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 $L = \frac{dE}{d\omega \cos\theta}$ to obtain:

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

- Then,

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

Recall: Radiance from an Area Light Source

- Light power is emitted per unit area (not from a single point)
- The emitted light goes in various directions (measured via 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 has a cosine term (similar to irradiance)

- Radiance – radiant intensity per area chunk

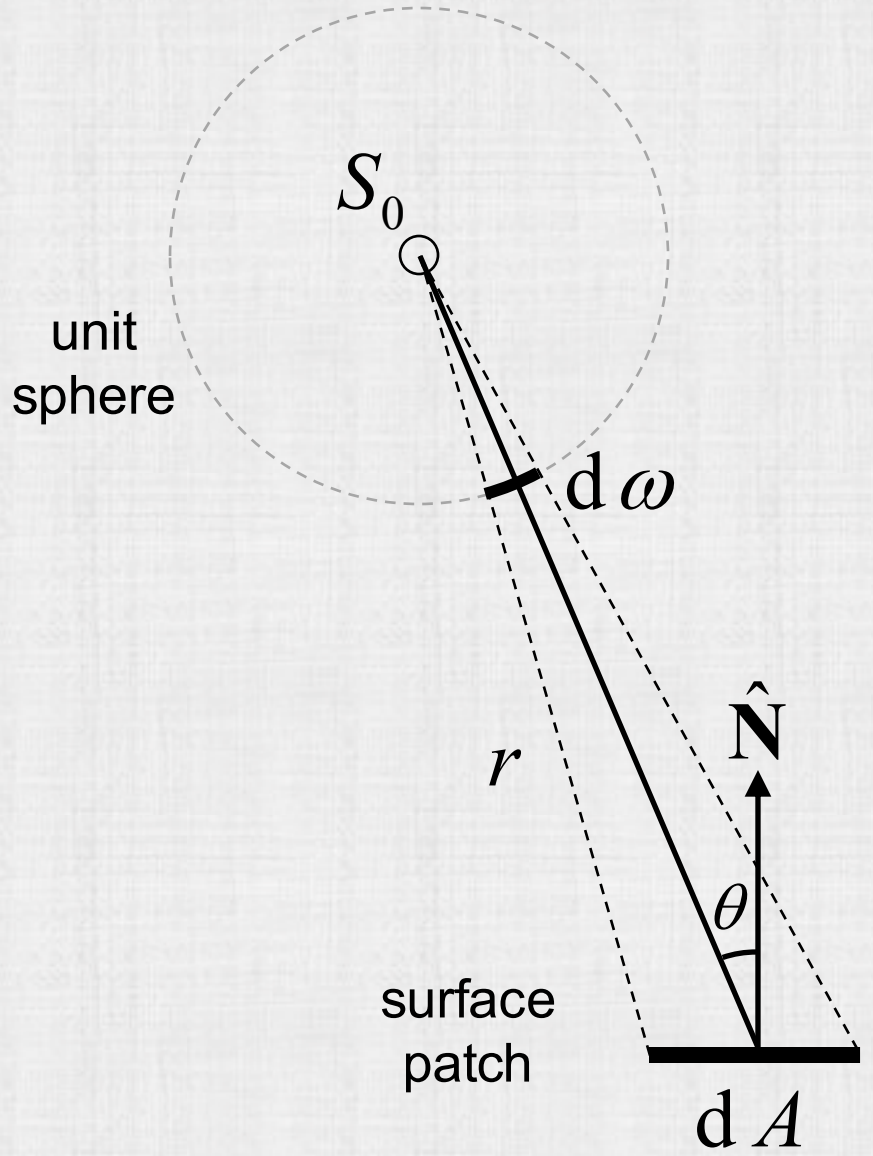
$$L = \frac{dI}{dA\cos\theta_{light}} \left(= \frac{d^2\Phi}{d\omega\, dA\cos\theta_{light}} = \frac{dE}{d\omega\cos\theta_{light}} \right)$$

Recall: Solid Angle vs. Cross-Sectional Area

- Definition of solid angle: $d\omega = \frac{dA_{sphere}}{r^2}$
- From the previous slide: the (orthogonal) cross-sectional area of the surface patch is $dA \cos\theta$
- So, given a sphere of radius r (in the figure):

$$d\omega = \frac{dA \cos\theta}{r^2}$$

- The solid angle decreases as the surface tilts away from the light (increasing θ, decreasing $\cos\theta$)
- The solid angle decreases as the surface is moved further from the light (increasing r)



Point Lights

- Assume that the incoming light comes from a single point light source (with direction ω_{light})
- Then the BRDF and the cosine terms are approximately constant:

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

- Since $L = \frac{dI}{dA \cos\theta}$ and $d\omega = \frac{dA\, cos\theta}{r^2}$, the integral is $\int_{i \in hemi} \frac{1}{r^2} dI = \frac{I}{r^2}$
- If all the objects in a scene are approximately equidistant from the light (e.g. from the sun), then r is approximately constant and can be folded into I_{light} to get $\tilde{I}_{light}$:

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

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

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

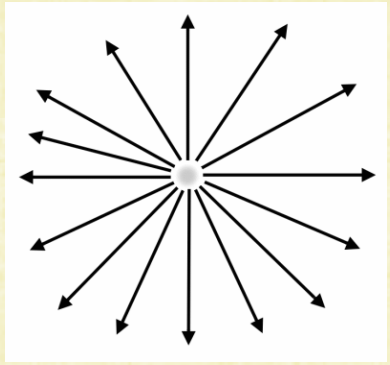
Point Light Drawbacks

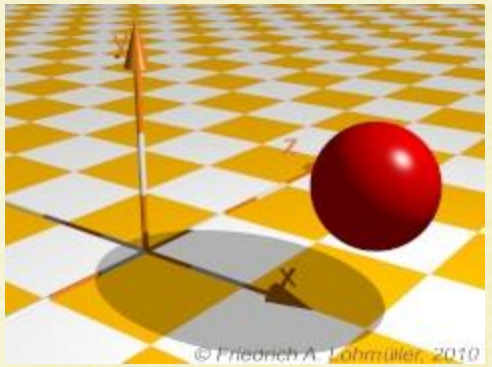
- All the lighting from other objects in the scene is turned off; thus:
  - The scene is darker overall
  - Surfaces occluded from all point light sources are completely black
  - Shadows have harsh boundaries
  - There is no color bleeding
  - Etc.

- Ignoring the dependence on distance (i.e. on r^2) causes an issue too:
  - Objects closer to a light source are not brighter than those father away

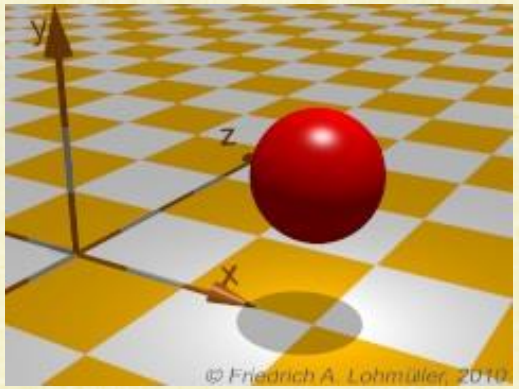
Point Light Examples

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

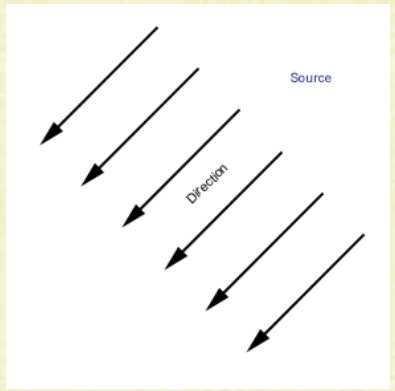


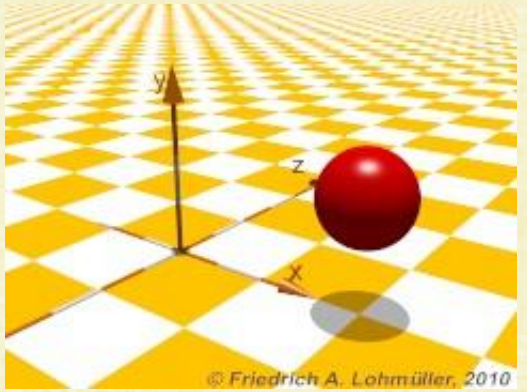


Spotlight
• Angular subset of a point light
• Prune directions a cutoff angle away from a central direction (use a dot product)



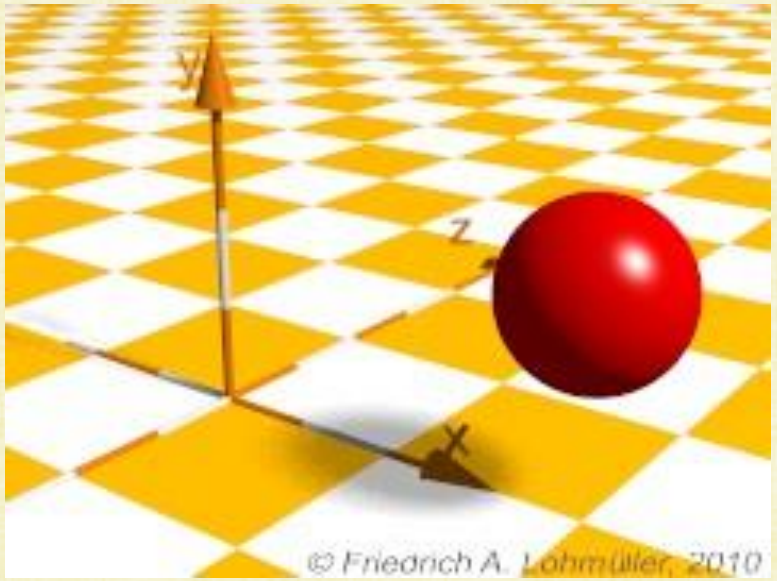
Directional Light
• Always use the the same incoming ray direction
• Models a very distant point light (like the sun)

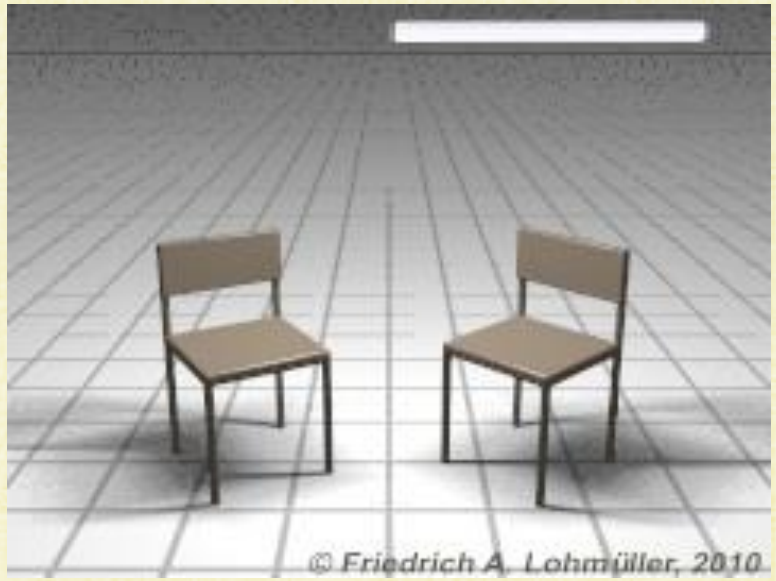


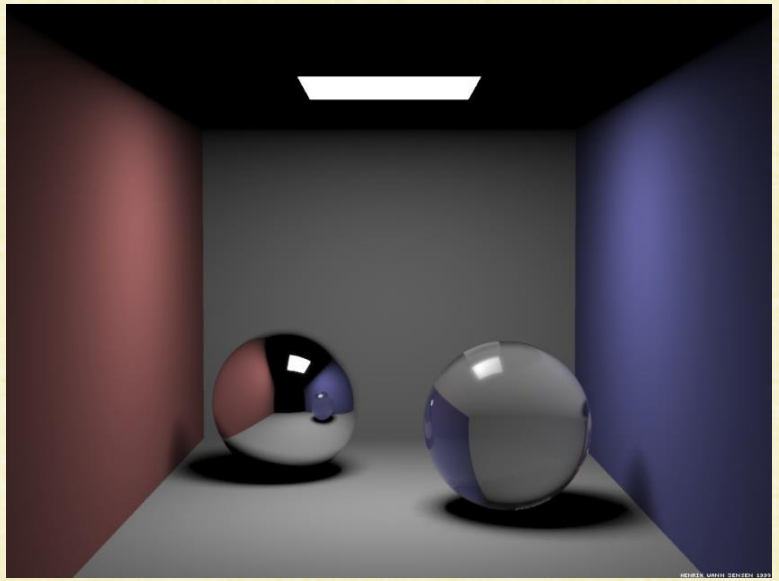


Area Lights (approximated with point lights)

- Approximate an area light by distributing a (large) number of point lights across the surface
- The sum of the strengths of all the point lights should equal the strength of the area light
- Creates softer shadows!





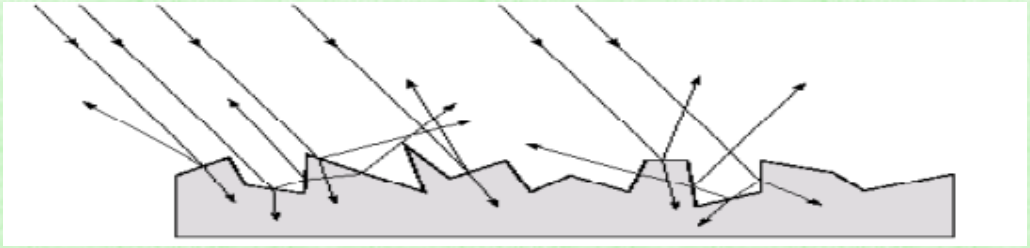


Volume Lights (approximated with point lights)

- Distribute a (large) number of point lights throughout the volume
- The point lights can be animated in time too

Diffuse Materials

- They reflect light equally in all directions, independent of the incoming light direction
- This happens with very rough surfaces, with many tiny microfacets

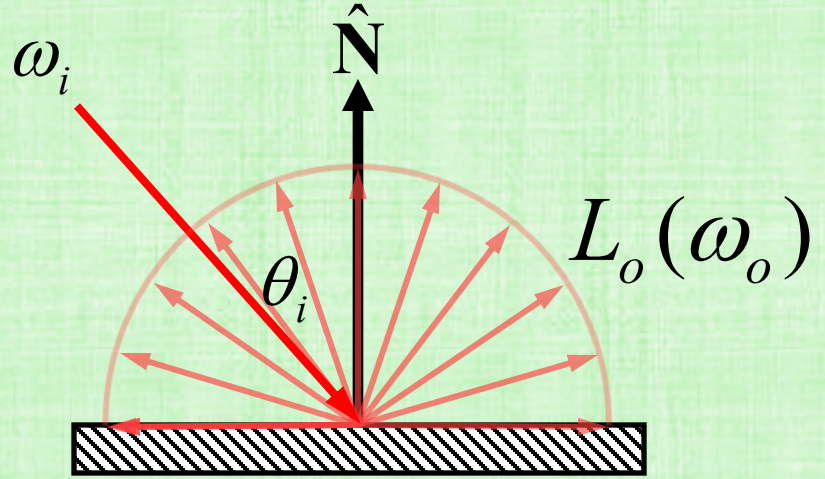


- This leads to a constant BRDF that doesn't depend on incoming/outgoing directions:

$$BRDF(\omega_i, \omega_o) = k_d$$

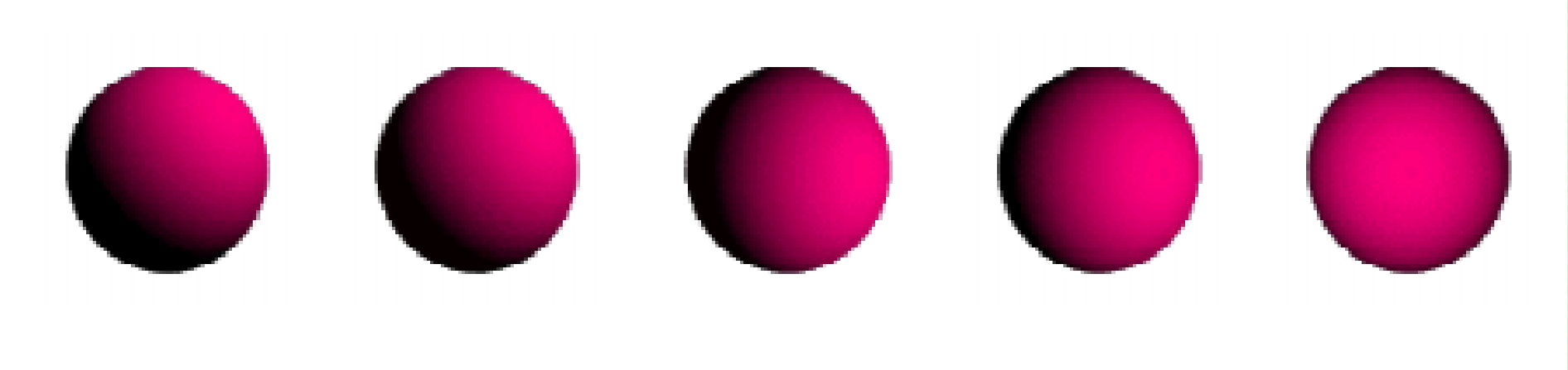
- Then,

$$L_o = k_d \cos\theta_{light}\, \tilde{I}_{light}$$



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
- A good approximation of diffuse/dull/matte surfaces (such as chalk)



- A diffuse object with 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:

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

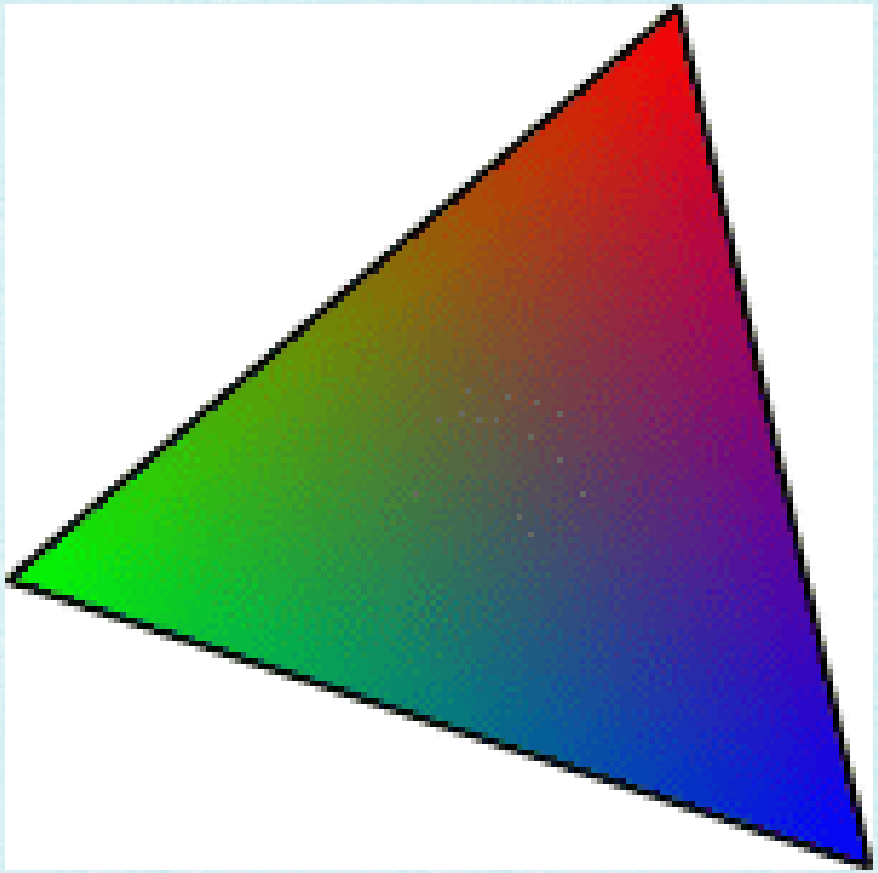
Ambient Lighting

- Useful for adding light in regions obscured from all the light sources
- Ignores the incident light direction (drops the cosine term)
- An ambient light (I_a^R, I_a^G, I_a^B) hitting an object with ambient color(k_a^R, k_a^G, k_a^B) results in:

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

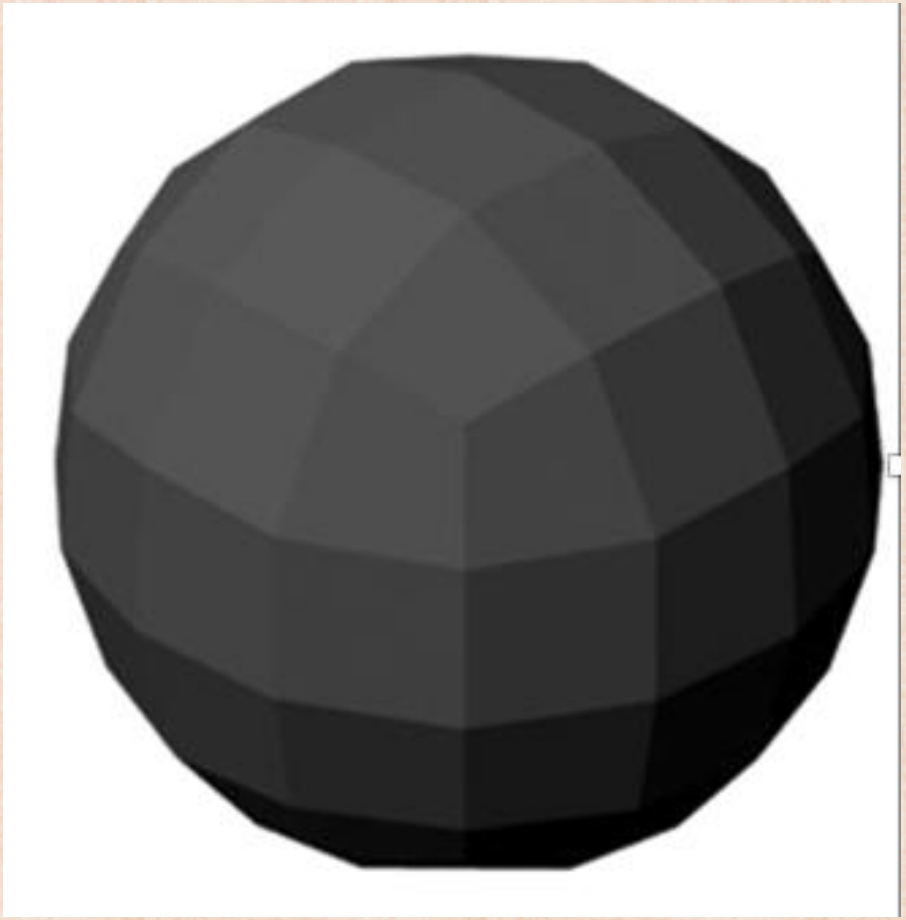
Vertex Colors

- The k_a and k_d values are stored on the vertices of triangles: e.g. 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, use $k = \alpha_0 k_0 + \alpha_1 k_1 + \alpha_2 k_2$ to interpolate k_a^R, k_a^G, k_a^B k_d^R, k_d^G, k_d^B to the point p



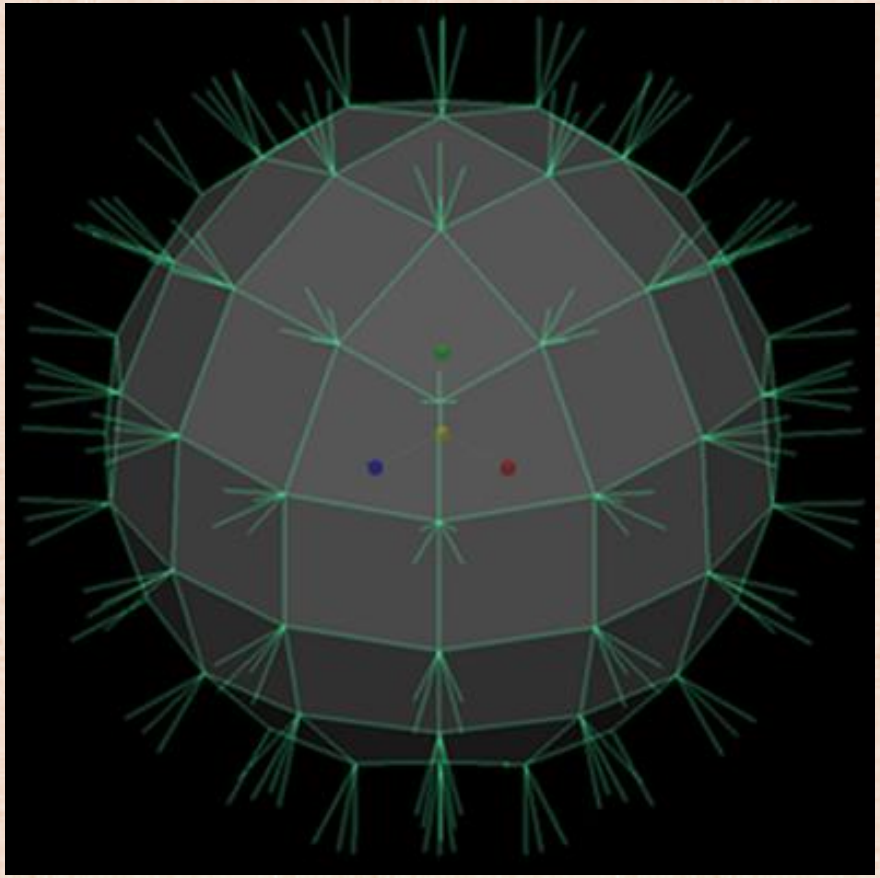
Flat Shading

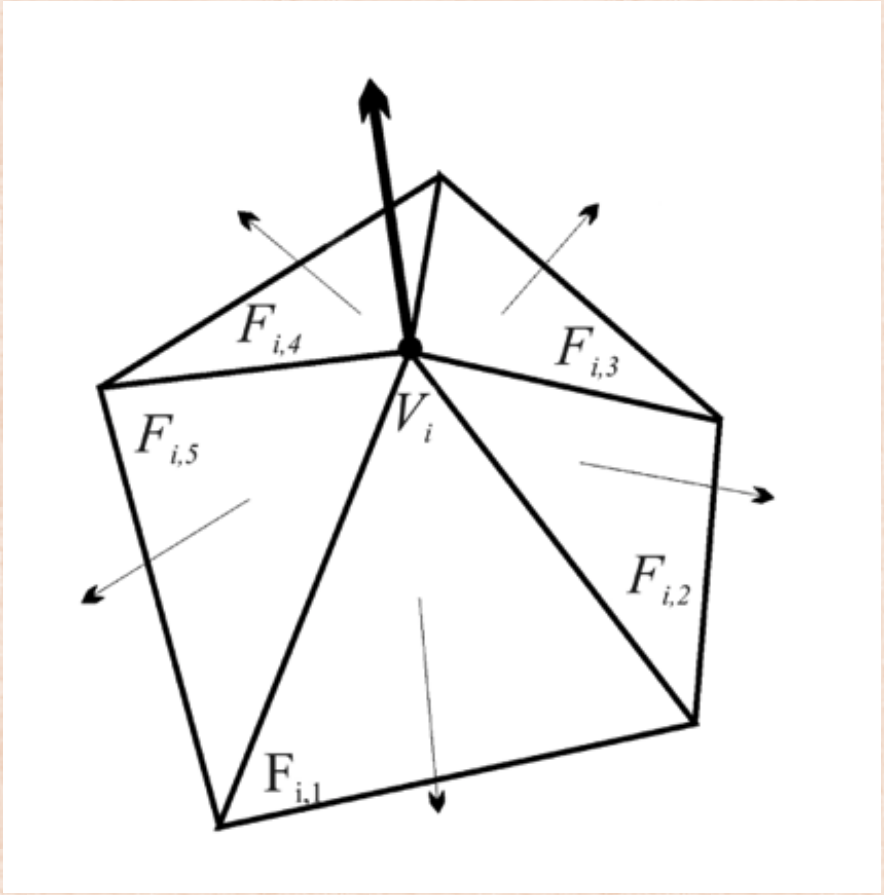
- The change in normal direction from one triangle to another allows us to see individual triangles (as expected)
- This can be alleviated by using a lot more triangles, but that's computationally expensive



Averaged Vertex Normals

- Each vertex belongs to a number of triangles, each with their own normal
- Averaging those normals (weighted averaging, based on: area, angle, etc.) gives a unique normal for each vertex

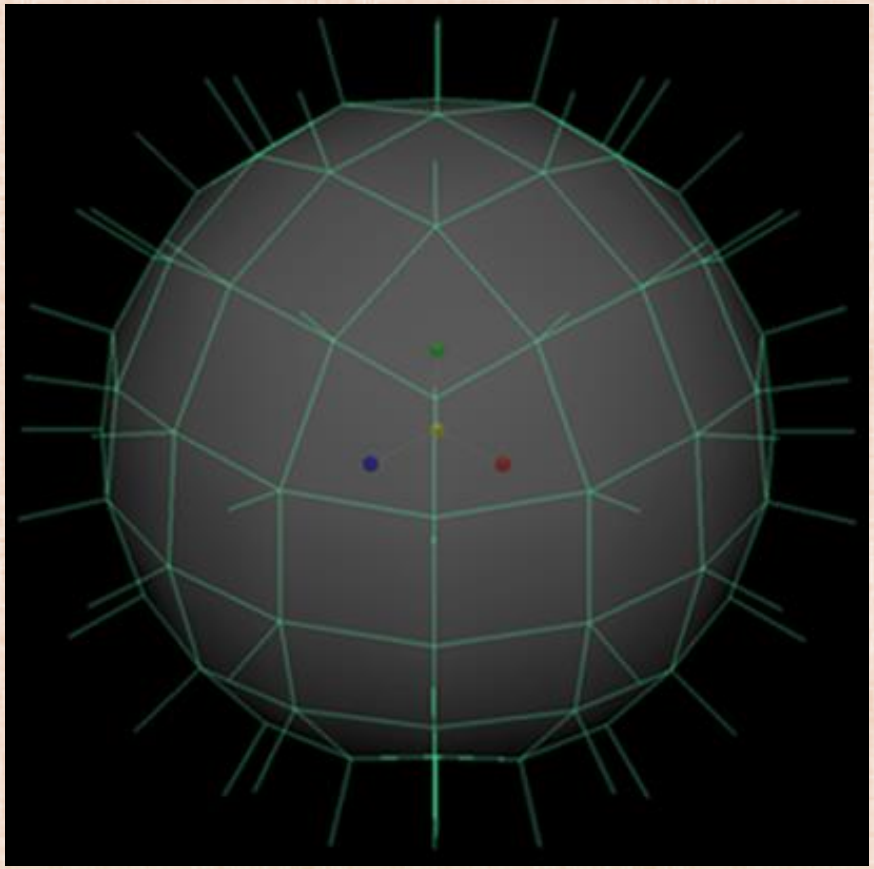


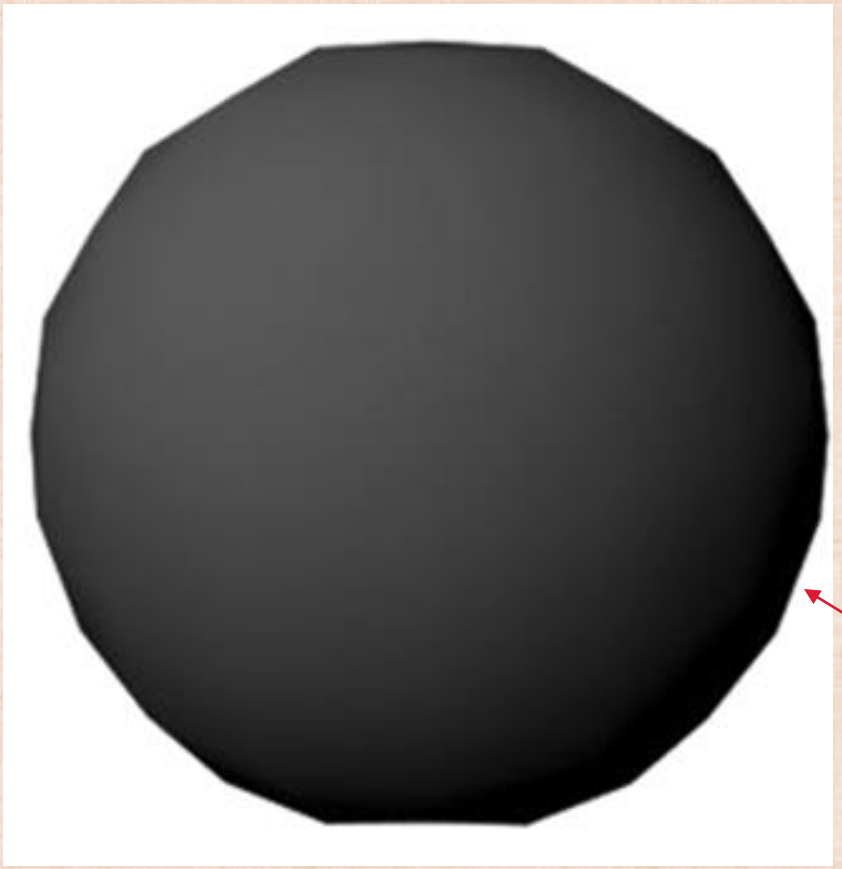


Smooth Shading

- Use barycentric weights to interpolate averaged vertex normals to the interior of a triangle:

$$\hat{N}_p = \frac{\alpha_0\hat{N}_0 + \alpha_1\hat{N}_1 + \alpha_2\hat{N}_2}{\left\|\alpha_0\hat{N}_0 + \alpha_1\hat{N}_1 + \alpha_2\hat{N}_2\right\|_2}$$

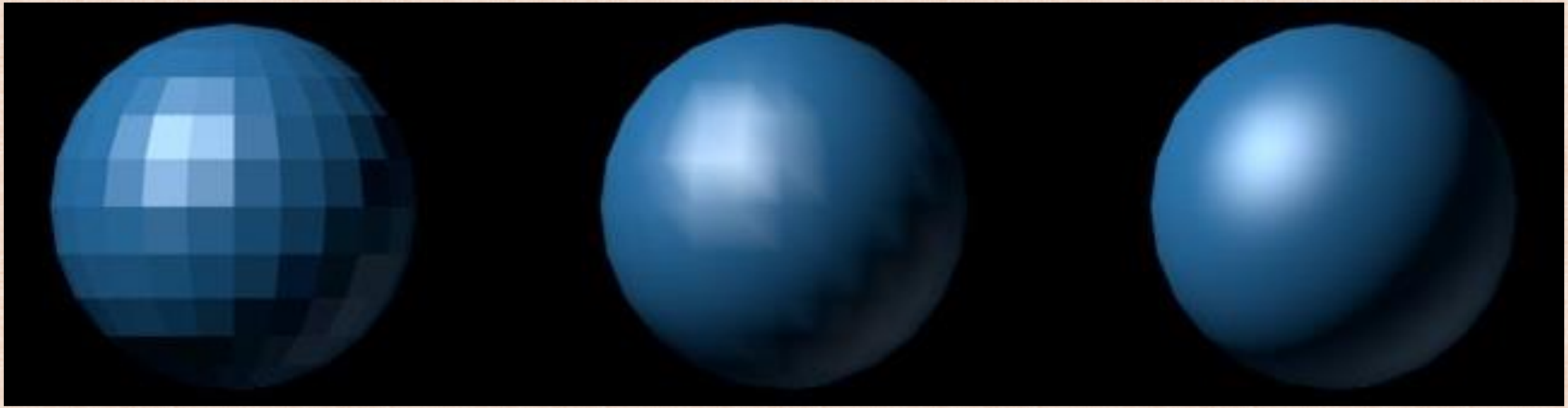




faceted silhouette

Flat vs. Gouraud vs. Phong (Shading)

- Flat: use the actual normal, i.e. the real geometry (the triangles can be seen, as expected)
- Gouraud: use averaged vertex normals to compute a color at each vertex; then, interpolate those vertex colors to the triangle interior
- Phong: interpolate averaged vertex normals to the triangle interior for shading (smooth shading)

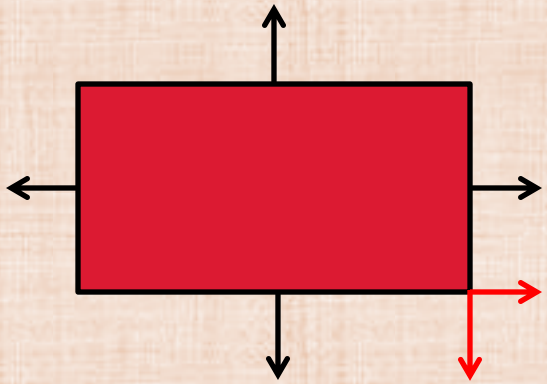


flat Gouraud Phong

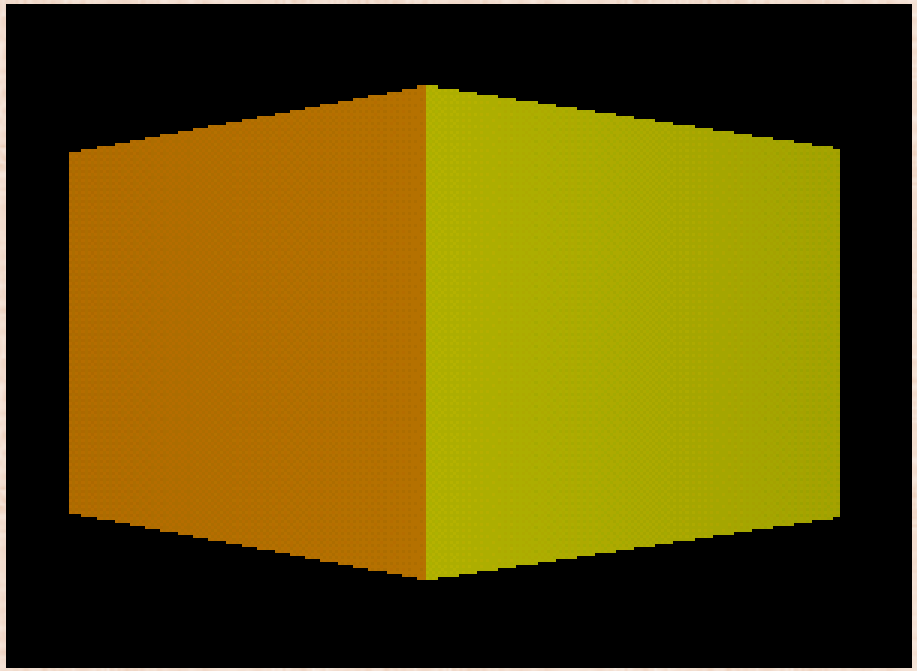
*Don't mix up Phong shading with the Phong reflection

Edges and Corners

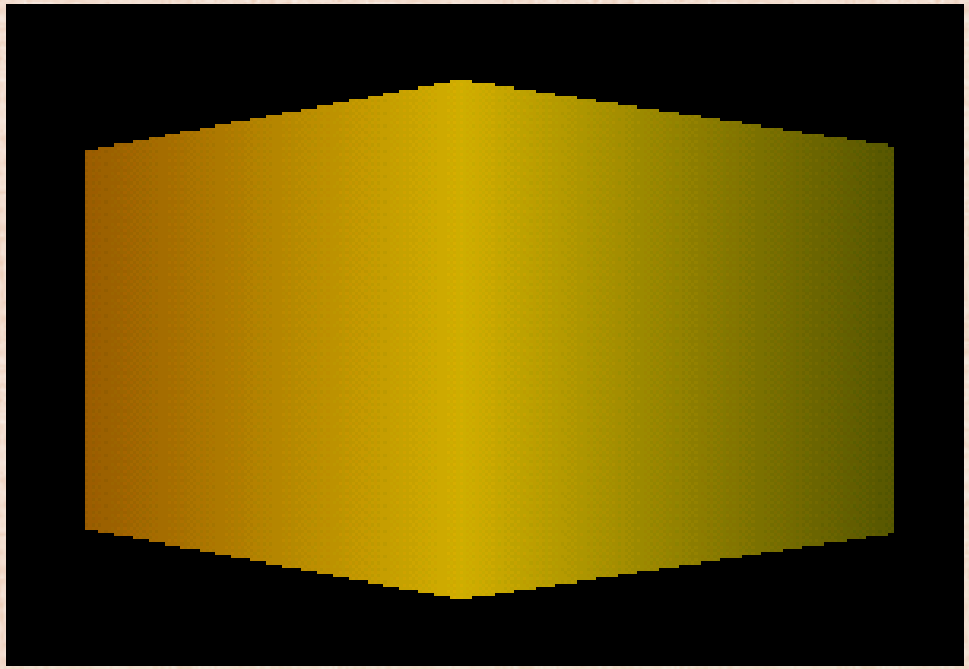
- Normals should not be averaged at edges/corners
- Note: different types of shading can be used on different parts of the same object (in fact, the same triangle may require both flat and smooth shading)



What should the normal be at the corner?



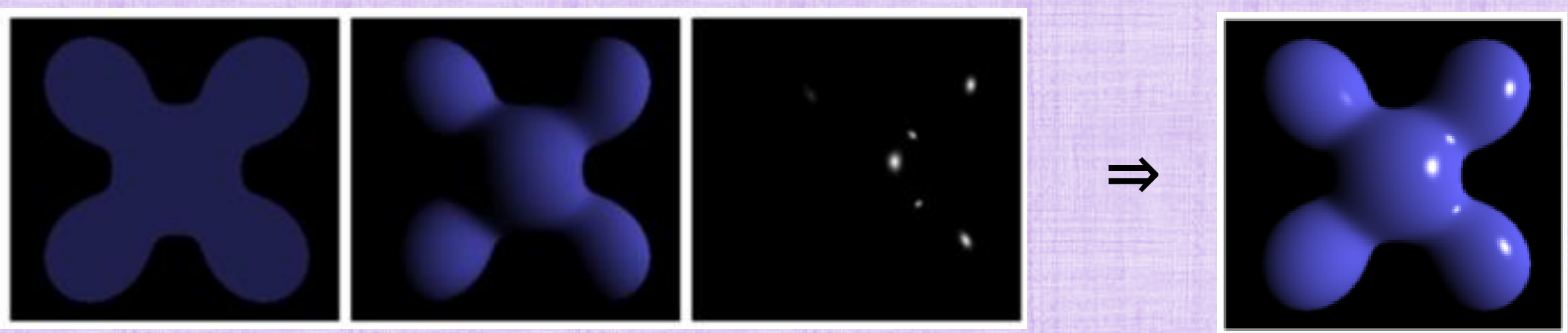
flat



smooth

Phong Reflection Model

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



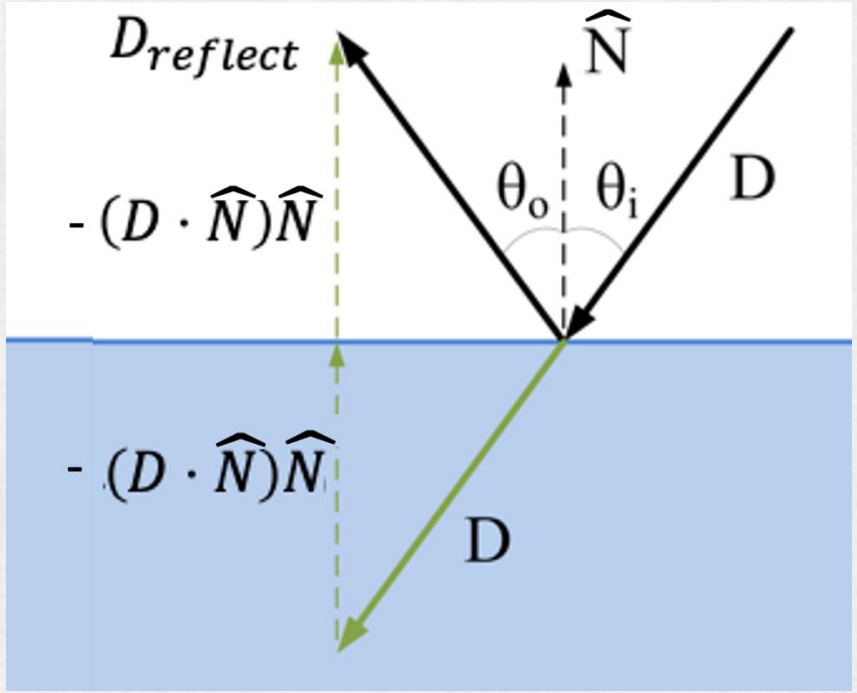
Ambient Diffuse Specular Final Image

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

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

Recall: Reflected Ray

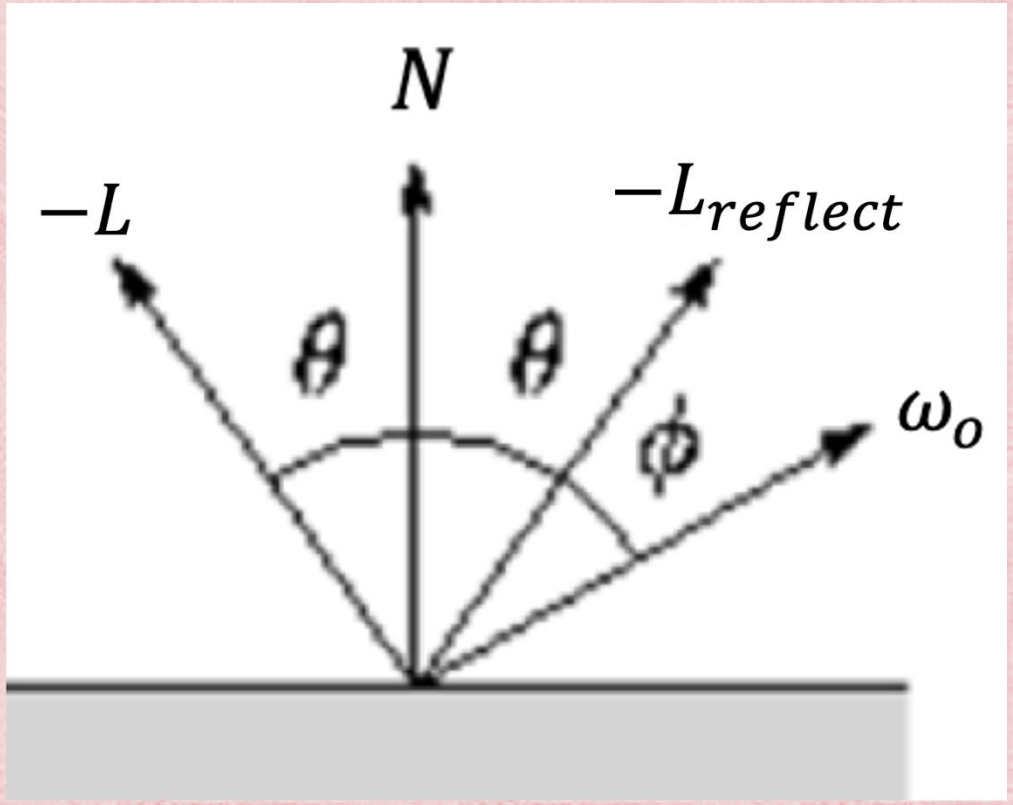
- Given an incoming ray $R(t) = A + Dt$, and (outward) unit normal $\hat{N}$, the angle of incidence is defined via $D \cdot \hat{N} = -\|D\|_2 \cos\theta_i$
- Mirror reflection: incoming/outgoing rays make the same angle with $\hat{N}$, i.e. $\theta_o = \theta_i$
  - Note: all the rays and the normal are all coplanar
- Reflected ray direction: $D_{reflect} = D - 2(D \cdot \hat{N})\hat{N}$
- Reflected ray: $R_{reflect}(t) = R_o + D_{reflect}t$



Specular Highlights

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

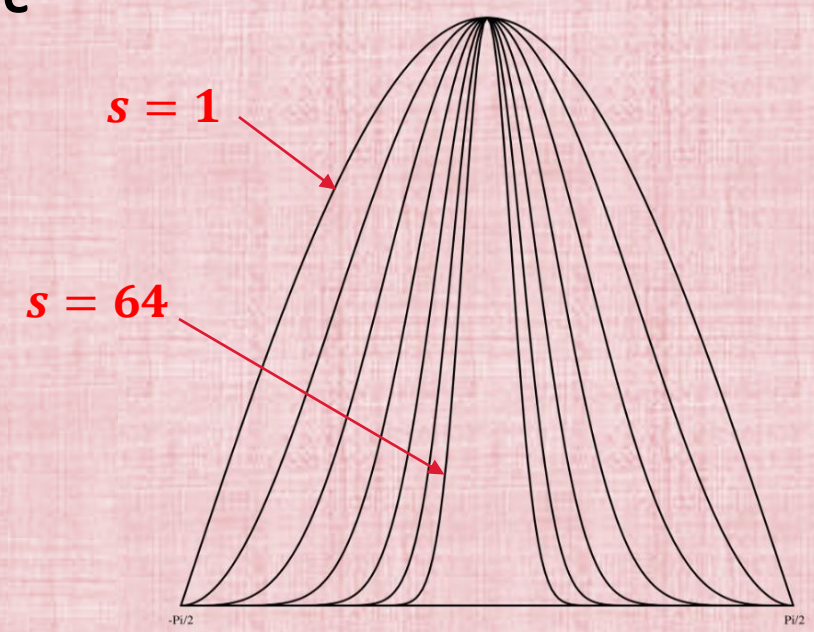
$$L_o(\omega_o) = k_s \, \tilde{I}_{\text{light}} \max\left(0, -\hat{\omega}_o \cdot \hat{L}_{reflect}\right)^s$$

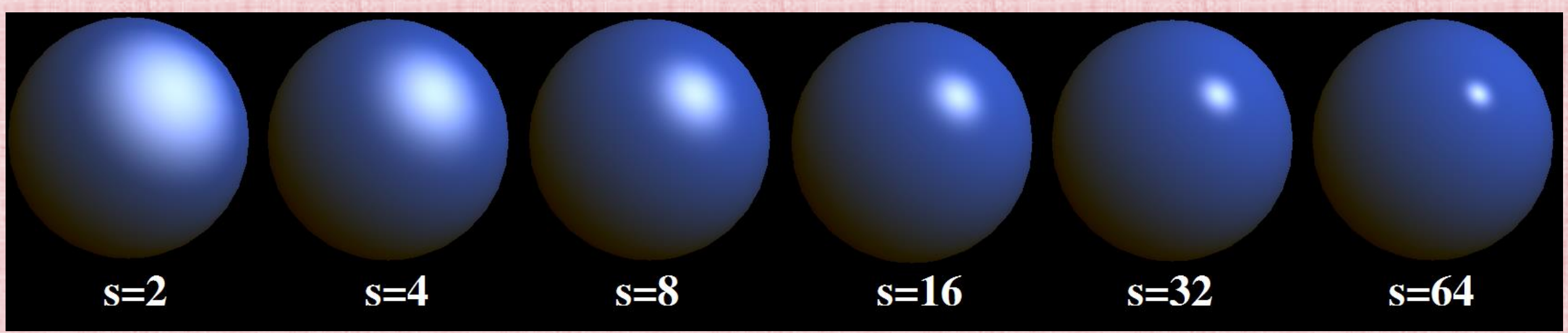


Shininess Exponent

- A shininess exponent s determines the size of the lobe
- A larger s gives a smaller highlight

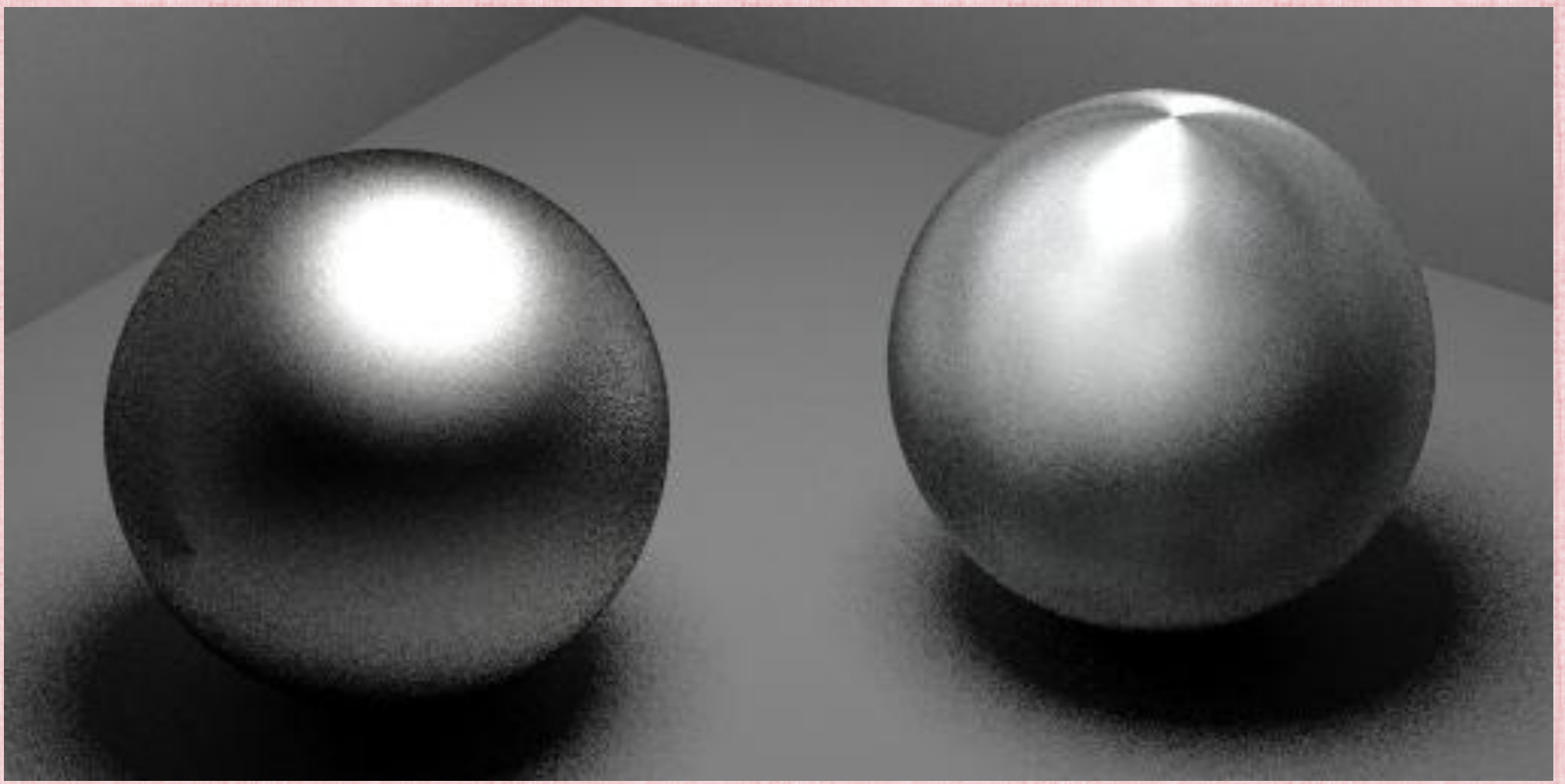
$$L_o(\omega_o) = k_s \, \tilde{I}_{\text{light}} \max\left(0, -\hat{\omega}_o \cdot \hat{L}_{reflect}\right)^s$$





Anisotropic Specular Highlights

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



isotropic

anisotropic