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 even more realistic lighting, we'll bounce light all around the scene
- It's tedious to convert between E and L, so use $dE = Ld\omega \cos\theta$ to obtain:

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

- Then,

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

Recall: Area Lights

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

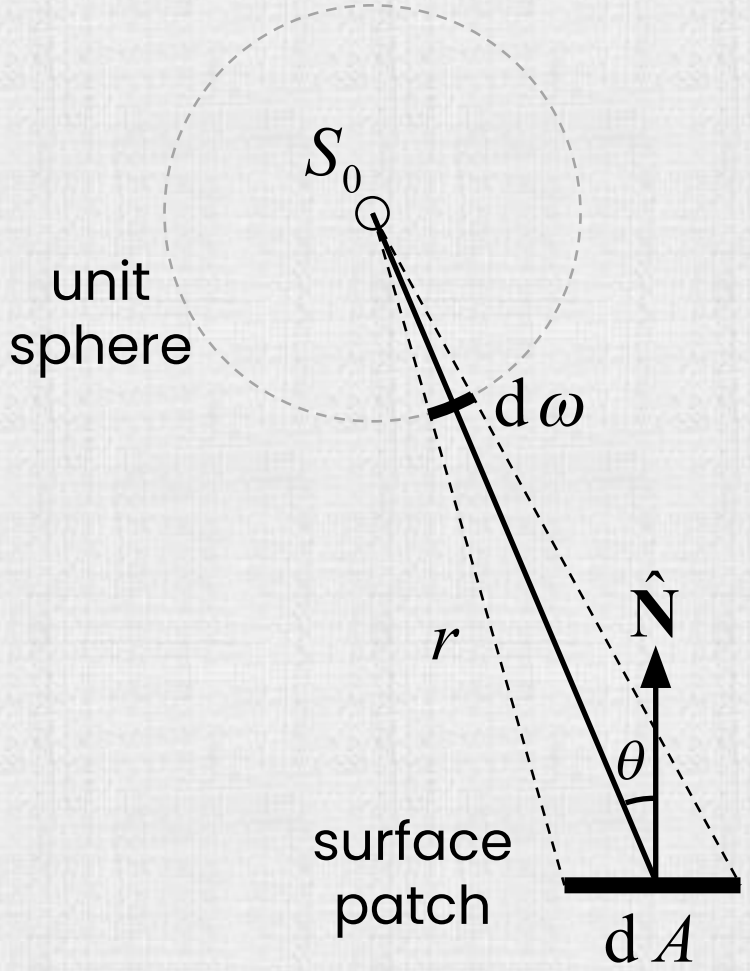
- Break an area light up into (infinitesimally) small area chunks
- Each area chunk emits light into each of the solid angle directions
  - i.e. radiant intensity per area chunk
- Each emitted direction also has a cosine term (similar to irradiance)

- Radiance – radiant intensity per area chunk

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

Recall: Solid Angle vs. Cross-Sectional Area

- The orthogonal cross-sectional area is $dA\ cos\theta$
- So, $d\omega = \frac{dA_{sphere}}{r^2} = \frac{dA\ cos\theta}{r^2}$ (solid angle decreases based on tilting θ and distance r)



Point Lights

- Assume incoming light only comes from a single point light source (with direction ω_{light})
- 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 $\int_{i \in in} \frac{dI}{r^2} = \frac{I}{r^2}$
- If objects are approximately equidistant from the light (e.g. the sun), then r is approximately constant and can be folded into I_{light} to get $\tilde{I}_{light}$:

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

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

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

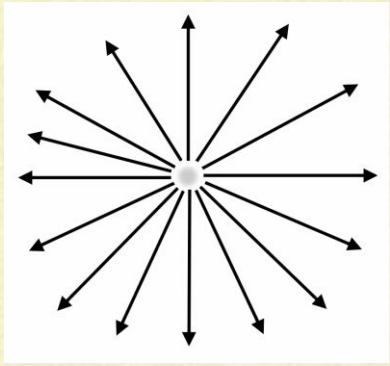
Some Drawbacks

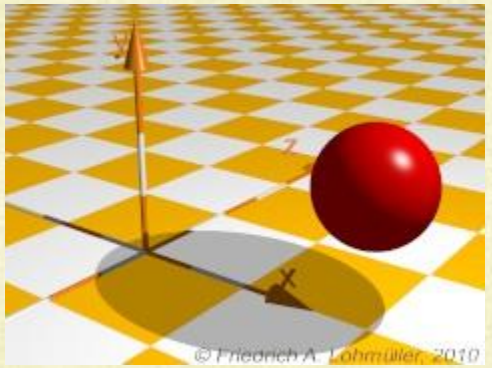
- All the lighting from other objects in the scene has been turned off, so the scene gets darker
- Surfaces that are occluded from all point light sources are completely black
- Shadows have harsh boundaries
- Objects closer to a light source are not brighter than those father away (variance with radius has been removed)
- Etc.

Examples

Point Light

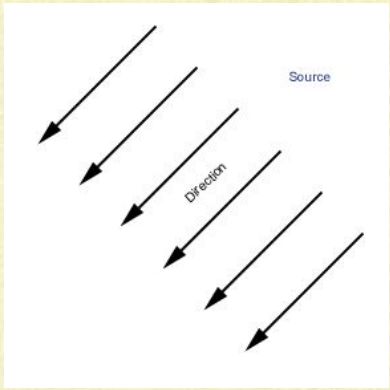
- Light emitted from a single point in space, outwards in every direction

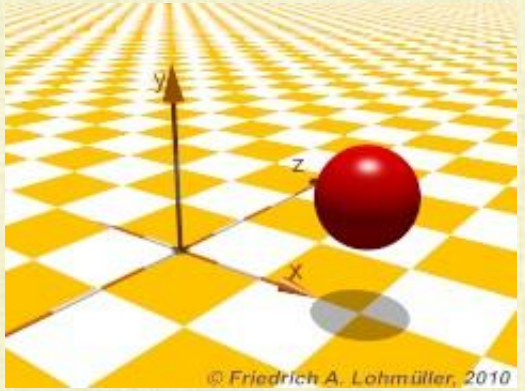




Directional Light

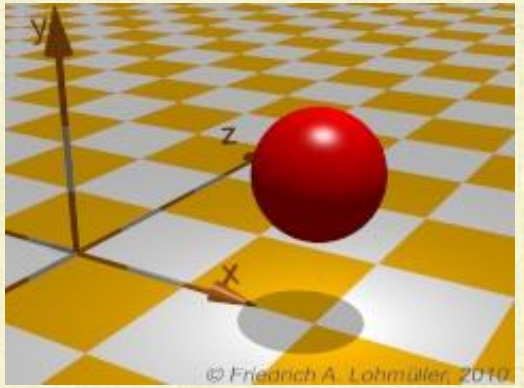
- Always use the the same incoming ray direction
- Good approximation to sunlight, since the sun is far away and rays of sunlight are approximately parallel





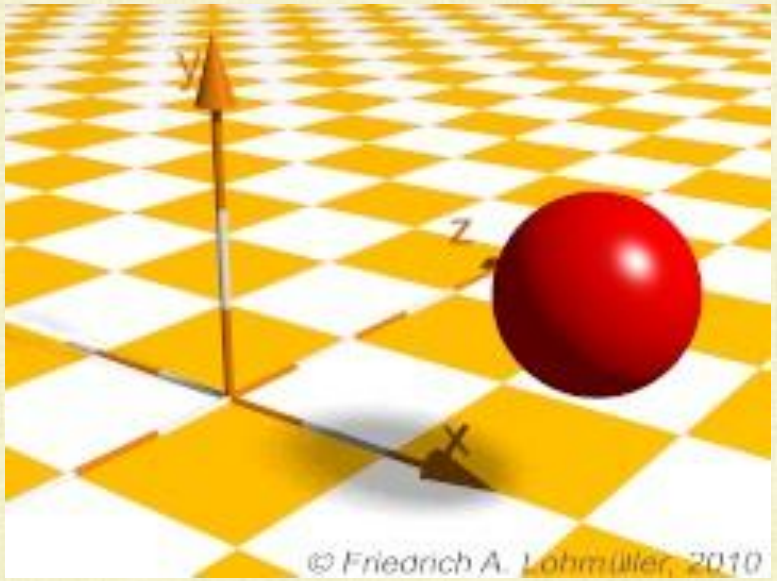
Spotlight

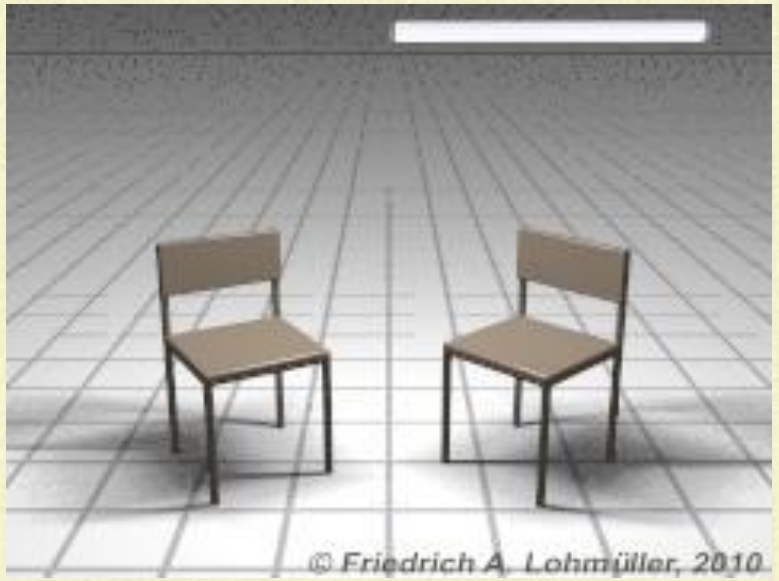
- Angular subset of a point light
- Dot product (outward) directions with a central direction, and prune if larger than some angle

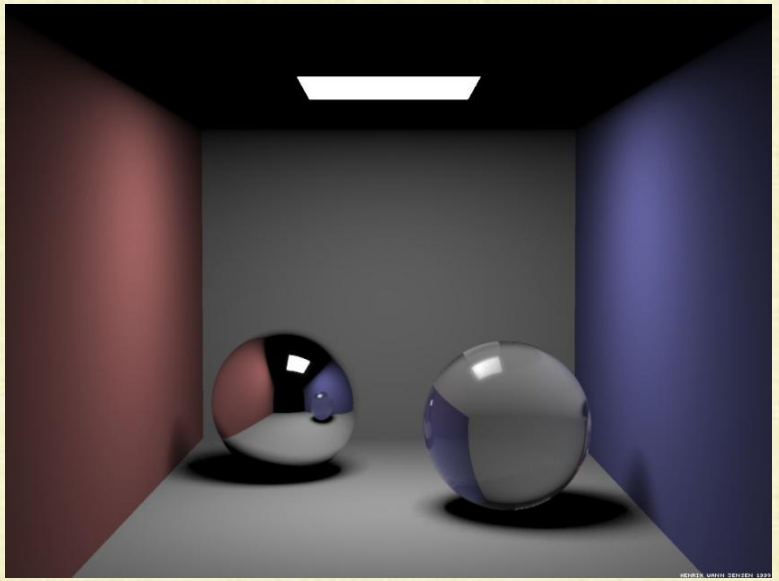


Area Light

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





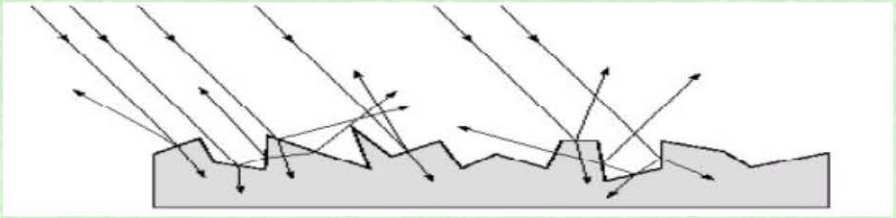


Volume Light

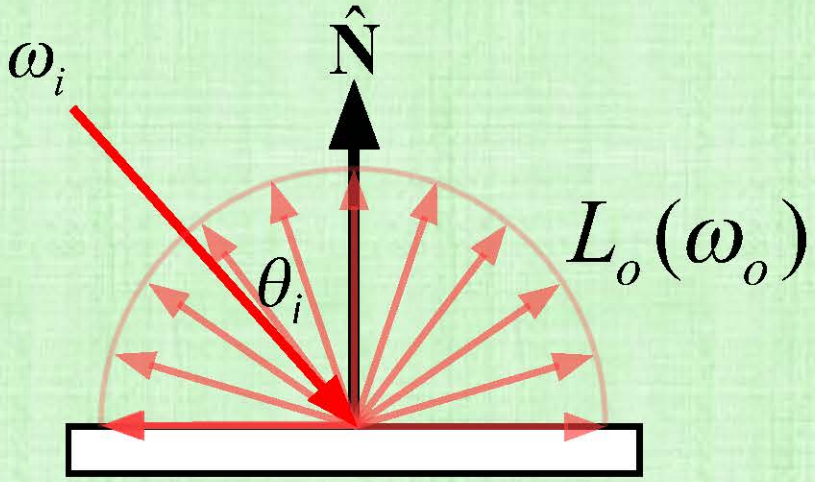
- Can treat as a large collection of point lights (in a volumetric region)

Diffuse Materials

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

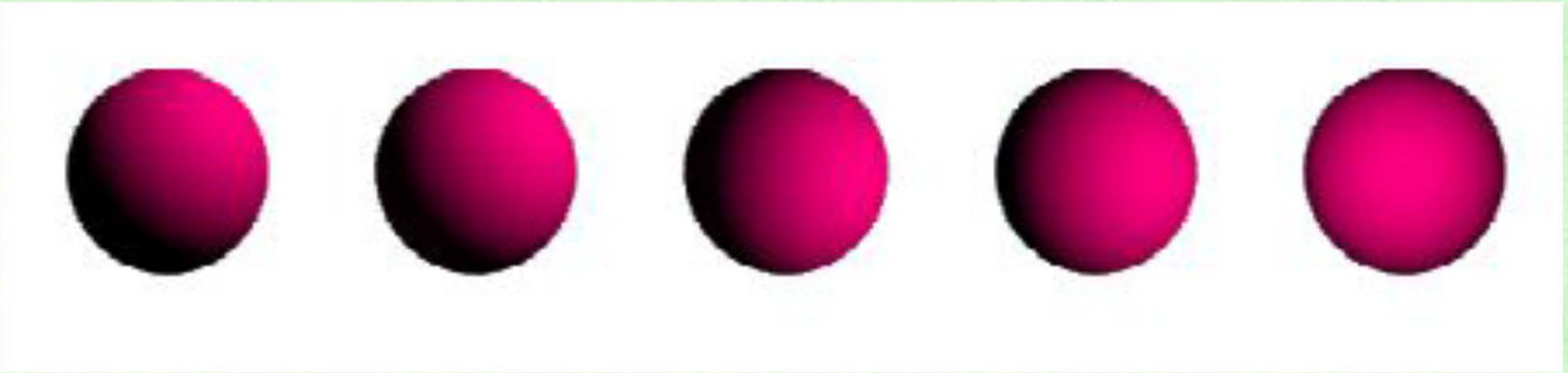


- The BRDF no longer depends on incoming/outgoing directions, and is simply a constant
- $BRDF(\omega_i, \omega_o) = k_d$ and $L_o = k_d \cos\theta_{light} \tilde{I}_{light}$



Diffuse Materials

- Shading depends 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



- An object with (diffuse) color $\left(k_d^R, k_d^G, k_d^B\right)$ hit by a light with color $(\tilde{I}^R, \tilde{I}^G, \tilde{I}^B)$ results in:

$$(L_o^R, L_o^G, L_o^B) = \left(k_d^R \tilde{I}^R, k_d^G \tilde{I}^G, k_d^B \tilde{I}^B\right) \max\left(0, -\hat{\omega}_{light} \cdot \hat{N}\right)$$

$\cos\theta_{light}$

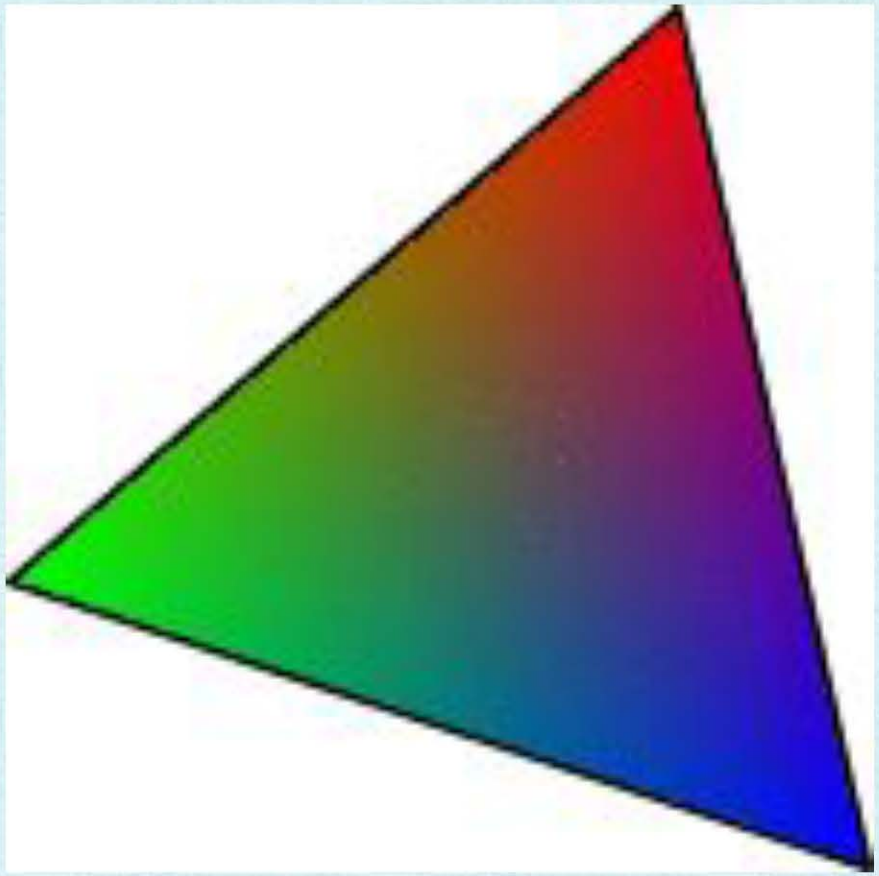
Ambient Lighting

- Add some light in regions that cannot see any light sources
- Constant illumination independent of incident light direction (drop the cosine term)
- An ambient light (I_a^R, I_a^G, I_a^B) on an object with color $\left(k_d^R, k_d^G, k_d^B\right)$ gives:

$$(L_o^R, L_o^G, L_o^B) = (k_a^R I_a^R, k_a^G I_a^G, k_a^B I_a^B)$$

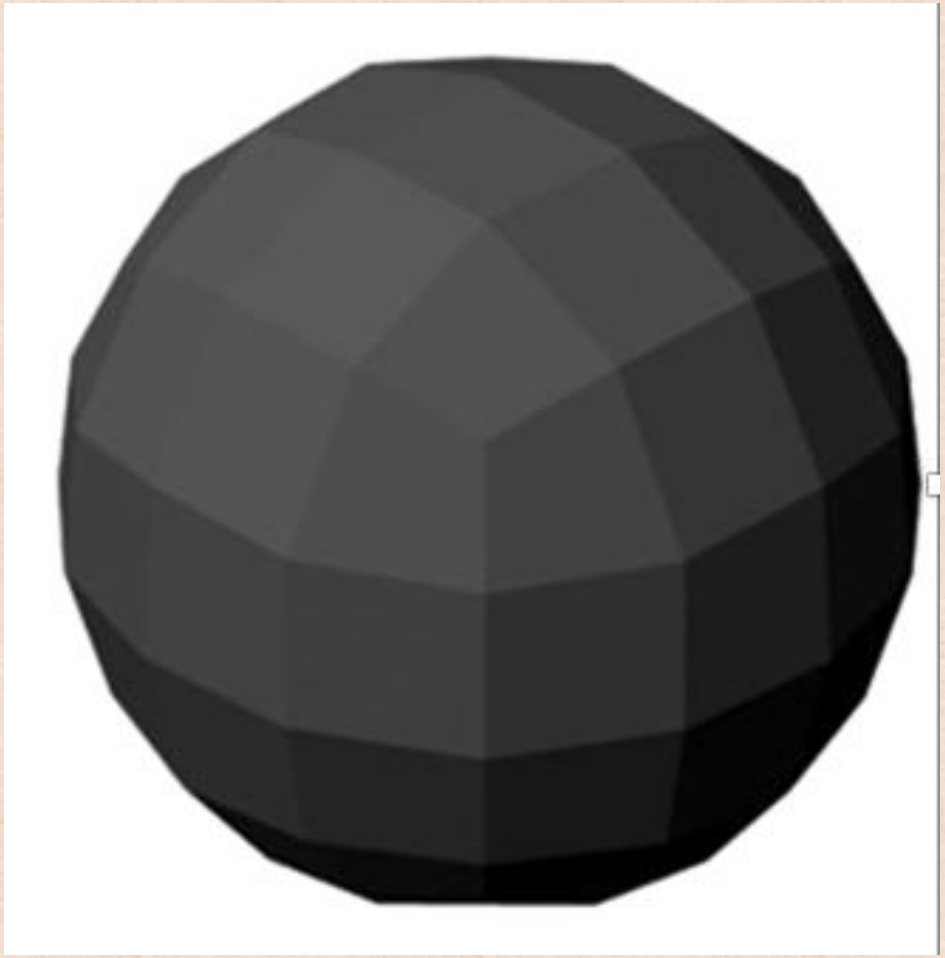
Vertex Colors

- The various k_a and k_d values are stored on the vertices of triangles: p_0, p_1, p_2
- Given a sub-triangle point p, compute barycentric weights: $p = \alpha_0 p_0 + \alpha_1 p_1 + \alpha_2 p_2$
- Then, compute $k = \alpha_0 k_0 + \alpha_1 k_1 + \alpha_2 k_2$ for all relevant k values (R, G, B for ambient/diffuse)



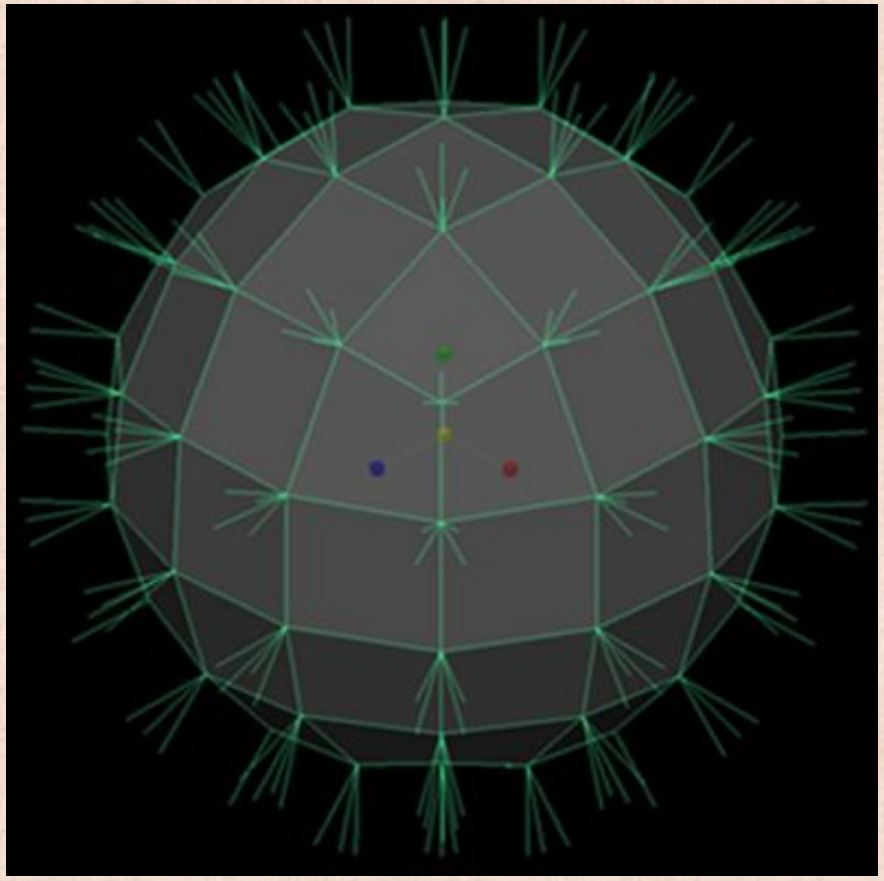
Flat Shading

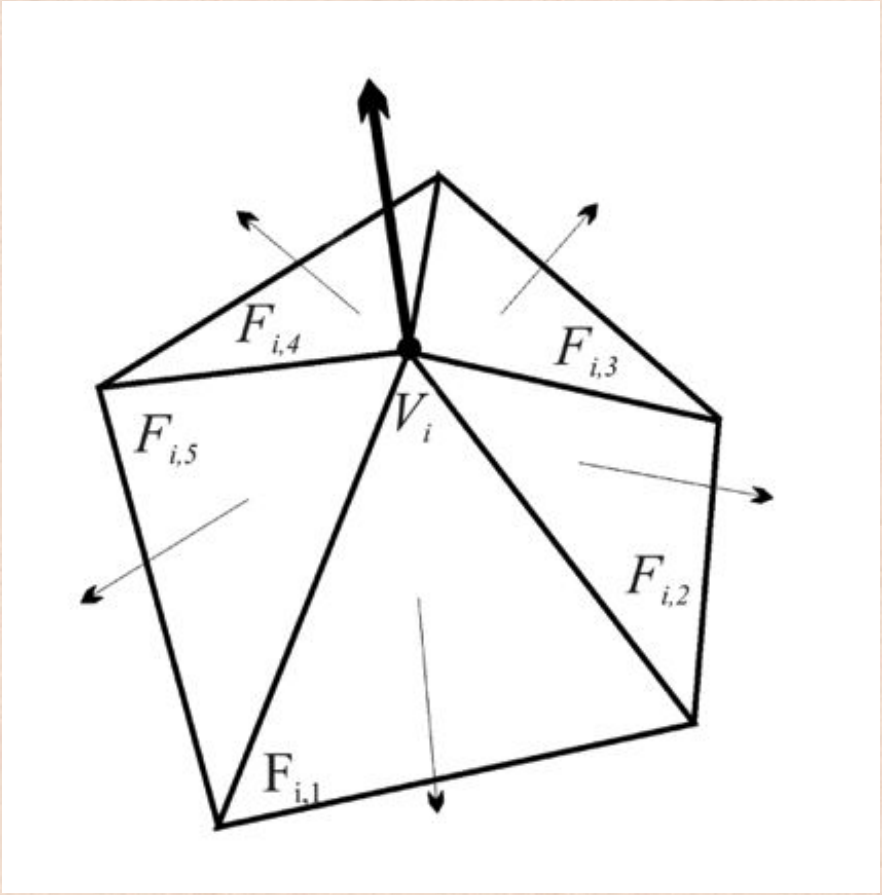
- Since the normal changes from triangle face to face, one can see the triangles (as expected)
- This can be alleviated by using more triangles (but that's computationally expensive)



(Averaged) Vertex Normals

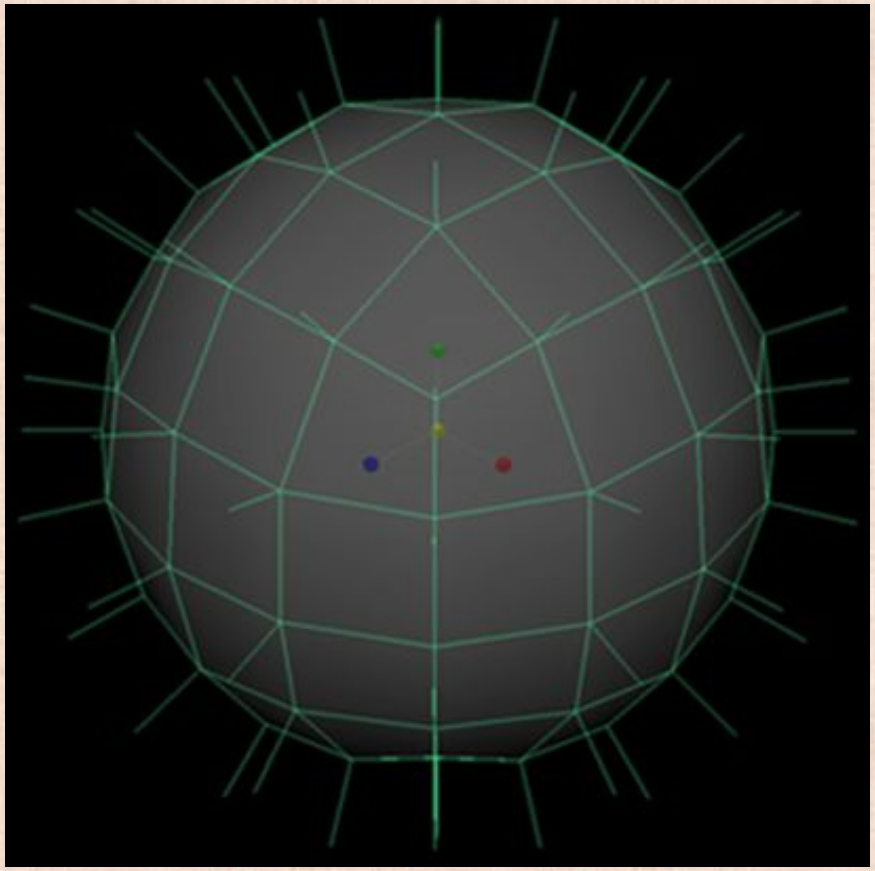
- Each vertex has a number of incident triangles, each with their own normal
- Average those face normals (or a weighted average based on: area, angle, etc.) gives a unique normal for each vertex

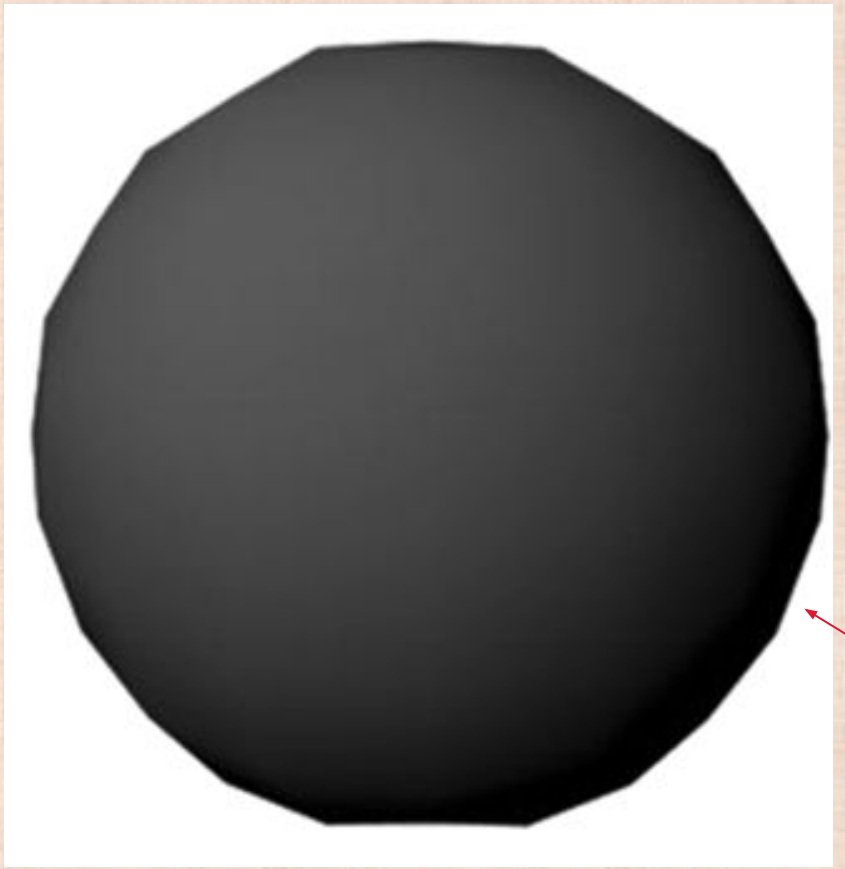




Smooth Shading

- Given barycentric weights at a point p, interpolate a normal at p from the unique (precomputed) vertex normals: $\hat{N}_p = \frac{\alpha_0\hat{N}_0+\alpha_1\hat{N}_1+\alpha_2\hat{N}_2}{\|\alpha_0\hat{N}_0+\alpha_1\hat{N}_1+\alpha_2\hat{N}_2\|_2}$

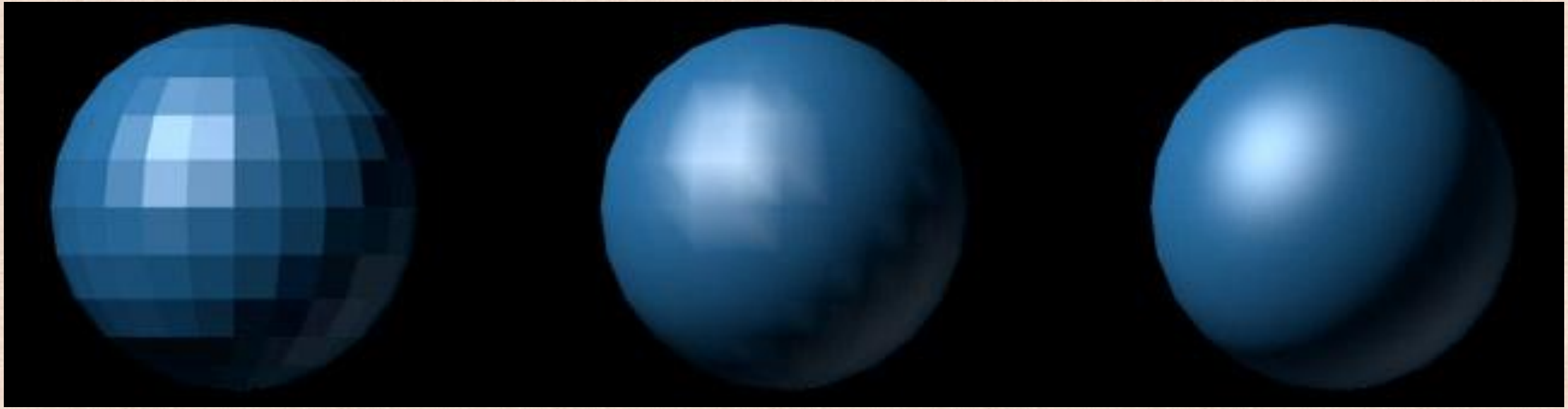




faceted silhouette

Flat vs. Gouraud vs. Phong (Shading)

- Flat uses the actual normals (i.e. the real geometry), so you can see the triangles
- Gouraud uses (averaged) vertex normals; however, it evaluates the BRDF at each vertex and uses barycentric interpolation of *final* color to the triangle interior
- Phong uses (averaged) vertex normals, and barycentrically interpolates them to the triangle interior

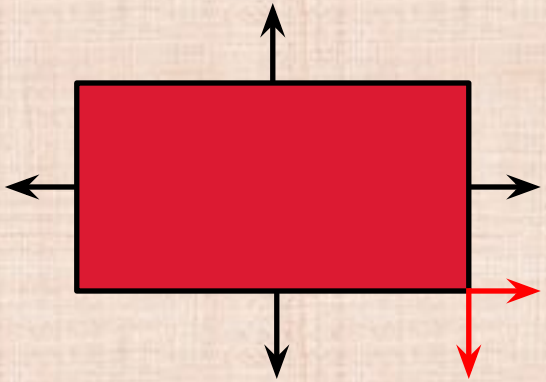


flat Gouraud Phong

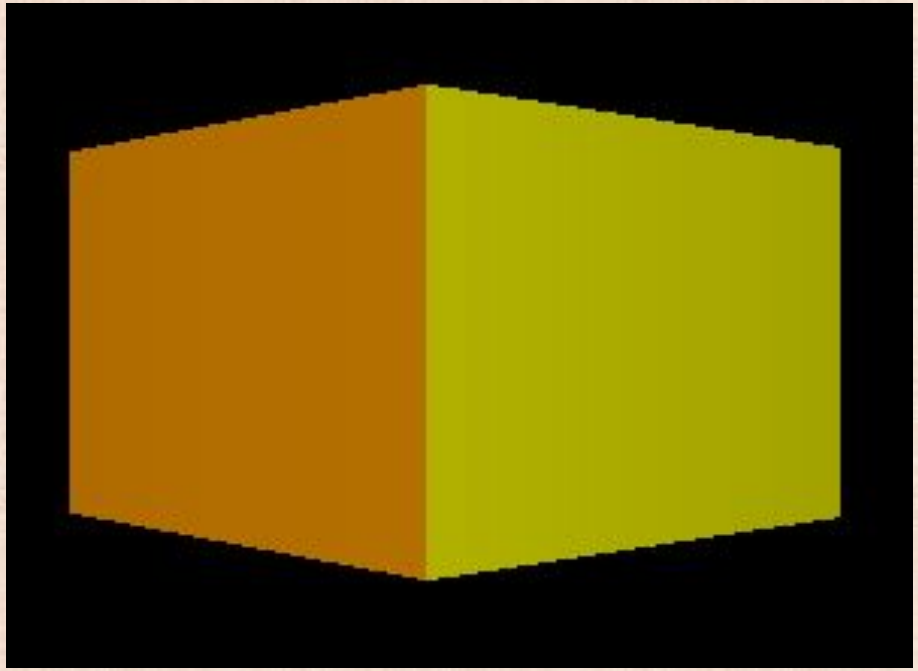
*Don't mix up Phong shading with the Phong reflection

Edges and Corners

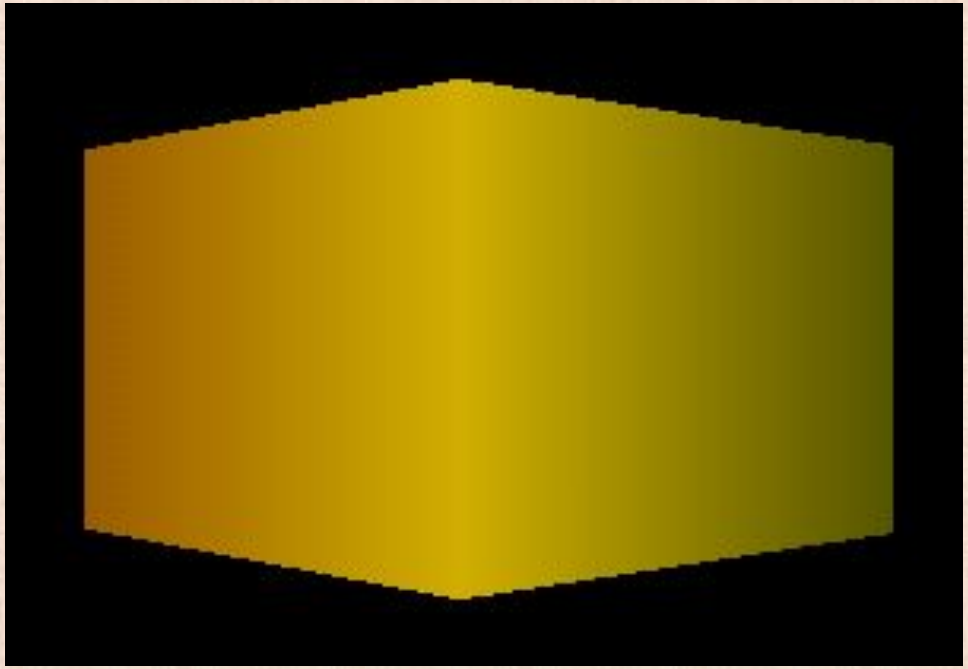
- Normals are poorly defined and difficult to compute at corners
- Averaging vertex normals creates an unrealistic appearance on desired edges and corners
- Use different types of shading on different parts of the object (the same triangle may need both flat and smooth shading)



What should the normal be at the corner?



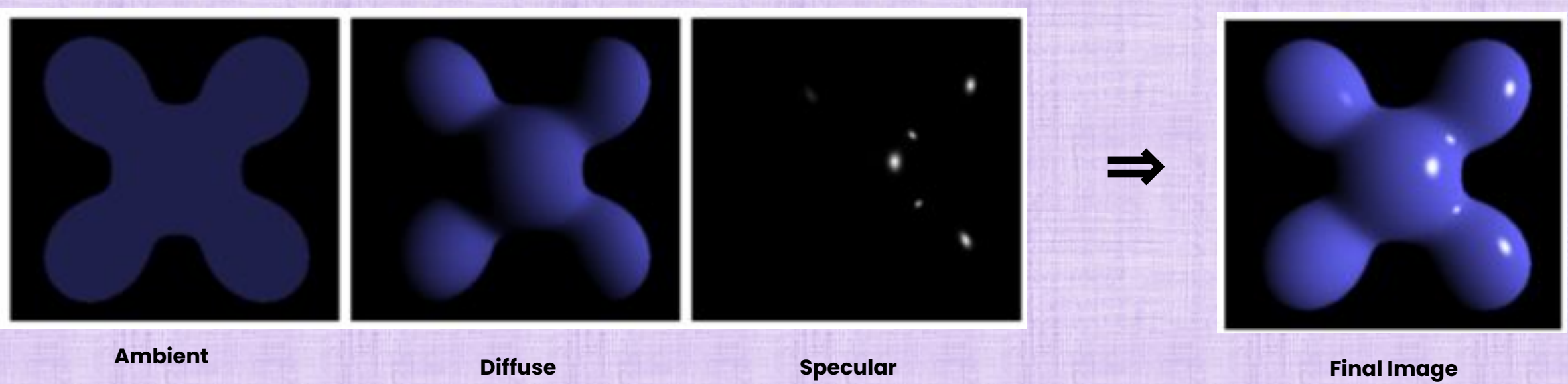
flat



smooth

Phong Reflection Model

- Ambient, Diffuse, and Specular lighting (specular approximates glossy surfaces)



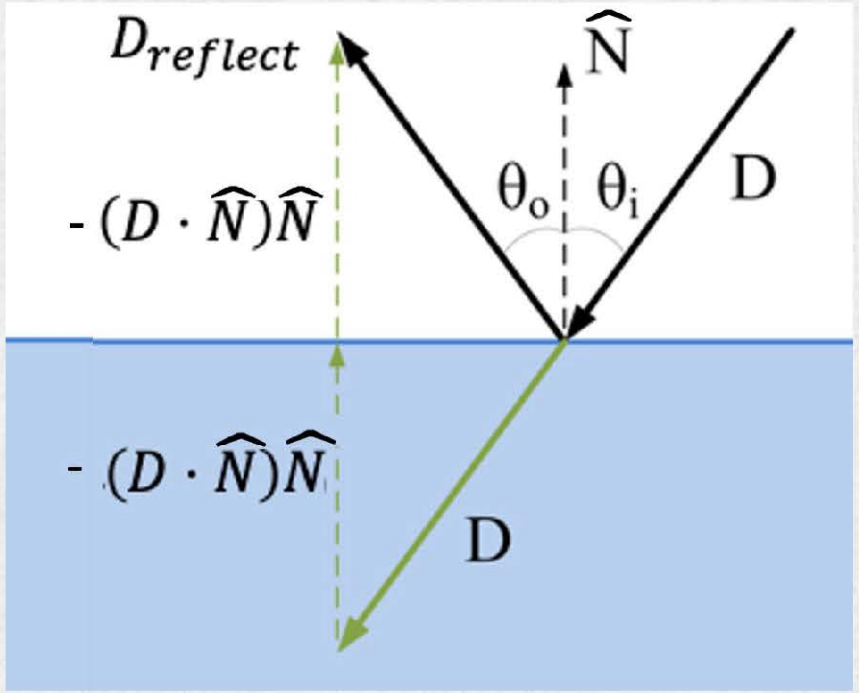
Ambient **Diffuse** **Specular** **Final Image**

$$L_o(\omega_o) = \sum_{j=1}^{\#lights} \underbrace{k_a I_a^j}_{\text{Ambient}} + \underbrace{k_d \tilde{I}_d^j \max(0, -\hat{\omega}_{light} \cdot \hat{N})}_{\text{Diffuse}} + \underbrace{k_s \tilde{I}_s^j \max(0, \hat{\omega}_o \cdot \hat{D}_{reflect})^s}_{\text{Specular}}$$

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

Recall: Reflected Ray

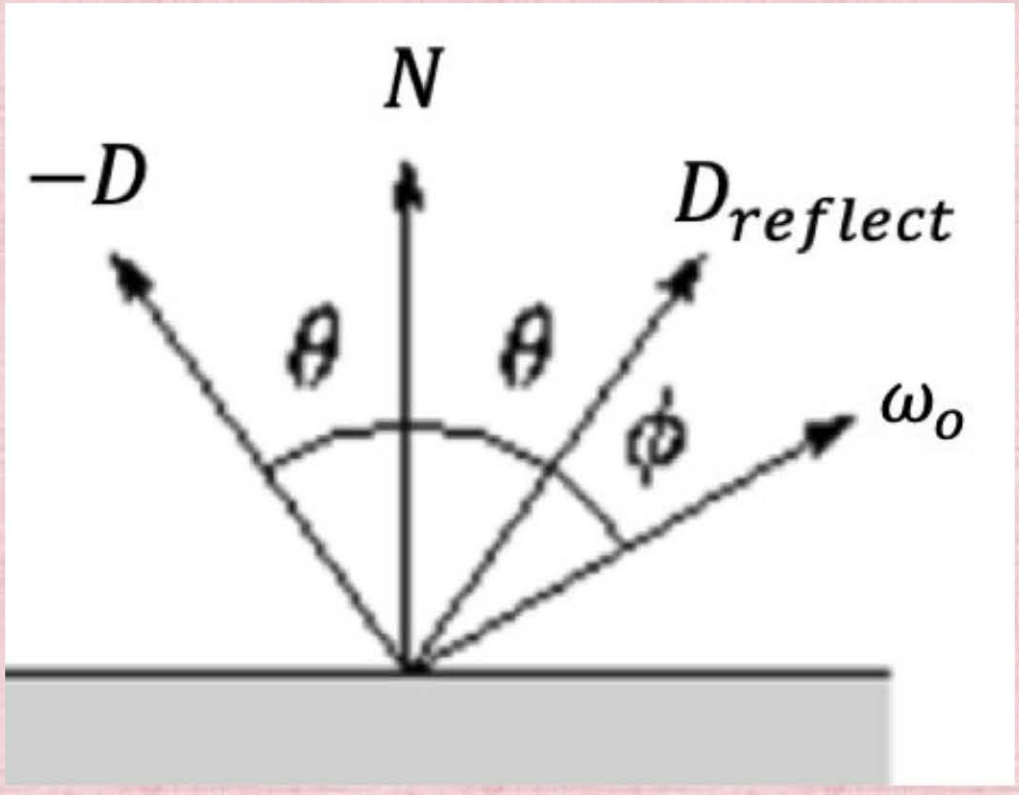
- Given an incoming ray $R(t) = A + Dt$ with direction D, and local (outward) unit normal to the geometry $\widehat{N}$, the angle of incidence is defined via $D \cdot \widehat{N} = -\|D\|_2 \cos\theta_i$
- For mirror reflection, incoming/outgoing rays make the same angle with $\widehat{N}$
- That is,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



Specular Highlights

- For a glossy (but not completely smooth) surface, microscopic spatial variation (of normals) smooths the reflection into a lobe
- Intensity falls off as the viewing direction differs from the mirror reflection direction:

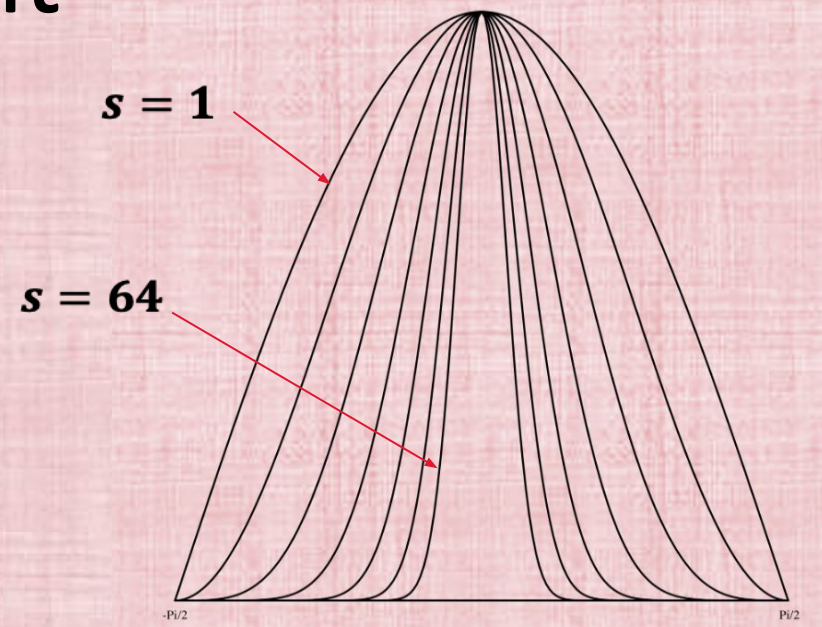
$$L_o(\omega_o) = k_s \, \tilde{I}_{\text{light}} \max\left(0, \hat{\omega}_o \cdot \hat{D}_{reflect}\right)^s$$

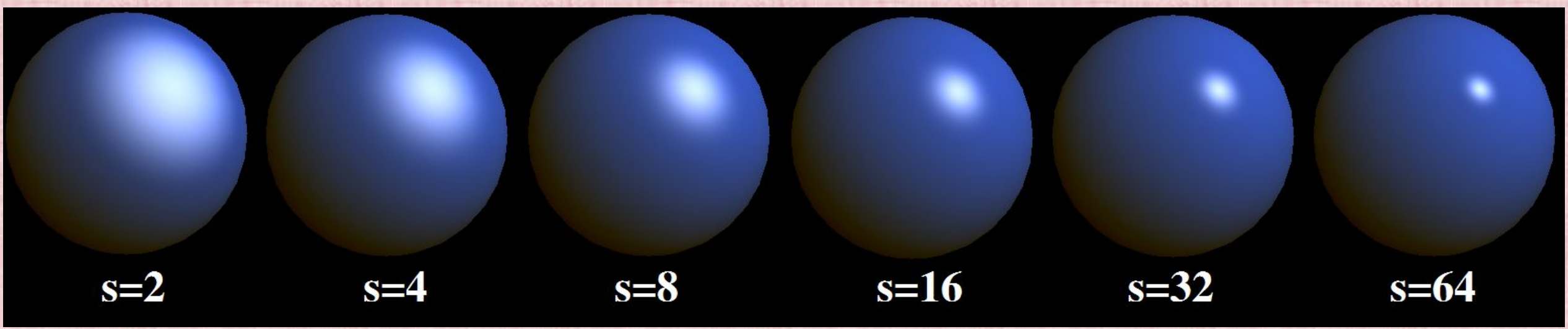


Shininess Coefficient

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

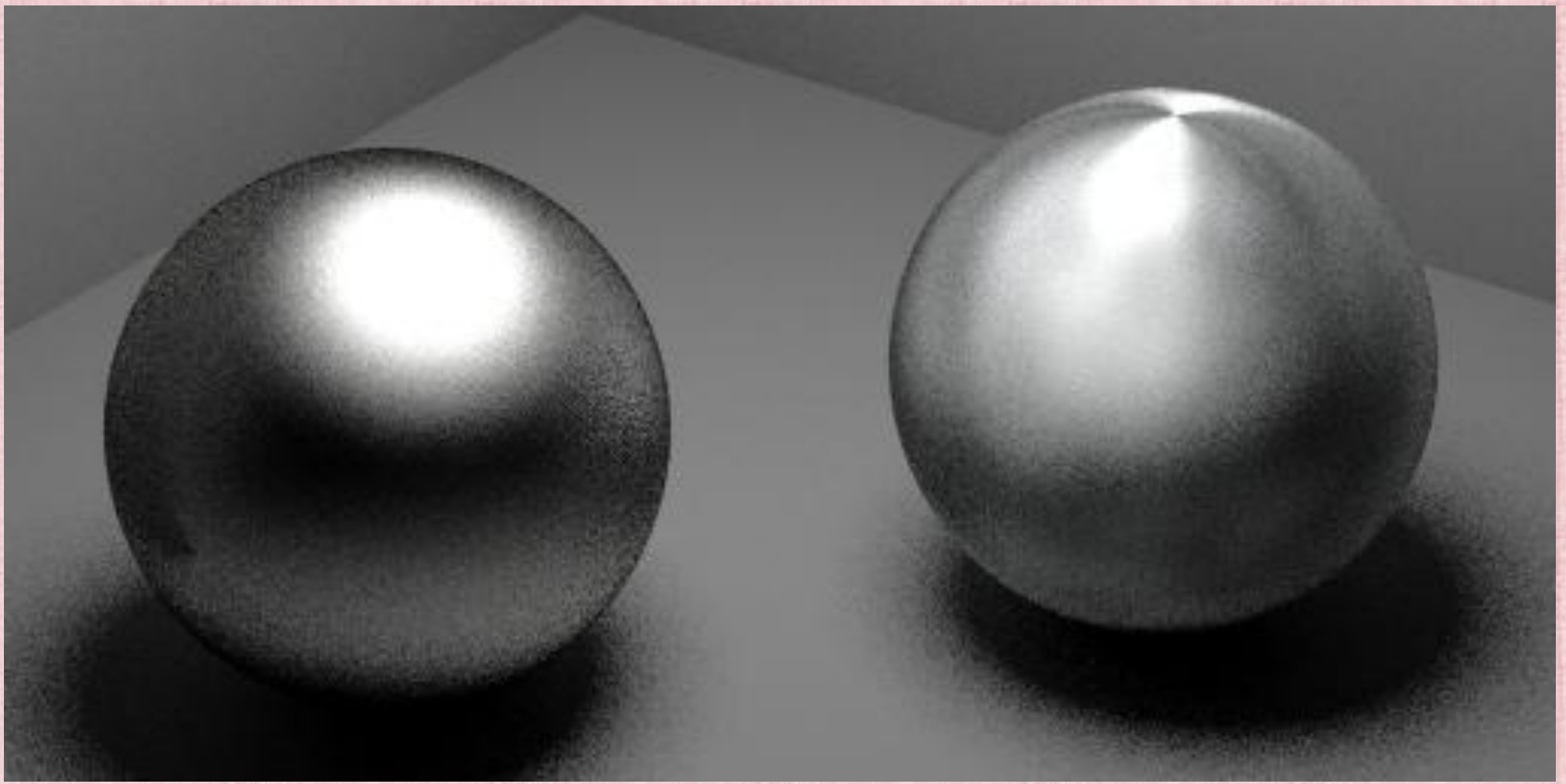
$$L_o(\omega_o) = k_s \, \tilde{I}_{\text{light}} \max\left(0, \hat{\omega}_o \cdot \hat{D}_{reflect}\right)^s$$





Anisotropic Specular Highlights

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



isotropic

anisotropic