Shaders

Recall: Lighting Equation

- Multiplying the BRDF by an incoming irradiance gives the outgoing radiance

$$dL_{o\ due\ to\ i}(\omega_i, \omega_o) = BRDF(\omega_i, \omega_o) dE_i(\omega_i)$$

- For more realistic lighting, we bounce light all around the scene, and it is tedious to convert between irradiance and radiance, so we use $dE = Ld\omega \cos\theta$ to obtain:

$$dL_{o\ due\ to\ i}(\omega_i, \omega_o) = BRDF(\omega_i, \omega_o) L_i d\omega_i \cos\theta_i$$

- The outgoing radiance considering the light coming from all incoming directions is:

$$L_o(\omega_o) = \int_{i \in in} BRDF(\omega_i, \omega_o) L_i \cos\theta_i \, d\omega_i$$

Recall: Area Lights

- Light power is emitted per unit area (not from a single point)
- The emitted light goes in various directions, or solid angles

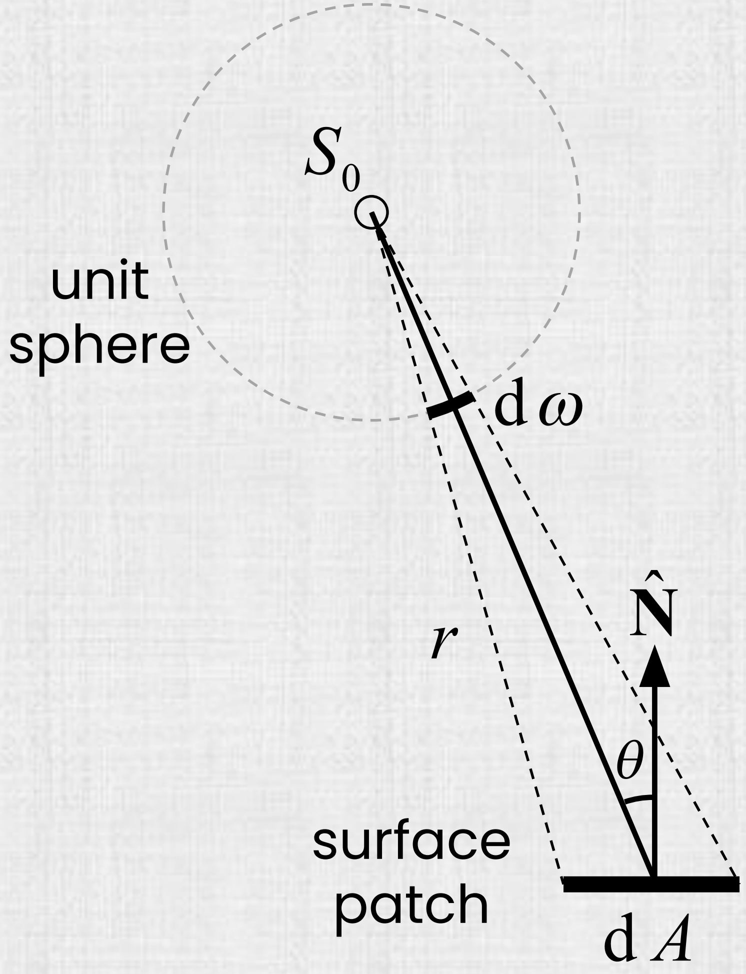
- Approximate an area light by breaking it up into small area chunks
- Each area chunk emits light into each of the solid angle directions
  - i.e. radiant intensity per area chunk
- Each direction has a cosine term similar to irradiance

- Radiance – radiant intensity per area chunk

$$L = \frac{dI}{dA\cos\theta} \quad \left(= \frac{d^2\Phi}{d\omega dA\cos\theta} = \frac{dE}{d\omega\cos\theta} \right)$$

Recall: Solid Angle vs. Area

- The relationship between solid angle and cross-sectional area is $d\omega = \frac{dA_{sphere}}{r^2} = \frac{dA\, cos\theta}{r^2}$, since the area of the orthogonal cross section is $dA\, cos\theta$ (see previous slide)



Point Lights

- Assume incoming light only comes from a single point light source from direction ω_i
- Then the BRDF and the cosine terms are approximately constant:

$$L_o(\omega_o) = BRDF(\omega_{light}, \omega_o) \cos\theta_{light} \int_{i \in in} L_i d\omega_i$$

- Since $L = \frac{dI}{dA\cos\theta}$ and $d\omega = \frac{dA\,cos\theta}{r^2}$, the integral becomes $\int_{i \in in} \frac{dI}{r^2} = \frac{I}{r^2}$
- Assume all objects are approximately the same distance from the point light (e.g. the sun), then r is approximately constant and can be folded into I_{light} to get $\hat{I}_{light}$:

$$L_o(\omega_o) = BRDF(\omega_{light}, \omega_o) \cos\theta_{light}\, \hat{I}_{light}$$

- In summary, for each channel (R,G,B), sum over all the point lights:

$$L_o(\omega_o) = \sum_{j=1}^{\#lights} BRDF(\omega_j, \omega_o) \cos\theta_j\, \hat{I}_j$$

Point Lights

- It is good to understand what is lost in such an approximation, because it affects the image

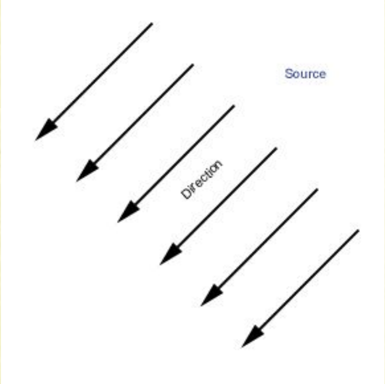
- All the lighting from other objects in the scene has been turned off, so the scene gets darker
- Surfaces that are not visible from a point light source are completely black
- Shadows have harsh boundaries (unlike the soft shadows obtained from area lights)
- Objects closer to a light source (e.g. in indoor scenes) will not be brighter than those father away (since the variance with radius has been removed)
- Etc.

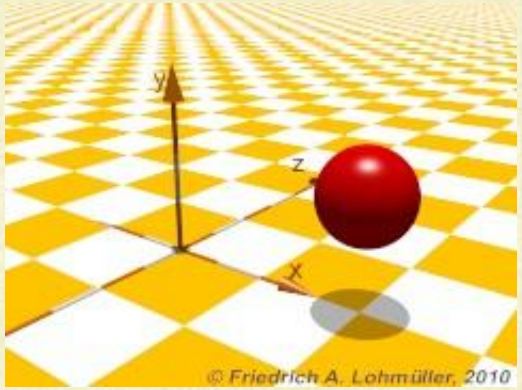
- The key to using strong (or ”hacky”) approximations successfully/effectively is in understanding what is lost, so that it can be modeled back in when necessary

Simple Light Types

Directional Light

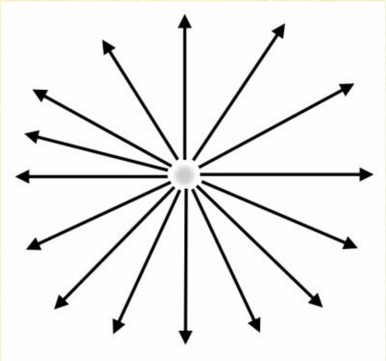
- Always use the the same incoming ray direction
- Good approximation to sunlight, since the sun is far away and rays of sunlight are approximately parallel

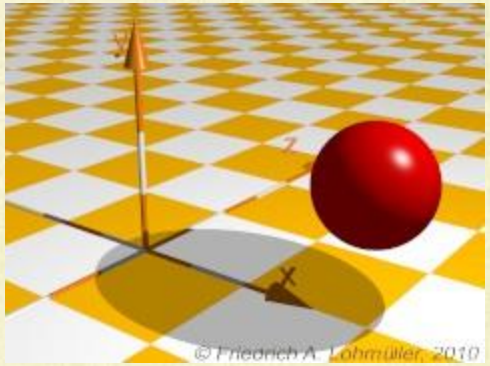




Point Light

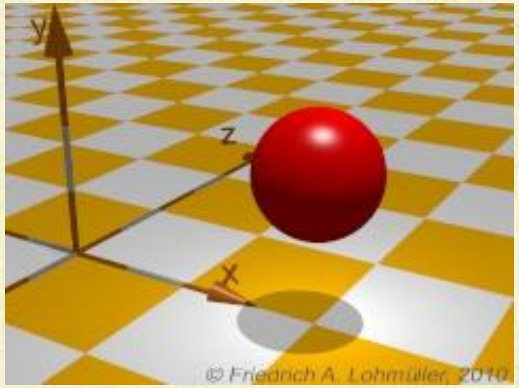
- Light emitted from a single point in space, outwards in every direction





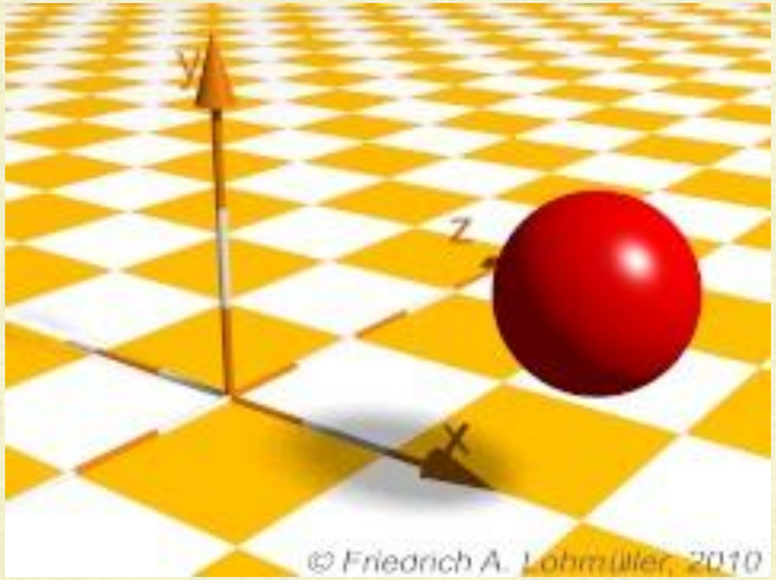
Spotlight

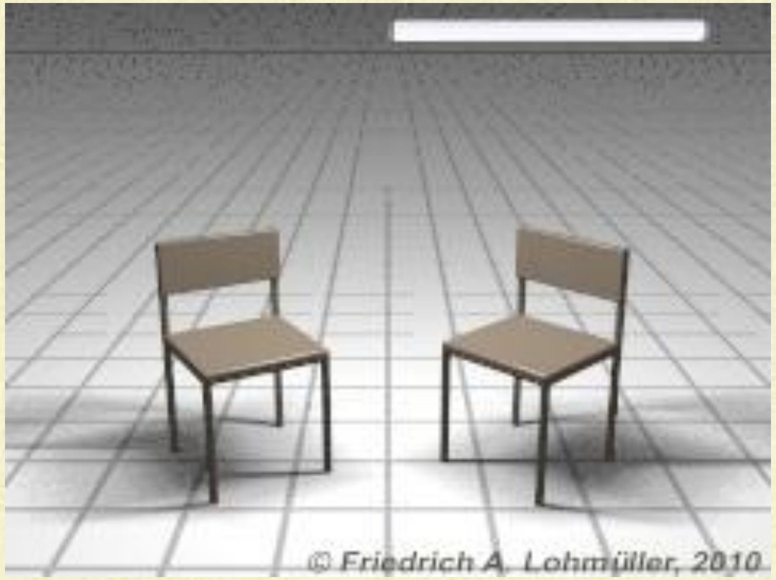
- Angular subset of a point light
- Dot product outward directions with a central direction and prune if larger than some angle

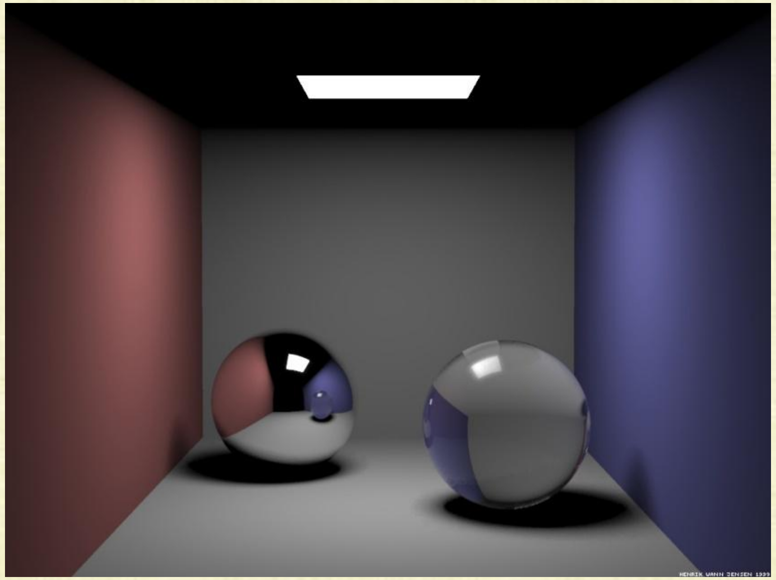


Area Light

- Light emitted from a surface (with objects behind the surface not illuminated)
- Can treat as a large collection of point lights on the surface (setting their strength to be the total area light strength divided by the number of points)
- Creates softer shadows





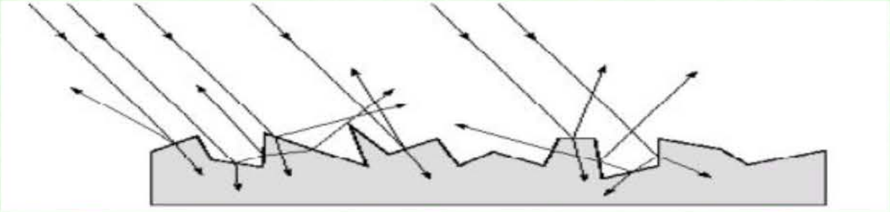


Volume Light

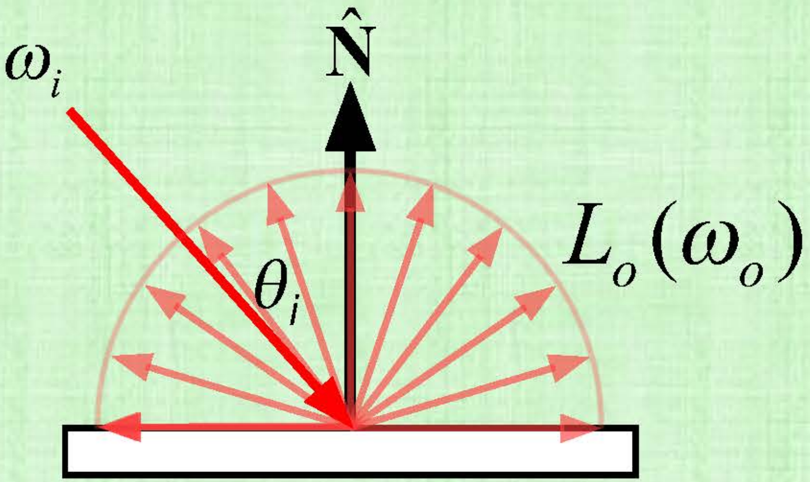
- Can treat as a large collection of point lights in a volumetric region

Diffuse Materials

- Assume a surface reflects light equally in all directions, independently of the incoming direction
- This can happen with a rough surface, with many tiny microfacets randomly reflecting incoming light outwards in every possible direction:

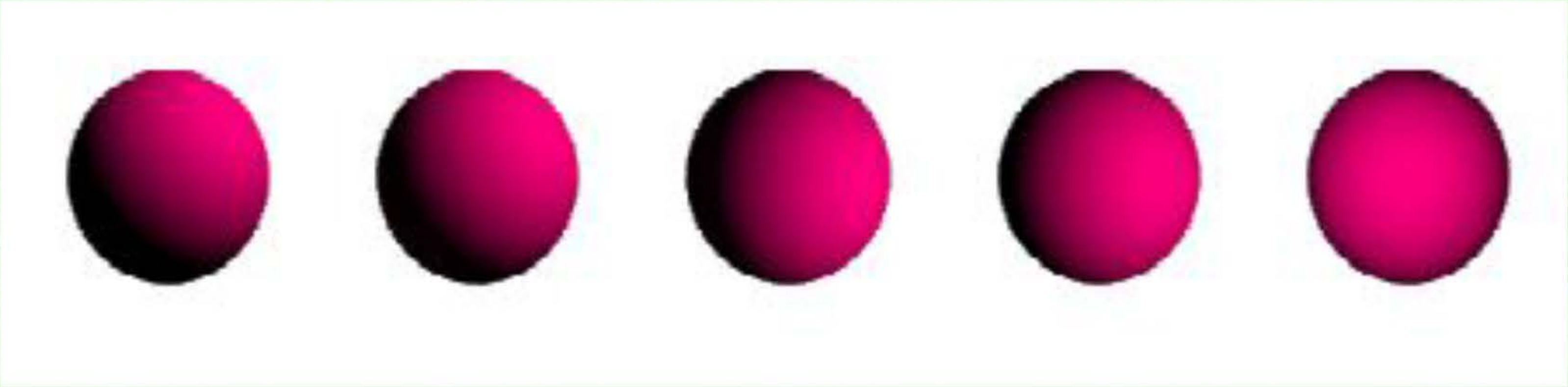


- Then, the BRDF no longer depends on incoming/outgoing directions, and is simply a constant, i.e. $BRDF(\omega_i, \omega_o) = k_d$ and $L_o = k_d \cos\theta_{light}\, \hat{I}_{light} = k_d\, \hat{I}_{\text{light}} \max\left(0, -\omega_{light} \cdot \hat{N}\right)$



Diffuse Materials

- Shading does depend on the position of the light source, because of the cosine term
- Shading does not depend on the position of the viewer/camera
- Good for diffuse/dull/matte surfaces, such as chalk



- Don't forget (R,G,B): an object with (diffuse) color $\left(k_{d,R}, k_{d,G}, k_{d,B}\right)$ being hit by a light with color $\left(\hat{I}_R, \hat{I}_G, \hat{I}_B\right)$ results in:

$$\left(L_{o,R}, L_{o,G}, L_{o,B}\right) = \left(k_{d,R}\hat{I}_R, k_{d,G}\hat{I}_G, k_{d,B}\hat{I}_B\right)\max(0, -\omega_{light} \cdot \hat{N})$$

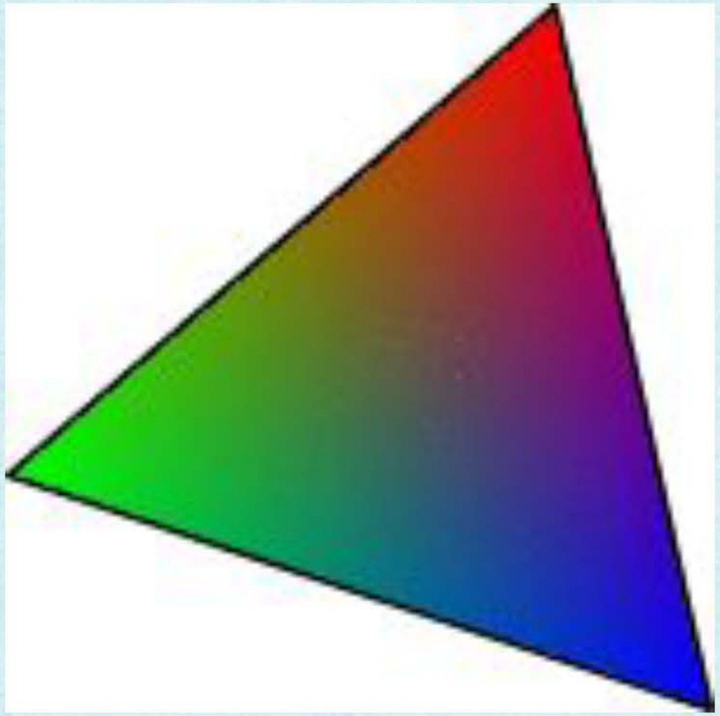
Ambient Lighting

- Add some light in the shadowed regions that would otherwise be completely black (without using global illumination)
- Constant illumination independent of incident light direction (drop the cosine term)
- One can even choose a special ambient light (I_R^a, I_G^a, I_B^a) and/or a special object color $(k_{a,R}, k_{a,G}, k_{a,B})$ for ambient lighting/shading:

$$(L_{o,R}, L_{o,G}, L_{o,B}) = (k_{a,R} I_R^a, k_{a,G} I_G^a, k_{a,B} I_B^a)$$

Triangle Vertex Colors

- The various k_a and k_d values are typically stored on the vertices of triangles
- Given a sub-triangle point p where color needs to be computed, barycentric weights are computed such that $p = \alpha_0 p_0 + \alpha_1 p_1 + \alpha_2 p_2$ for vertices p_0, p_1, and p_2
- Then, $k = \alpha_0 k_0 + \alpha_1 k_1 + \alpha_2 k_2$ defines (at the point p) the various required k values (R, G, and B for both ambient/diffuse). Note: k_0, k_1, k_2 are the corresponding k values at triangle vertices.

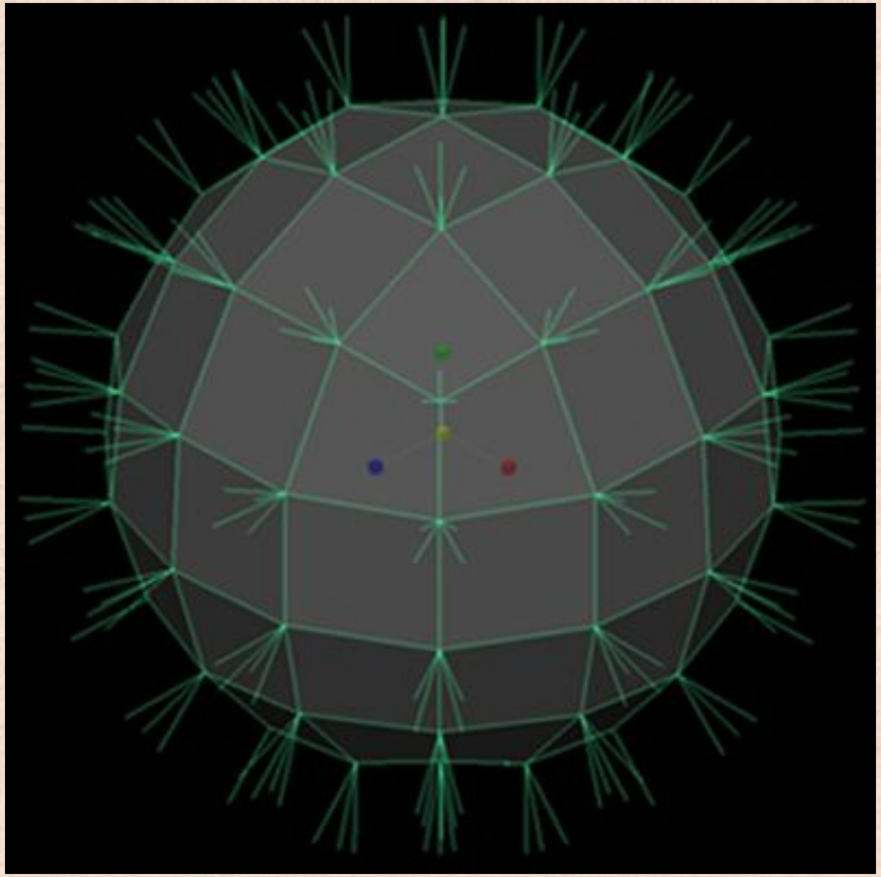


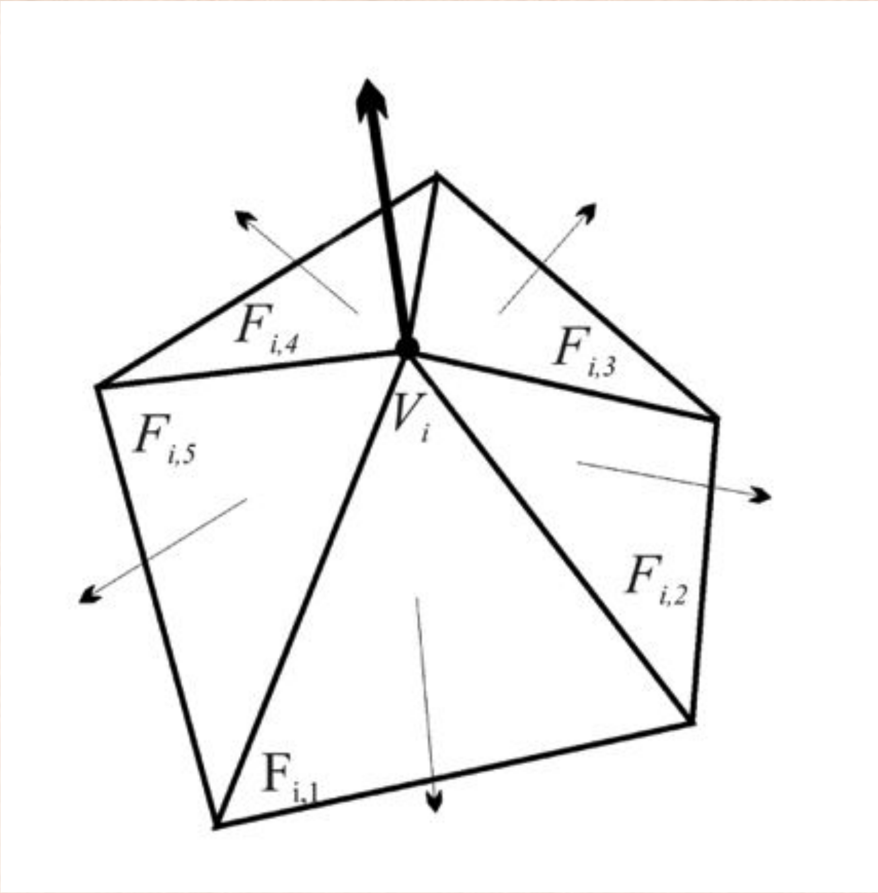
Flat Shading

- Since the normal changes from triangle face to face, one can see all the triangles (as expected)
- This can be alleviated by using more triangles, but that's computationally expensive

(Averaged) Vertex Normals

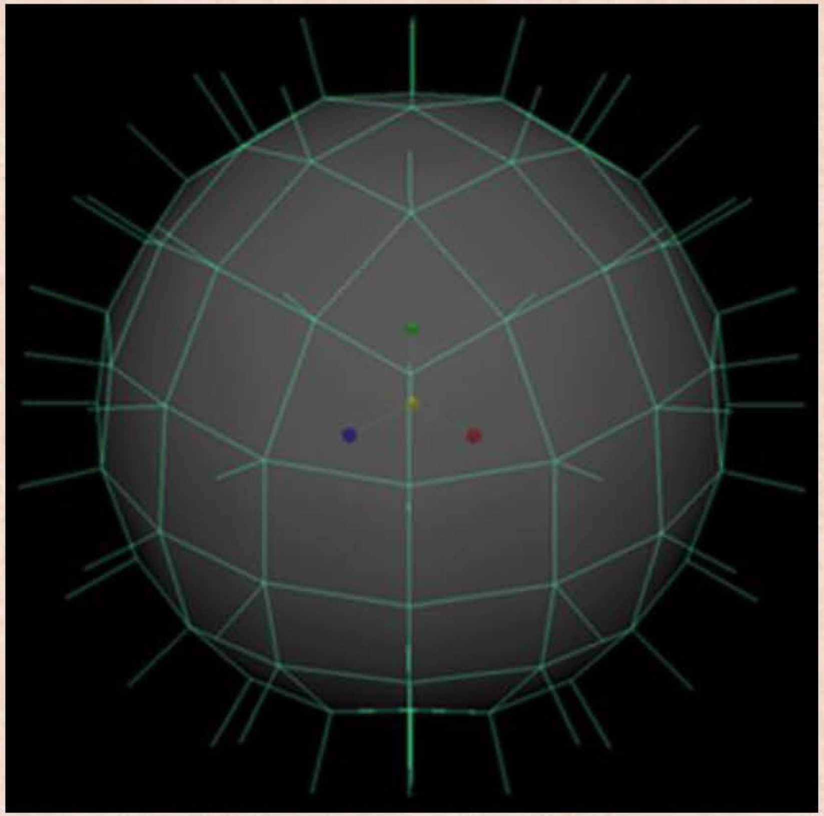
- Each vertex has a number of incident triangles, each with their own normal
- Averaging those face normals (possibly using a weighted average based on area, angle, etc.) yields a unique normal for each vertex





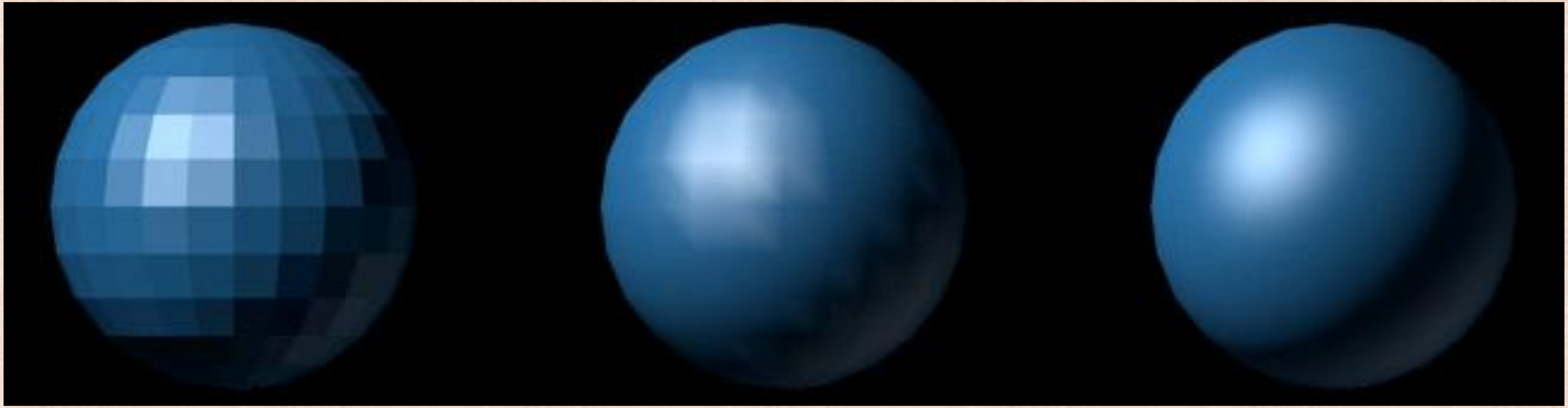
Interpolating Vertex Normals

- Given barycentric weights at a point p, interpolate a normal at p from the unique (precomputed) vertex normals: $\widehat{N}_p = \frac{\alpha_0 N_0 + \alpha_1 N_1 + \alpha_2 N_2}{\|\alpha_0 N_0 + \alpha_1 N_1 + \alpha_2 N_2\|_2}$
- This is called <u>smooth shading</u> (as opposed to flat shading)



Flat vs. Gouraud vs. Phong

- Flat shading uses the actual triangle normal (i.e. the real geometry), and thus the color varies very little per triangle
- Gouraud shading uses averaged vertex normals; however, it evaluates the BRDF at each vertex and uses barycentric interpolation of colors (instead of normals) to the triangle interior
- Phong shading uses averaged vertex normals, and barycentrically interpolates those normals to the interior of the triangle

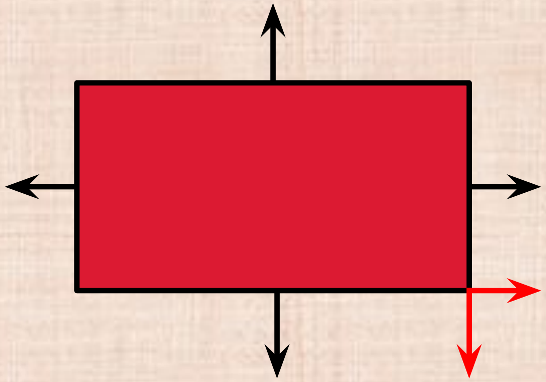


flat shading | Gouraud shading | Phong shading

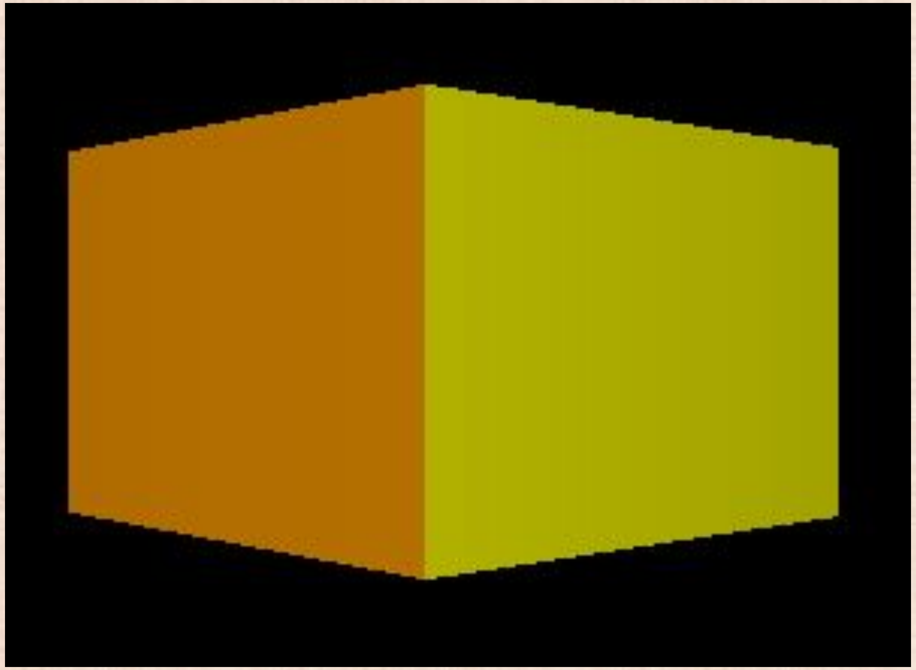
*Don't mix up Phong shading with the Phong reflection

Corners

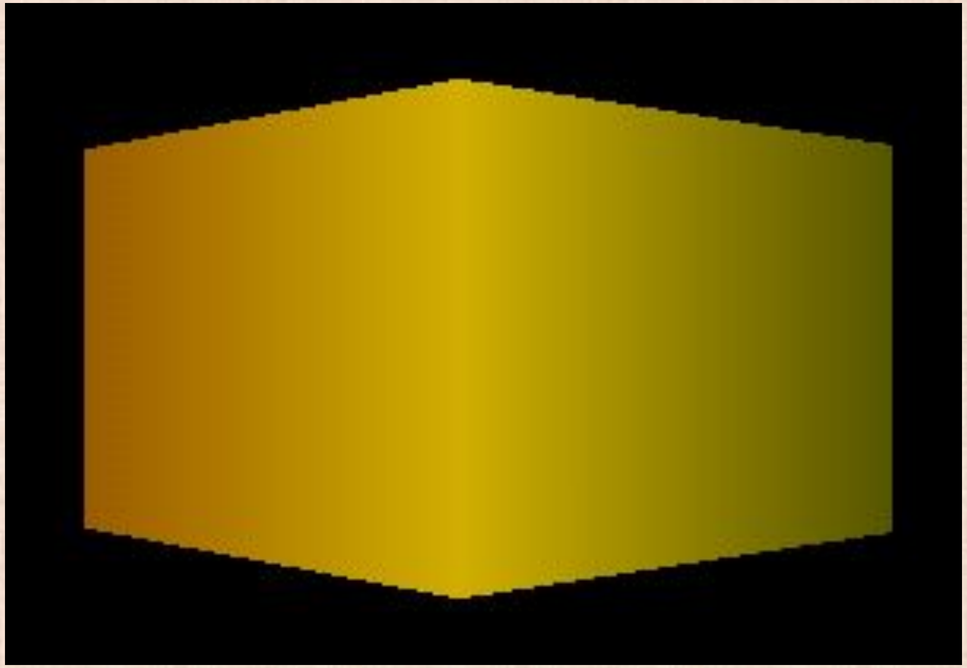
- Normals are poorly defined and difficult to compute at corners
- Averaging vertex normals creates an unrealistic appearance on true edges and corners
- Need to change the type of shading on different parts of the object (the same triangle may need both flat and smooth shading!)



What should the normal be at the corner?



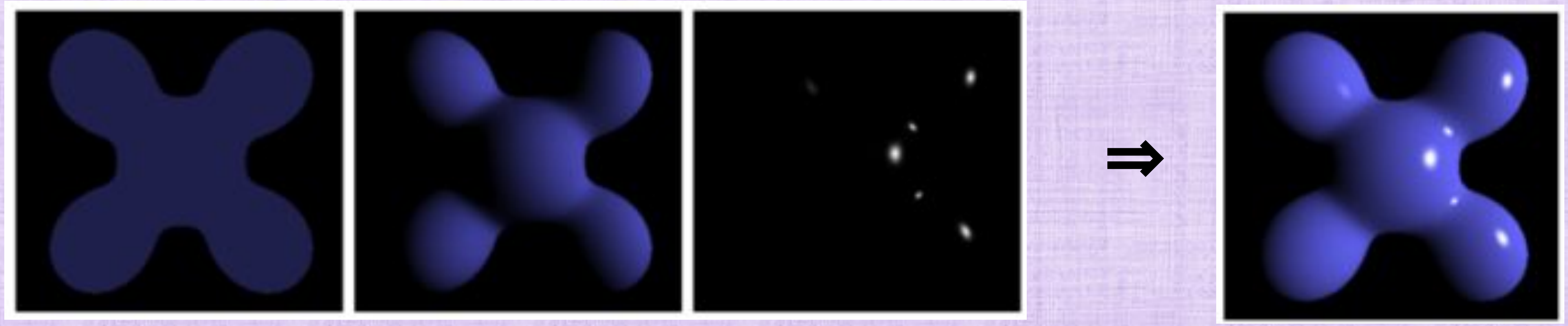
flat shading



smooth shading

Phong Reflection Model

- Uses Ambient, Diffuse, and Specular lighting (the Specular component approximates glossy surfaces that reflect light mostly in directions close to the mirror reflection direction)



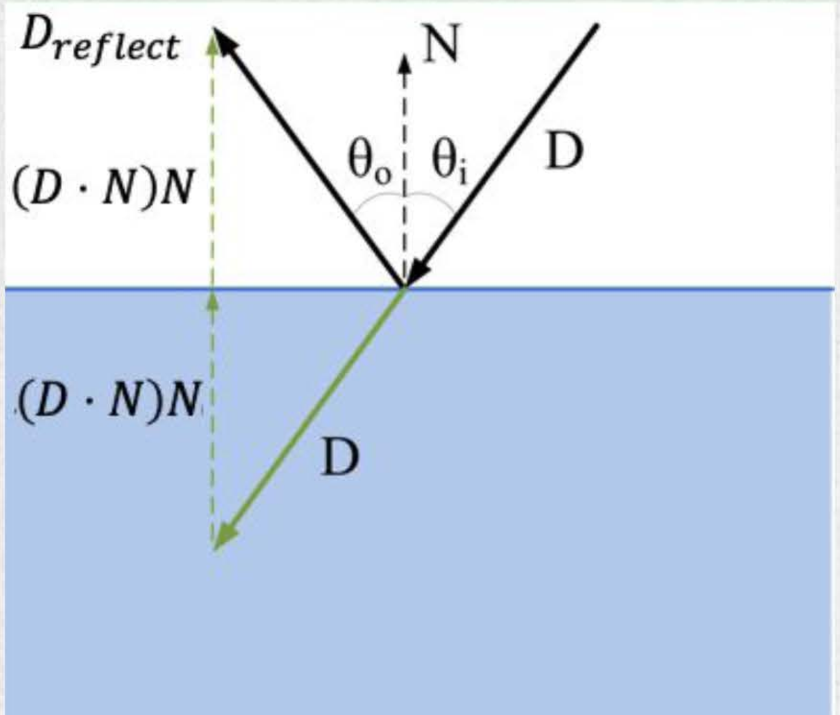
Ambient **Diffuse** **Specular** **Final Image**

$$L_o = \sum_{j \in lights} \left(\underbrace{k_a \hat{I}^j_{i,a}}_{\text{Ambient}} + \underbrace{k_d \hat{I}^j_{i,d} \max(0, \omega_{i,d} \cdot \hat{\mathrm{N}})}_{\text{Diffuse}} + \underbrace{k_s \hat{I}^j_{i,s} \max(\mathrm{V} \cdot \mathrm{R}^j, 0)^s}_{\text{Specular}} \right)$$

* Don’t mix up Phong shading with the Phong reflection

Recall: Reflected Ray

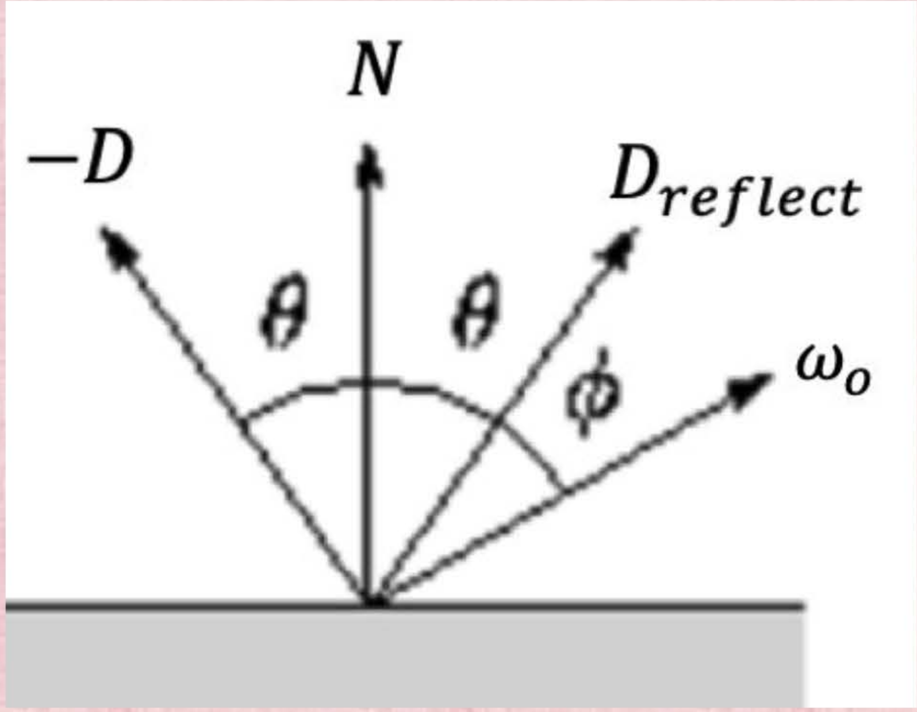
- Given an incoming ray $R(t) = A + Dt$ with direction D, and local (outward) unit normal to the geometry N, the angle of incidence is defined via $D \cdot N = -\|D\|_2 \cos\theta_i$
- For mirror reflection, the incoming/outgoing rays make the same angle with N, i.e. $\theta_o = \theta_i$, and those rays and the normal are all coplanar
- Thus, the reflected ray direction is $D_{reflect} = D - 2(D \cdot N)N$
- Then, the reflected ray is $R_{reflect}(t) = R(t_{int}) + D_{reflect}t$



Specular Highlights

- For a glossy (but not completely smooth) surface, the microscopic spatial variation of normal directions smooths the reflection into a lobe
- The intensity falls off as the viewing direction ω_o differs from the mirror reflection direction $D_{reflect}$:

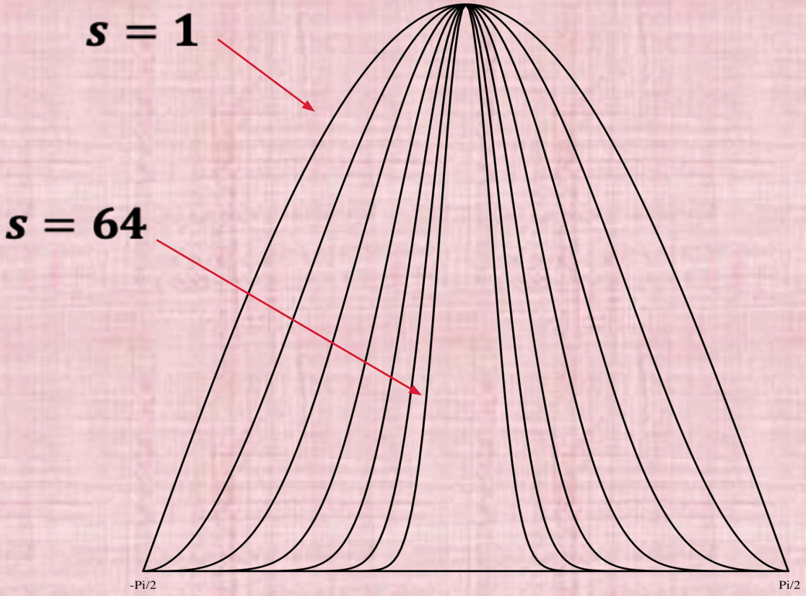
$$L_o(\omega_o) = k_s \, \hat{I}_{\text{light}} \max\left(0, \omega_o \cdot D_{reflect}\right)^s$$

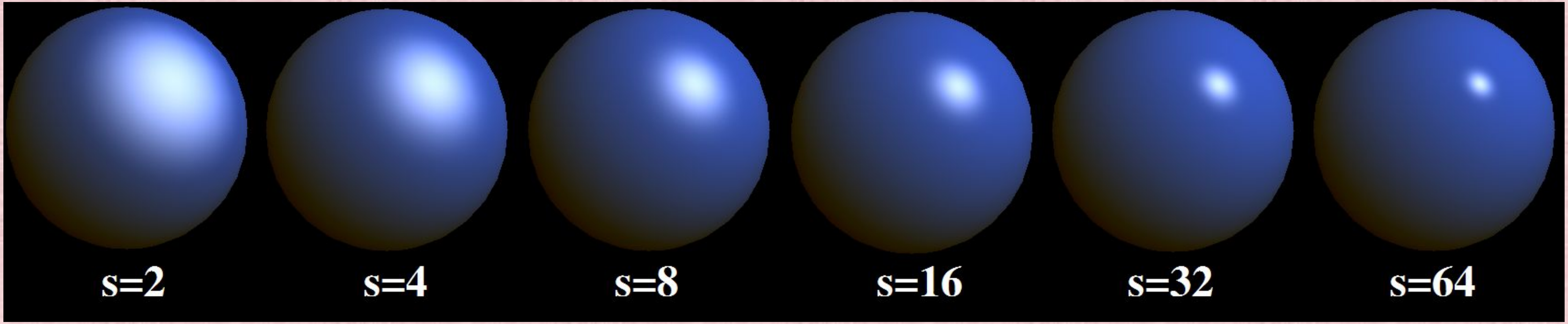


Specular Highlights

- A shininess coefficient s determines the size of the lobe
- A larger s gives a narrower highlight (which converges to a mirror reflection as $s \to \infty$)

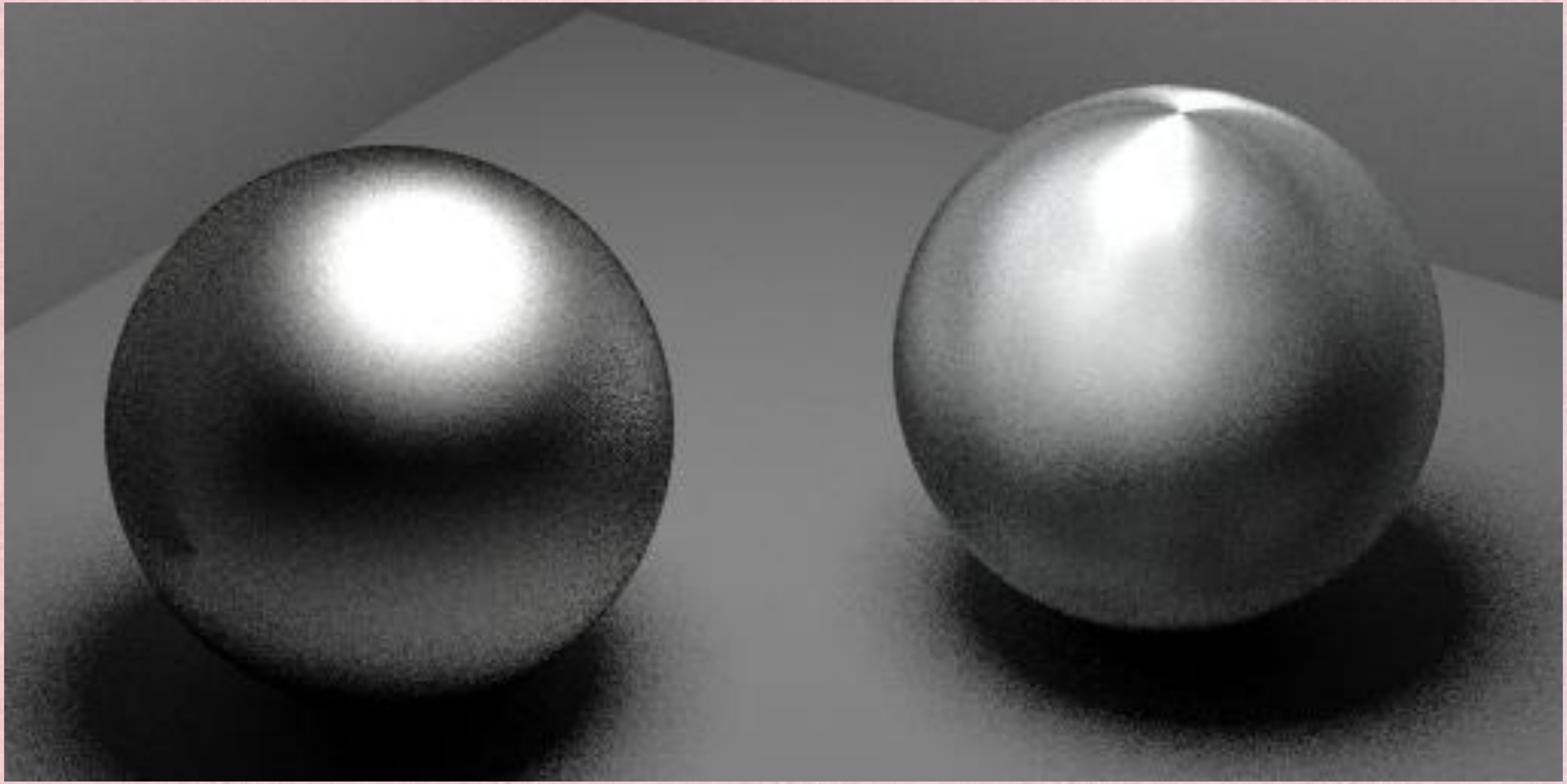
$$L_o(\omega_o) = k_s \, \hat{I}_{\text{light}} \max\left(0, \omega_o \cdot D_{reflect}\right)^s$$





Anisotropic Specular Highlights

- There are various other (and impressive) approximations to specular highlights as well



isotropic specular highlights **anisotropic specular highlights**