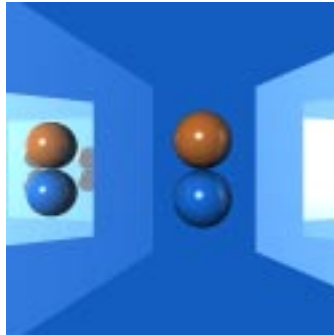
## Radically Different Approach



- Up to now:
  - Scan convert objects, computing color as we go
  - Can't easily do:
    - Shadows
    - Reflection
    - Accurate transparency
- Instead, let's think about light and the viewer's eye

## Model Light

- Follow it through the scene



## Forward Ray Tracing

- Lights emit photons
- Follow the photons
  - Trace paths (rays)
  - Bounce off objects
    - ┆ Reflect and refract, attenuate
  - When a ray enters eye
    - ┆ Calculate intersection with viewplane
    - ┆ Accumulate color in the pixel
- Expensive!



## Backward Ray Tracing

---



- Observation:
  - Only care about light that enters eye
- Trace those rays into the scene directly
  - For each pixel
    - Compute ray from eye through pixel
      - Intersect with objects
        - Compute color using closest object intersection
- "Ray Tracing" = backward ray tracing

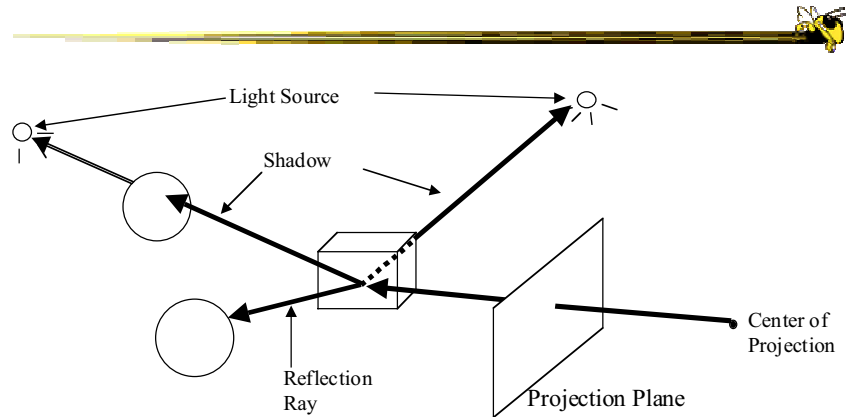
## Coloring Pixels

---



- Use illumination model to determine color at intersection of ray and object
  - Use the same model we've been using
  - Easily compute specular now!
- Can do proper transparency
  - Compute color of farther pixel
  - Combine it with transparent object
- Refraction, shadows?

## Reflection, Refraction and Shadows: Recursion




## Computing Intersections

- Handout: Chapter 2 of  
"An Introduction to Ray Tracing", Andrew Glassner
- Sphere/Ray Intersections (section 2)
  - Algebraic Solution (2.1)
  - Precision Problems (2.4)
- Polygon/Ray Intersections (section 3)
  - Ray/Plane Intersection (3.1)
  - Polygon Intersection (3.2)

## Sphere/Ray Intersections

---



### ■ Sphere

■ center  $Sc = [Xc, Yc, Zc]$

■ radius  $Sr$

■ surface points  $[Xs, Ys, Zs]$

$$\text{where } (Xs - Xc)^2 + (Ys - Yc)^2 + (Zs - Zc)^2 = Sr^2$$

### ■ Ray


■  $X = X0 + Xd * t$

■  $Y = Y0 + Yd * t$

■  $Z = Z0 + Zd * t$

## Sphere/Ray Intersections

---



### ■ Substitute Ray eq's into Sphere eq:

$$(X0 + Xd * t - Xc)^2 + (Y0 + Yd * t - Yc)^2 + (Z0 + Zd * t - Zc)^2 = Sr^2$$

### ■ Simplify to get $At^2 + Bt + C = 0$ , where

$$A = Xd^2 + Yd^2 + Zd^2$$

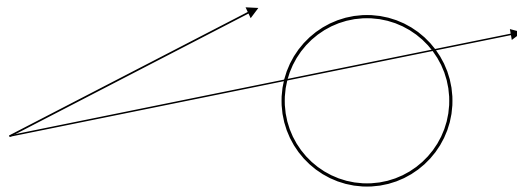
$$B = 2(Xd(X0 - Xc) + Yd(Y0 - Yc) + Zd(Z0 - Zc))$$

$$C = (X0 - Xc)^2 + (Y0 - Yc)^2 + (Z0 - Zc)^2 - Sr^2$$

## Sphere/Ray Intersections



- Solve using quadratic formula
- Discriminant (part under  $\text{sqrt}()$ )
  - Negative: no intersection
  - Positive: two solutions,  $t_0$  and  $t_1$



## Sphere/Ray Intersections



- Intersection point  $r_i = [X_i, Y_i, Z_i]$ 
  - Substitute  $t_1$  or  $t_0$  into ray eq's
- Surface normal
  - $r_n = [X_n, Y_n, Z_n]$   
=  $(r_i - S_c)/S_r$

## Illumination of a point

---



- Start with our illumination equation:

$$I = I_a k_a O_d + \sum_{1 \leq i \leq m} f_{att,i} I_{p,i} [k_d O_d (N \cdot L_i) + k_s (R_i \cdot V)^n]$$

- Change  $R_i \cdot V$  to  $N \cdot H_i$ 
  - See Section 16.1.4, p 731 for why
- Shadows and reflection?

## Illumination of a point: Shadows

---



- If  $L_i$  hits an object before the light, we are in shadow (16.4, p 745)
  - Add  $S_i$  component before  $f_{att,i}$ 
    - 0 if light  $i$  is blocked
    - 1 if light  $i$  is not blocked
    - Could be 0..1 if blocked by transparent object

## Illumination of a point: Reflection



- Compute illumination of reflected ray  $I_r$ 
  - Add  $k_s I_r$
  - Attenuate for distance

## Illumination of a point: Transparency



- Compute illumination of transmitted ray  $I_t$ 
  - May refract  $V$  to get  $I_t$
  - Add  $k_t I_t$ 
    - $k_t$  is the transmission coefficient
  - Attenuate for distance