CSE 681

Ray Tracing and Shadows

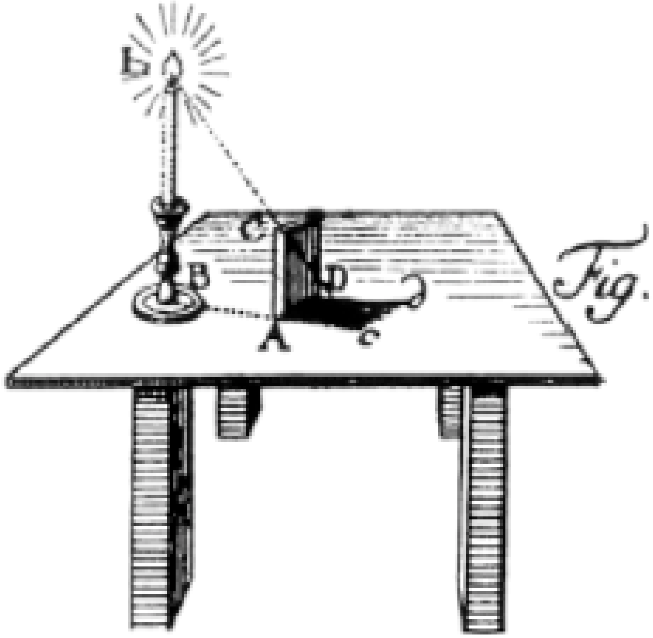
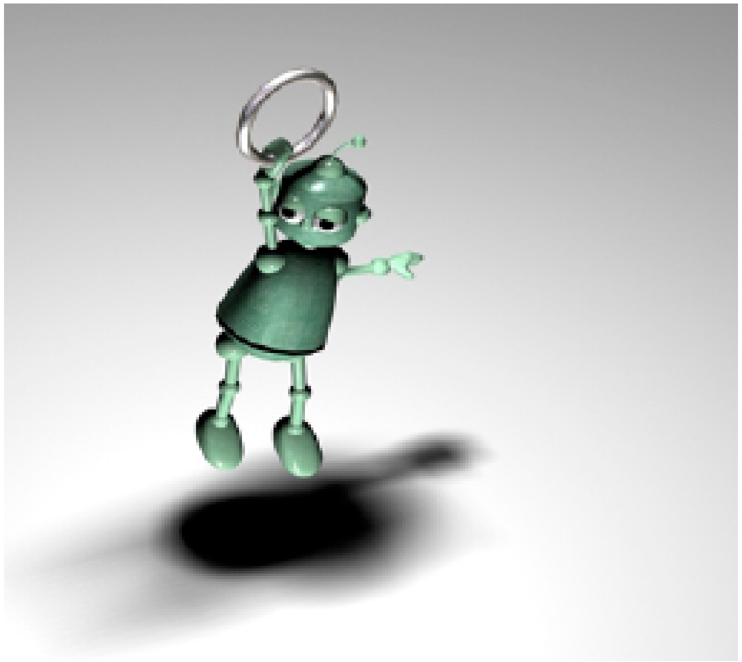
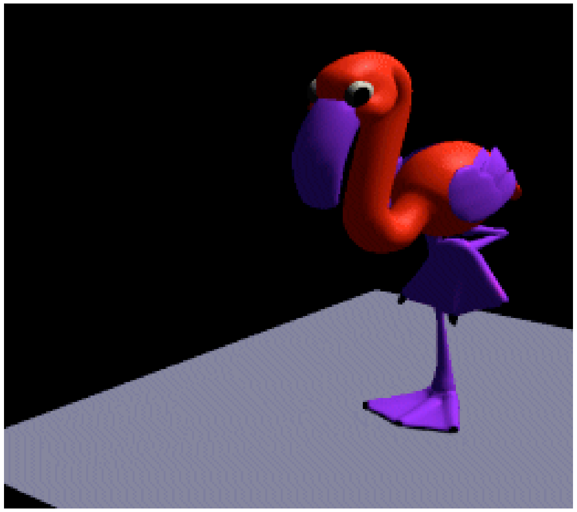


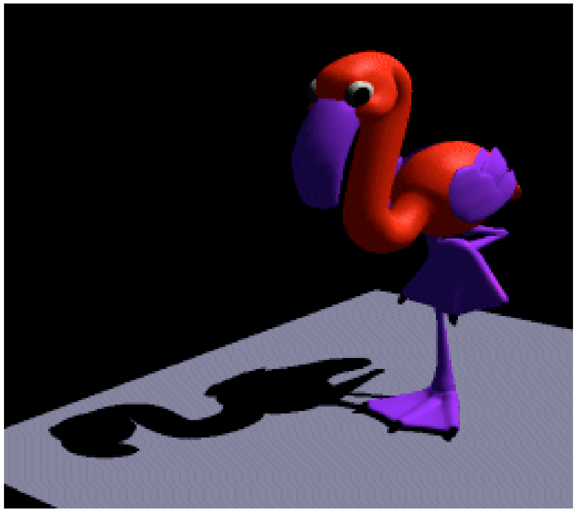
Fig. XV.



Why Shadows?

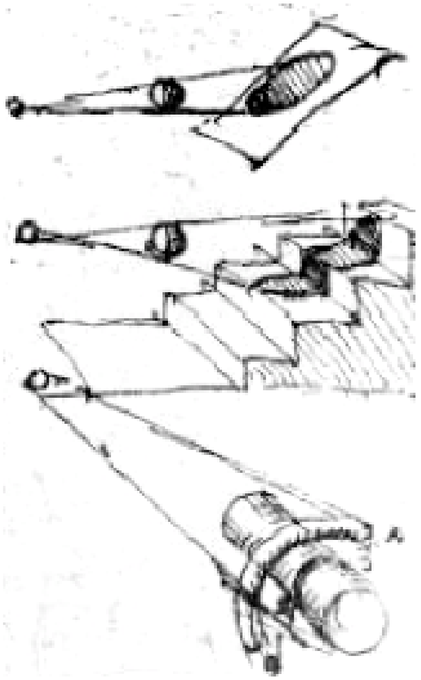
- Makes 3D Graphics more believable
- Provides additional cues for the shapes and relative positions of objects in 3D



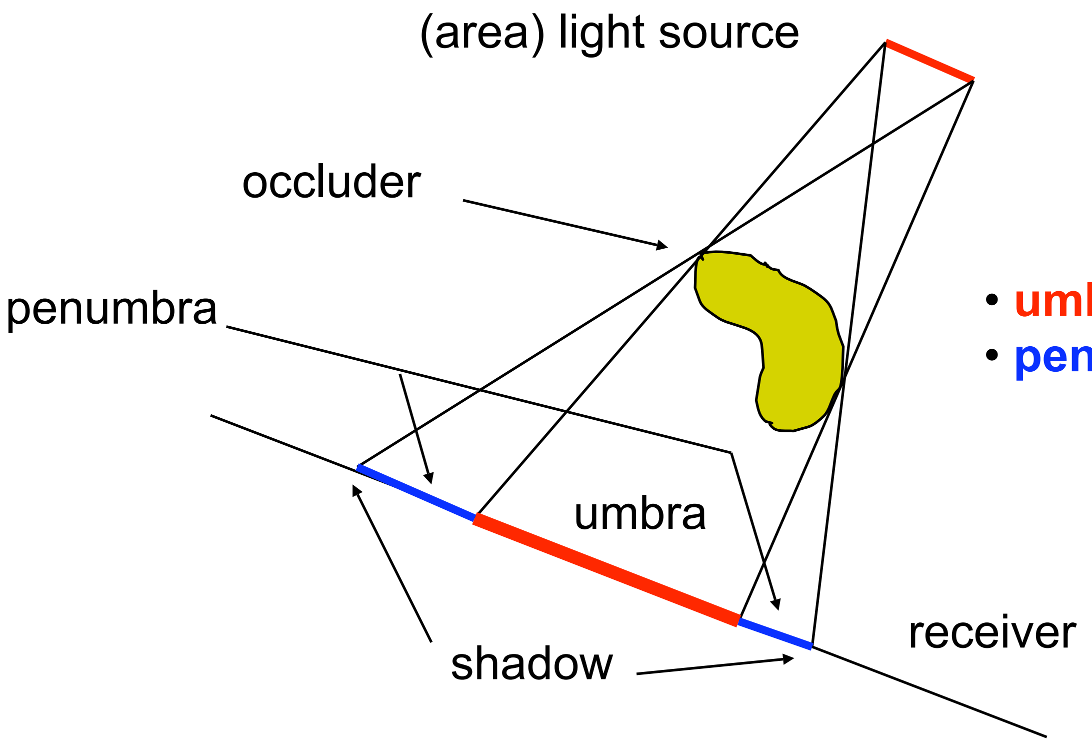


What is shadow?

- Shadow: comparative darkness given by shelter from direct light; patch of shade projected by a body intercepting light



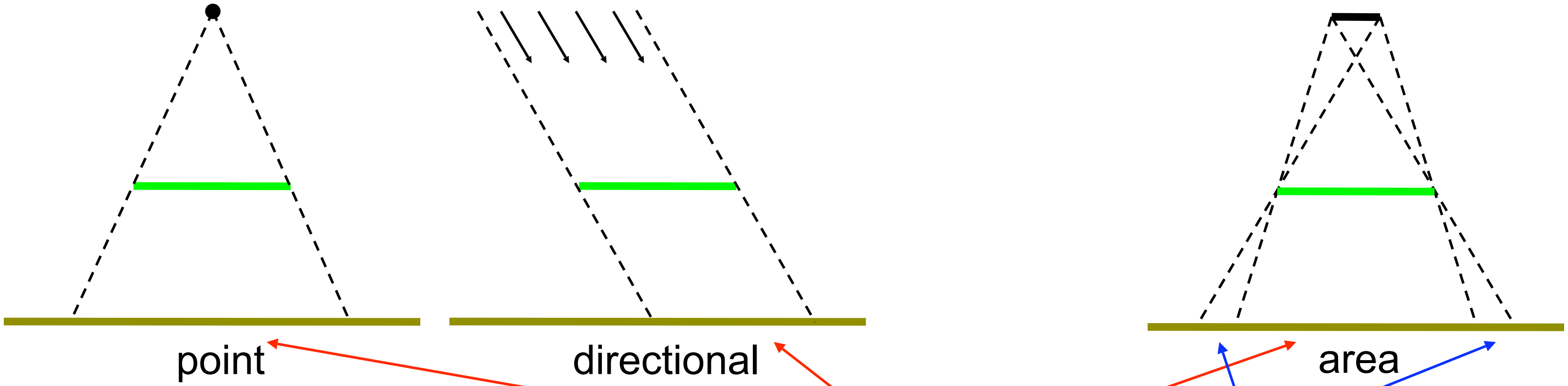
Terminology



- **umbra** – fully shadowed region
- **penumbra** – partially shadowed region

“Hard” and “Soft” Shadows

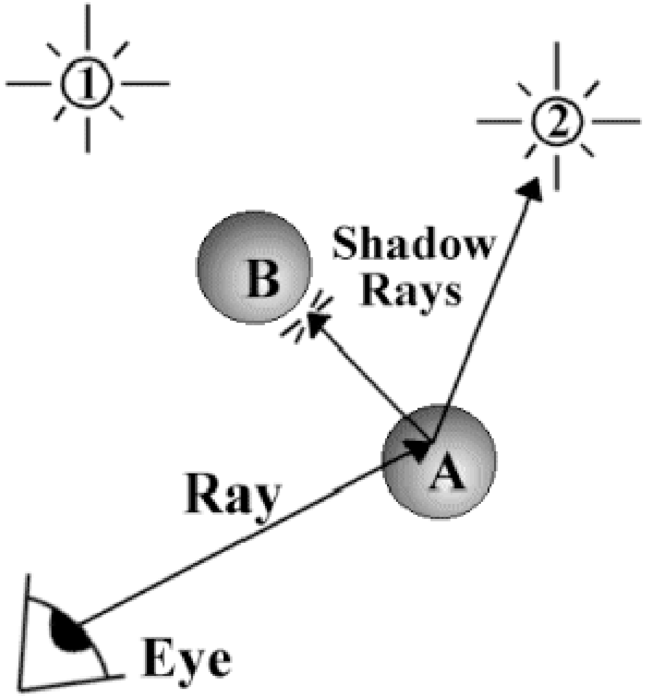
- Depends on the type of light sources
  - Point or Directional (“Hard Shadows”, umbra)



  - Area (“Soft Shadows”, *umbra, penumbra*), more difficult problem

Shadows in Ray Tracing

- Cast ray to light (*shadow rays*)
- Surface point in shadow if the shadow rays hits an occluder object.
- How do we add shadows in ray tracing?

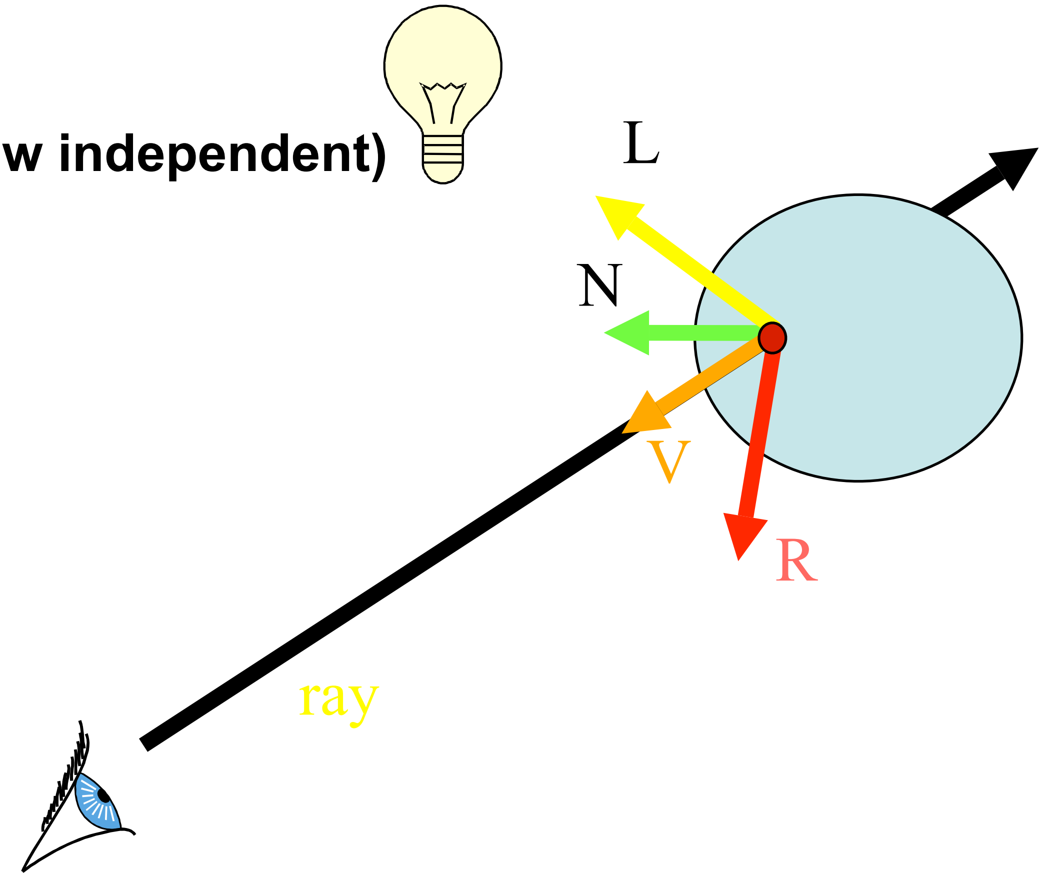


Quick Review: Phong Illumination

Ambient

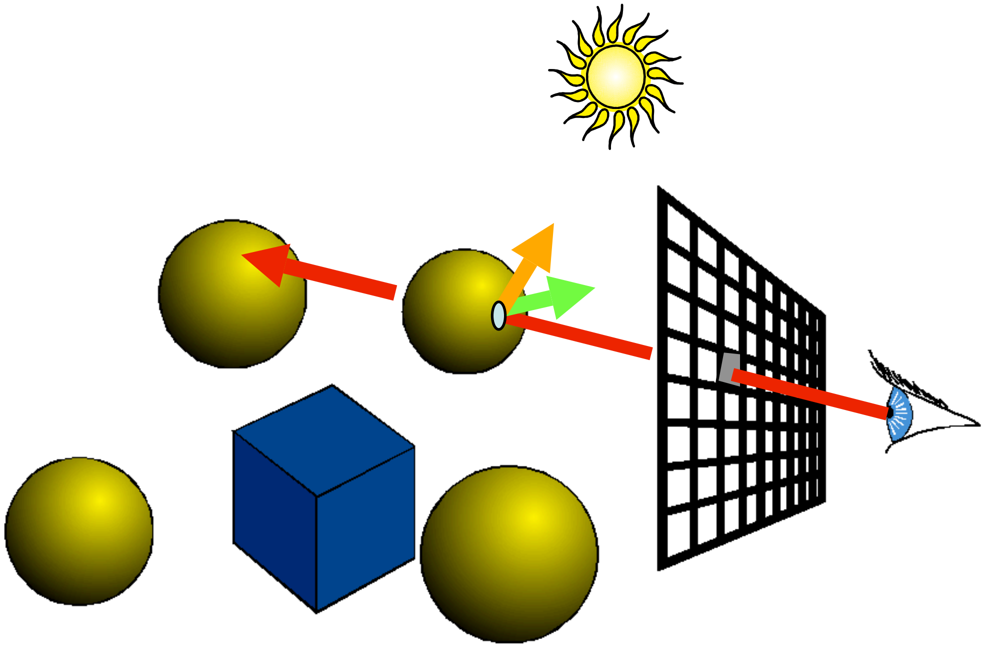
Diffuse **(view independent)**

Specular



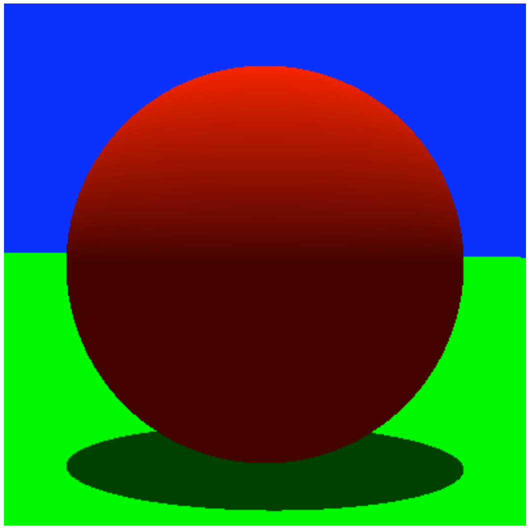
Phong Illumination

```
Color shade( ray )
{
        c = background  color;


        intersectFlag = FALSE;
        for each object
        intersectFlag = intersect ( ray, p );


        if intersectFlag is TRUE
             c = ambient;
             for each light source
                  compute reflective ray R (or H);
                  c += diffuse;
                  c += specular components;

        return c;
}
```



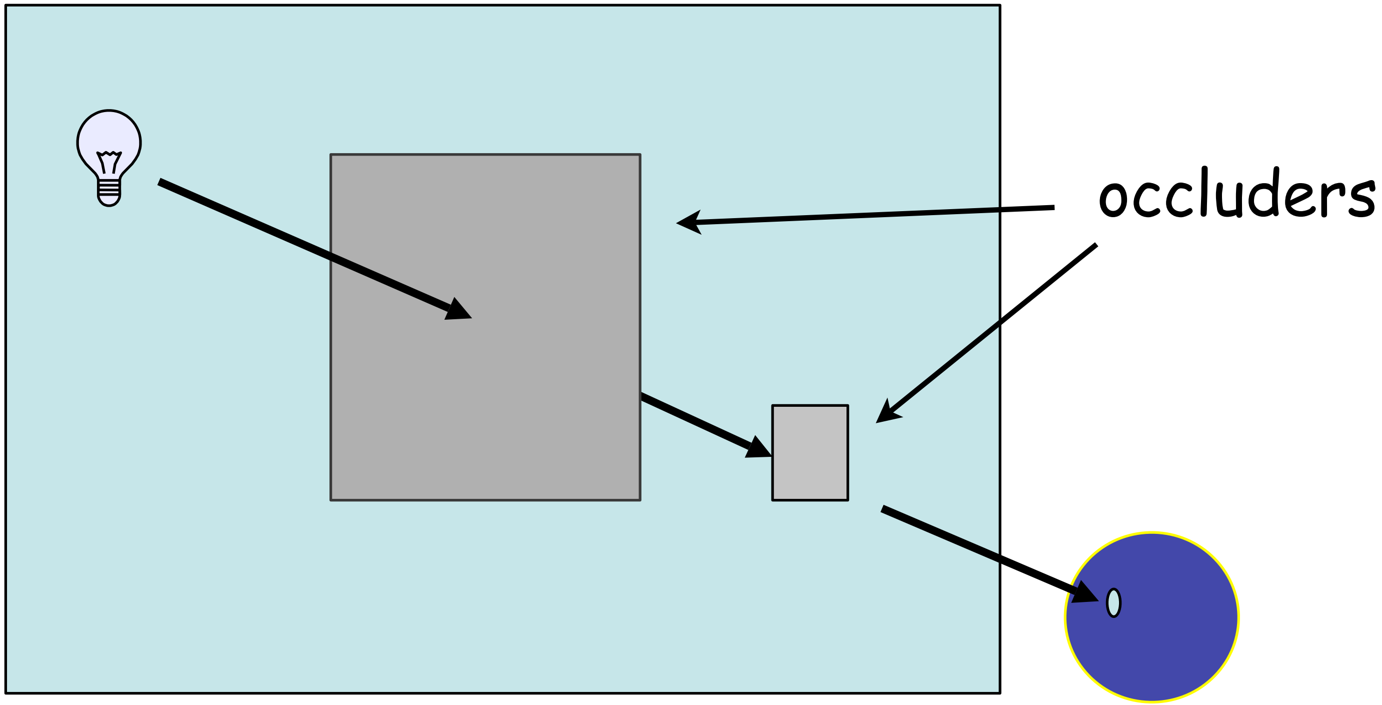
Shadows

- A ray-object intersection point is in shadow if an object occludes it from a light source
- Shoot a ray from the point to each light source and detect occluders



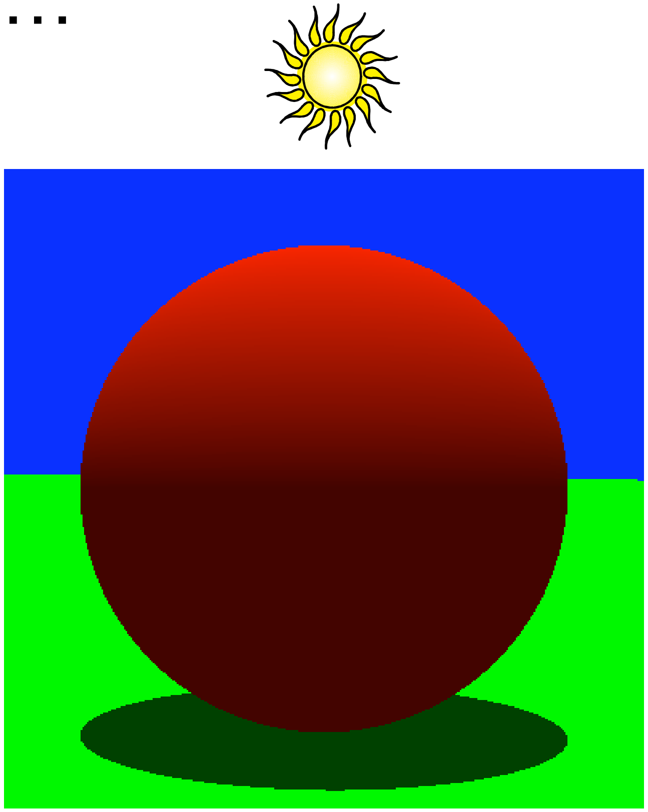
Shadows

- Is the light ray blocked from reaching the ray-object intersection point



Shading a Point In Shadow

- Assume Phong illumination …

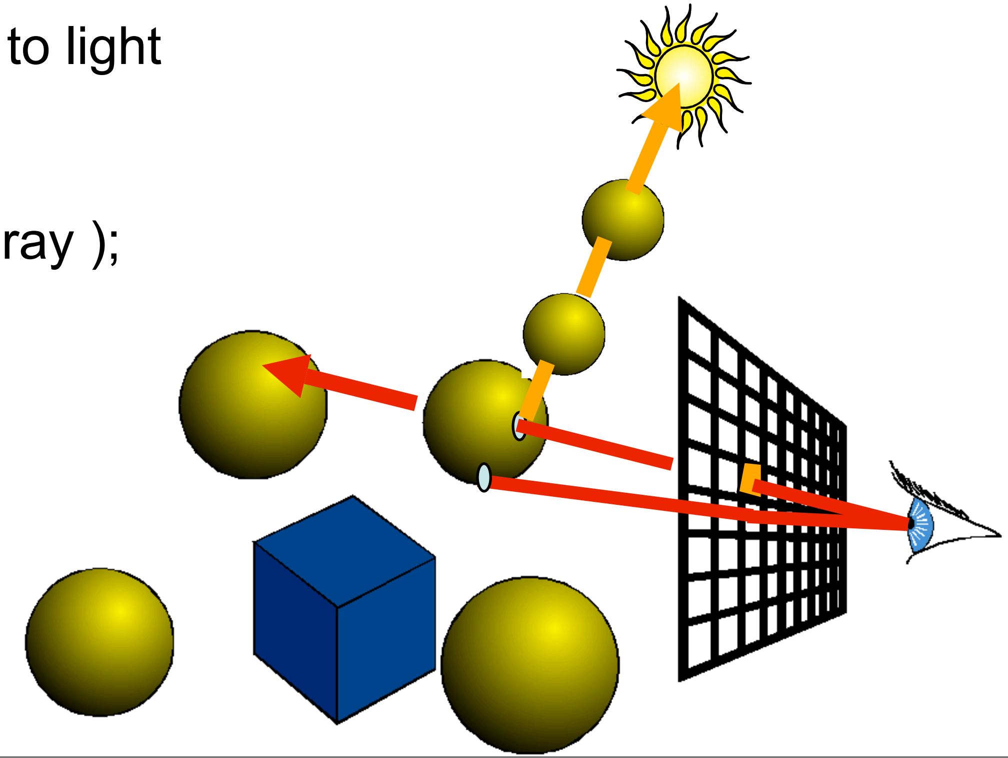


- Ambient?
  - Unaffected by a shadow
- Diffuse?
  - Turn off
- Specular?
  - Turn off

Pseudocode:

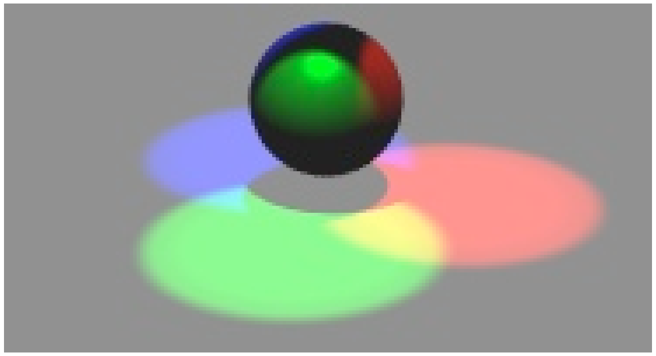
Optimization:
Stop at very first object intersection
Don't need closest intersection!!!

```
for each light source
        inShadow = FALSE;
        ray = intersection point p to light
           source;
        for each object
           inShadow = intersect ( ray );
           if inShadow is TRUE
                 break out of loop
return inShadow;
```



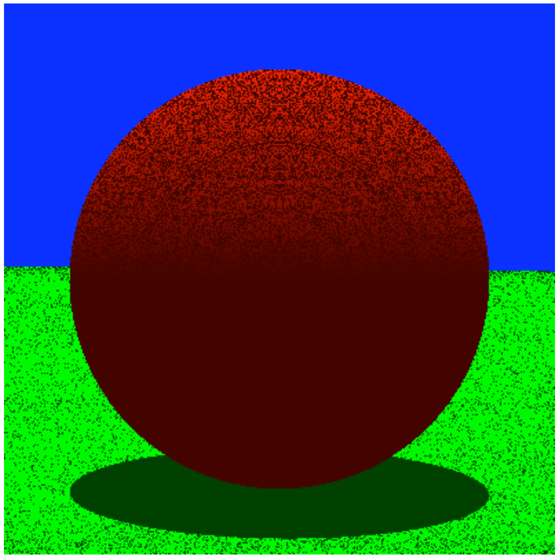
Shadows With Phong Illumination

```
Color shade( ray )
{
    c = background  color;


    intersectFlag = FALSE;
    for each object
            intersectFlag = intersect ( ray, p );


    if intersectFlag is TRUE
        c = ambient;
        shadowFlag = intersectShadowRay ( p );
        if shadowFlag is FALSE
            compute reflective ray R (or H);
            c += diffuse;
            c += specular components;

    return c;
}
```



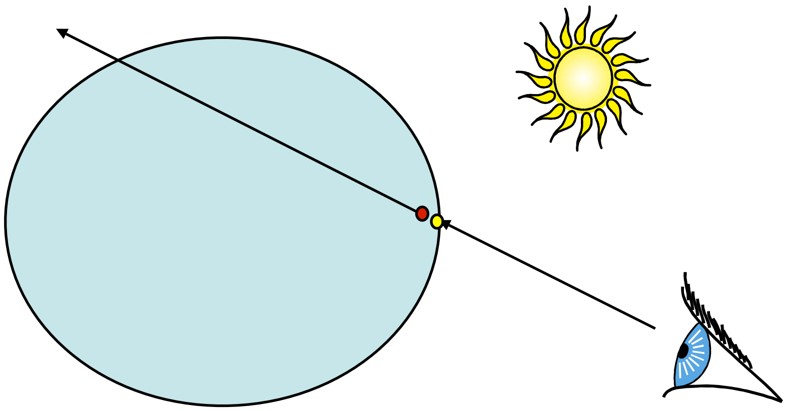
What!!??

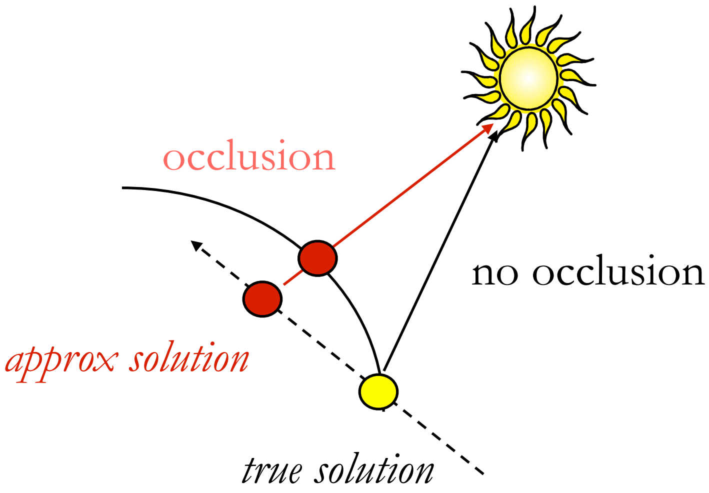
```
Color shade( ray )
{
    c = background  color;

    intersectFlag = FALSE;
    for each object
        intersectFlag = intersect ( ray, p );

    if intersectFlag is TRUE
        c = ambient;
        shadowFlag = intersectShadowRay ( p );
        if shadowFlag is FALSE
            compute reflective ray R (or H);
            c += diffuse;
            c += specular components;

    return c;
}
```



Problem: Self-Shadowing

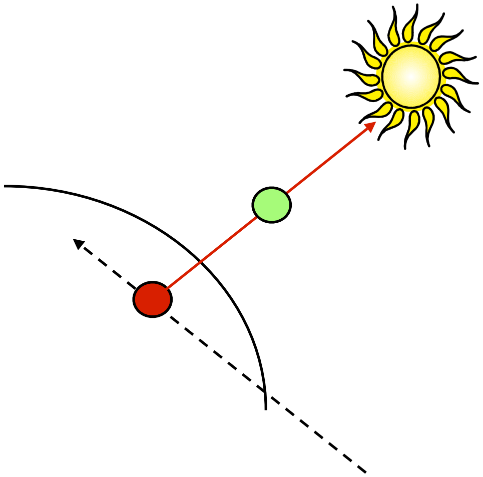
- Precision problems
- Your approximation to the ray-object intersection is off by a small amount ... sometimes





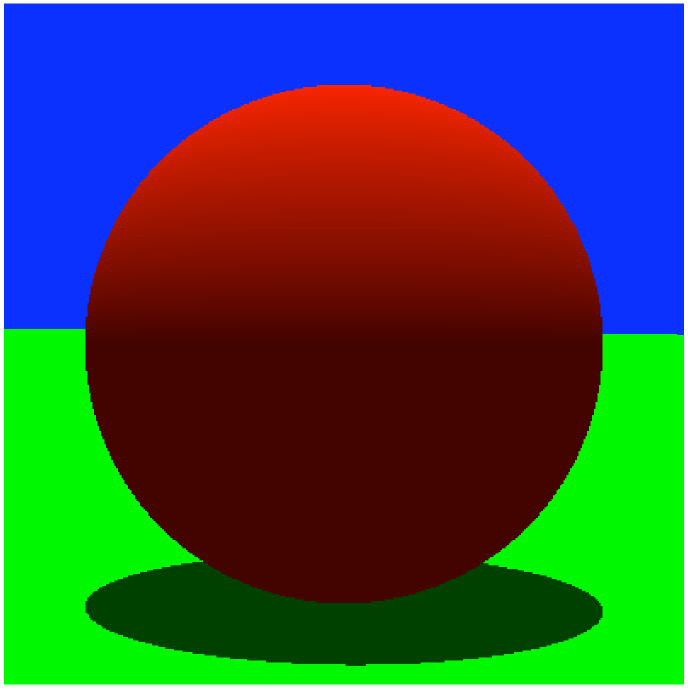
A Solution

- Move our approximate solution (intersection point) towards the light by some small amount $\varepsilon > 0$ so that our point is outside the object
- The value ε is pre-chosen to be some small number close to zero

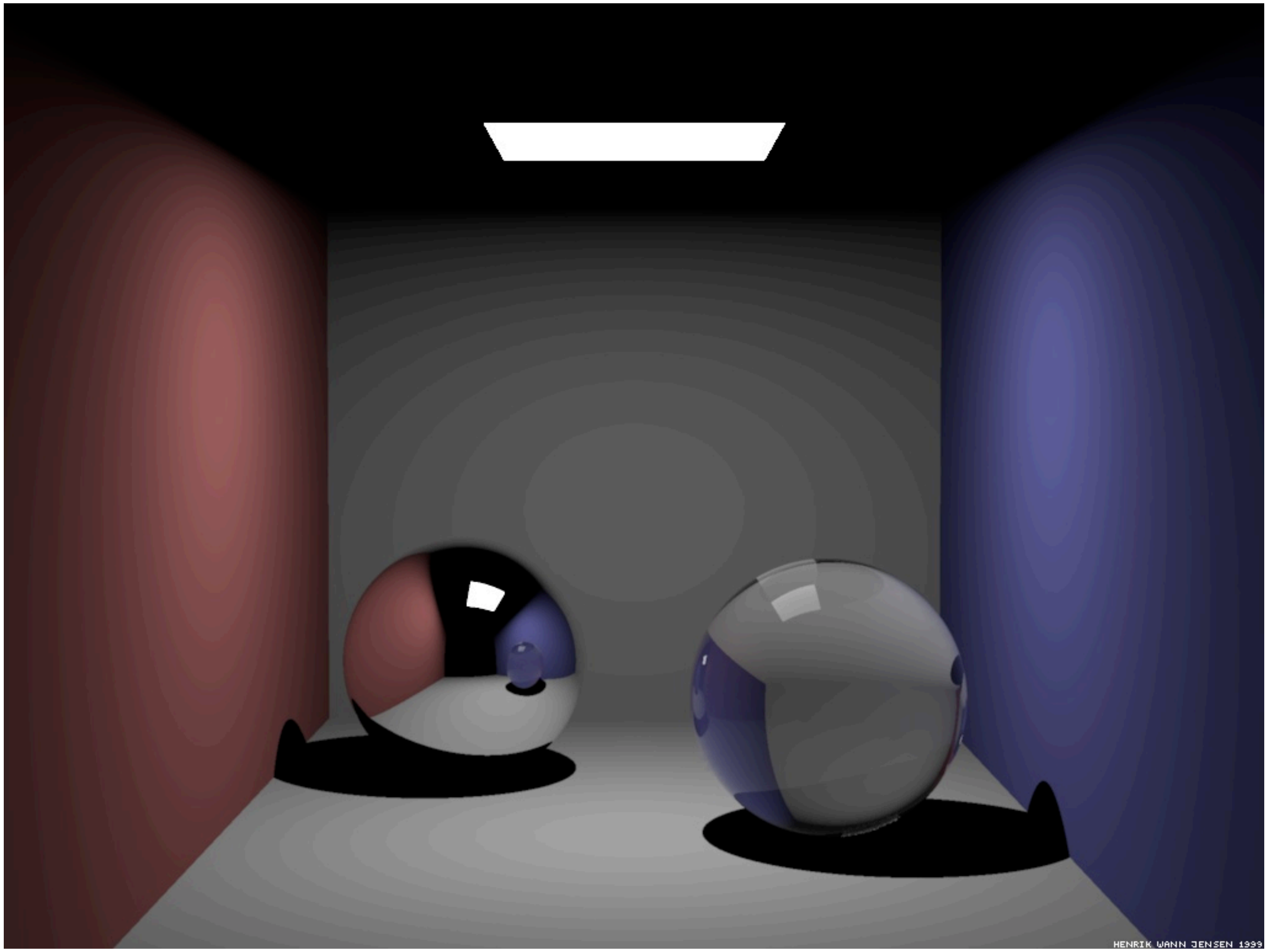


Pseudocode: IntersectShadowRay

```
for each light source
    if face is a backface wrt light
      source
        inShadow = TRUE;
    else
        inShadow = FALSE;
        $p = p + \varepsilon L$ // L is the light ray
        ray = intersection point p to light
          source;
        for each object
            inShadow = intersect ( ray );
            if inShadow is TRUE
                    break out of loop;
return inShadow;
```

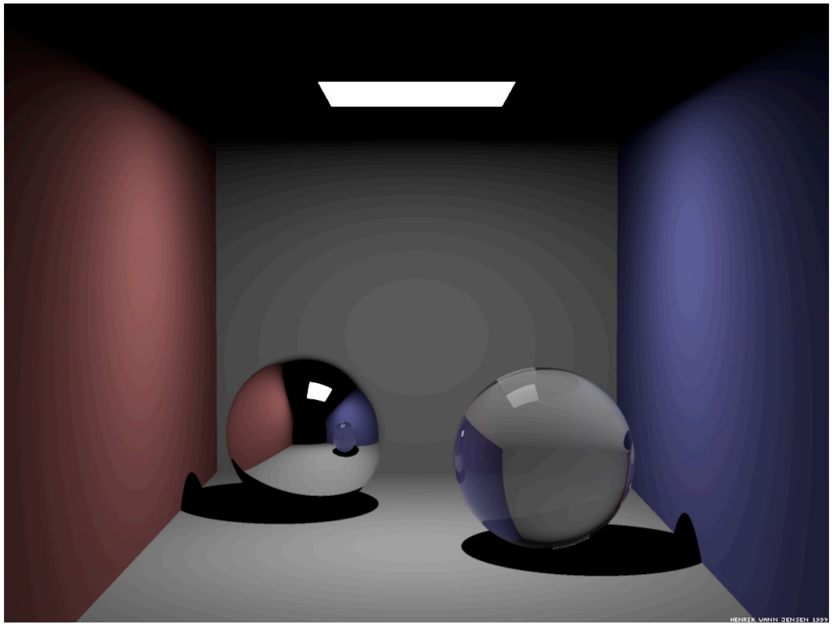


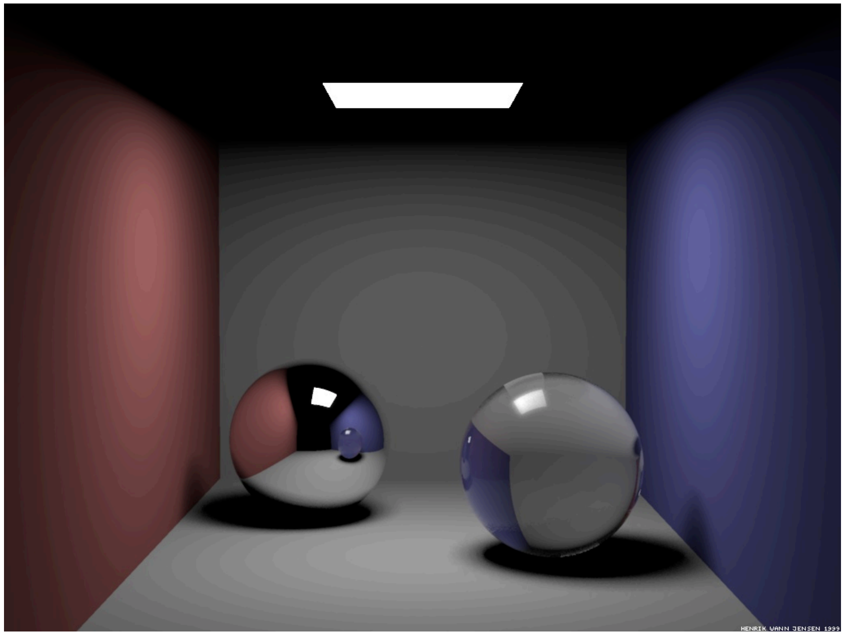
Cool Example



Soft Shadows

- Hard shadows (left) vs soft shadows (right)





Soft Shadows

- Hard shadows
  - – Assume an infinitely small (point) light source
- Soft shadows
  - – Umbra (invisible) and Penumbra (fuzzy looking drop off)
  - – Assumes an area light source
  - – Treat the light as many point lights
    - Expensive!!!