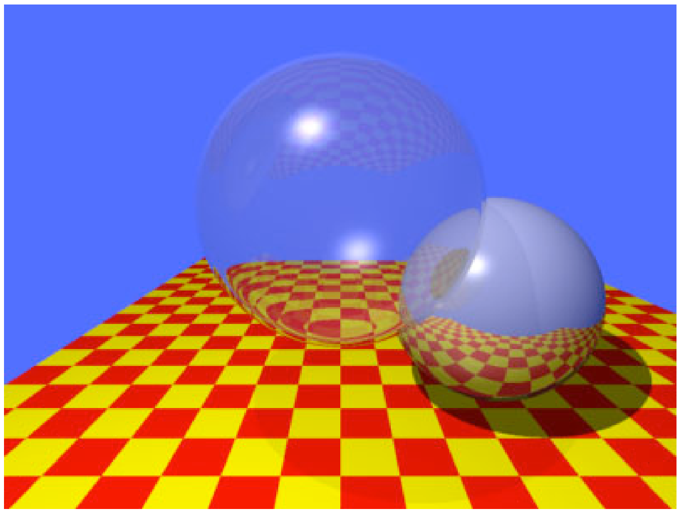
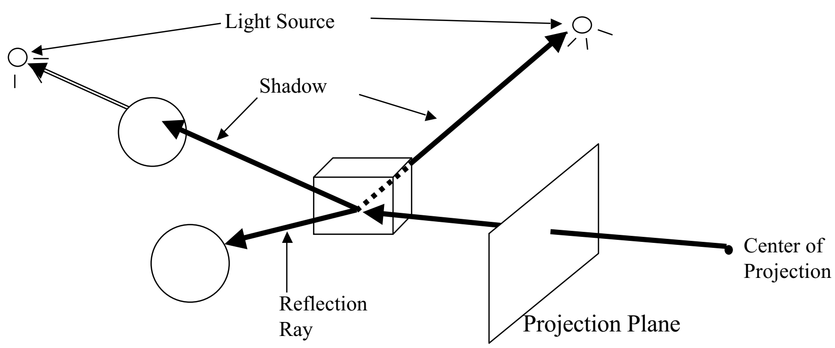
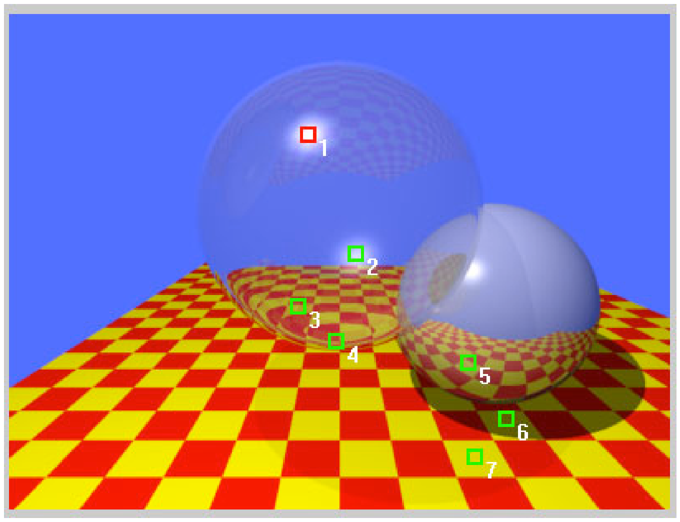
Ray Tracing



Recall basic idea



The Adventures of 7 Rays



Illumination of a point

- Start with our illumination equation:

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

- Change $R_i \cdot V$ to $N \cdot H_i$
  - See p 178 for why
- For A5, drop k_a
  - joogl doesn't use it, so scene is lit assuming k_a=1
- Shadows and reflection?

Illumination of a point: Shadows

- If L_i hits an object before the light, we are in shadow
- Add S_i component before f_{atti}:
  - 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

Illumination of a point

- Final illumination equation:

$$I = I_a O_d + k_s I_r + k_t I_t + \sum_{1 \le 0 \le m} S_i f_{att\,i} I_{pi} [\, k_d O_d (N \cdot L_i) + k_s (N \cdot H_i)^n]$$

Computing Intersections

Sphere/Ray Intersections

- Sphere
  - center Sc = [Xc,Yc,Zc]
  - radius Sr
  - surface points [Xs,Ys,Zs]
    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

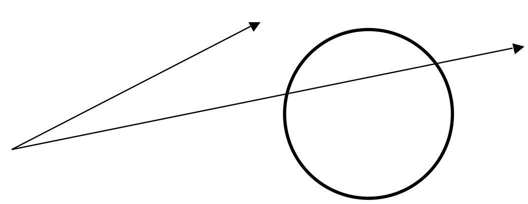
  $A = Xd^2 + Yd^2 + Zd^2$

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

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

Sphere/Ray Intersections

- Solve using quadratic formula
- Discriminant (part under sqrt())
  - Negative: no intersection
  - Positive: two solutions, t0 and t1



Sphere/Ray Intersections

- Intersection point ri = [Xi, Yi, Zi]
  - Substitute t1 or t0 into ray eq's
- Surface normal
  - rn = [Xn, Yn, Zn]
    = (ri - Sc)/Sr

Ray/Plane Intersection

- Ray: $R_0 = [X_0, Y_0, Z_0]$
  $R_d = [X_d, Y_d, Z_d]$ (normalized)
  where $R(t) = R_0 + R_d t$, $t>0$
- Plane: $Ax + By + Cz + D = 0$
  $P_n = [A, B, C]$
- for any [x,y,z],
  Ax + By + Cz + D = distance to plane

Ray/Plane intersection

- Substitute Ray eq. into Plane eq.

  $A(X_0 + X_d t) + B(Y_0 + Y_d t) + C(Z_0 + Z_d t) + D = 0$
- Solve for t

  $t = -(AX_0 + BY_0 + CZ_0 + C)/(AX_d + BY_d + CZ_d)$

  $= -(P_n \cdot R_0 + D)/(P_n \cdot R_d)$
- $v_d = AX_d + BY_d + CZ_d$
  - if 0, parallel
  - if > 0, plane facing away

Ray/Plane intersection

- $v_d = AX_d + BY_d + CZ_d$
  - if 0, parallel
  - if > 0, plane facing away
- Calculate $t = -(AX_0 + BY_0 + CZ_0 + C)/v_d$
  - if t < 0, plane intersects behind viewer
- Use t in ray eq. to compute intersection

Ray/Polygon Intersection

- Compute intersection with polygon plane
- Jordan Curve Theorem
  - Line from intersection point of ray/plane
  - Count number of poly edges
    - Even: outside
    - Odd: inside

Ray/Polygon Intersection

- Practical details
  - Project polygon onto 2D plane
    - Normal is [A B C], choose largest abs. Value
    - Area changes, *topology doesn't*
    - UV coordinates
  - Translate result so ray intersection point is at origin
  - Use +U axis as line to intersect
    - Look at lines that intersect it