CS174A : Introduction to Computer Graphics

Royce 190
TT 4-6pm

Scott Friedman, Ph.D
UCLA Institute for Digital Research and Education

Lighting and Shading

- Now that we have moved into 3D…
  - We need to talk about shading.
  - In order to enhance the effect of dimensionality.
  - Color in real world is rarely flat and uniform.
  - We would like to approximate how light interacts.
    - Both by the type of light source and how the light reflects off a surface.

Lighting and Shading

- This means we need to describe
    - The various types of light sources available
    - The properties of a surface that will affect how light reflects off if it.

    - We will develop a model of how light interacts
    - We will restrict ourselves to local interaction
    - Global light models are, generally, too computationally intensive for interactive graphics.

Lighting and Shading

- What do we mean by Global?
- We could use the physics of light.
  - Surfaces either emit, absorb or reflect light.
  - Light reflects, scatters, bleeds, etc.
  - Unfortunately…
    - There is no analytical solution fast enough – even for offline rendering – to be useful.
    - Even approximate solutions are too slow for us…
      - » Radiosity (thermodynamics based)
      - » Ray Tracing (light ray based)
    - GPUs have sped some of this up but only for simple to moderately complex scenes.

Lighting and Shading

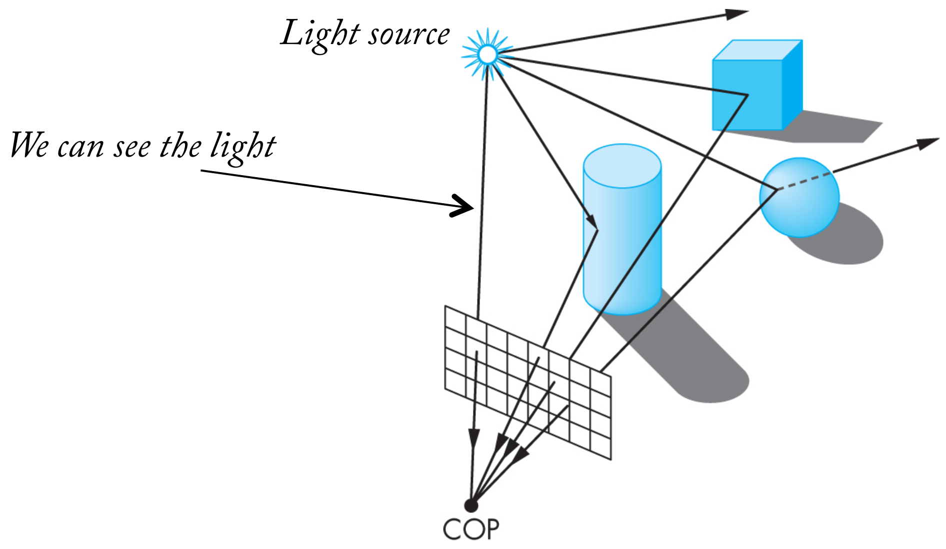
- We will simplify things.
  - Single interactions only.
  - Between light source and a surface.

  - There are two parts to this problem we need to understand in order to solve.
    - We need a model for a light source(s)
    - We need a model for how a surface can reflect that light source.

UCLA

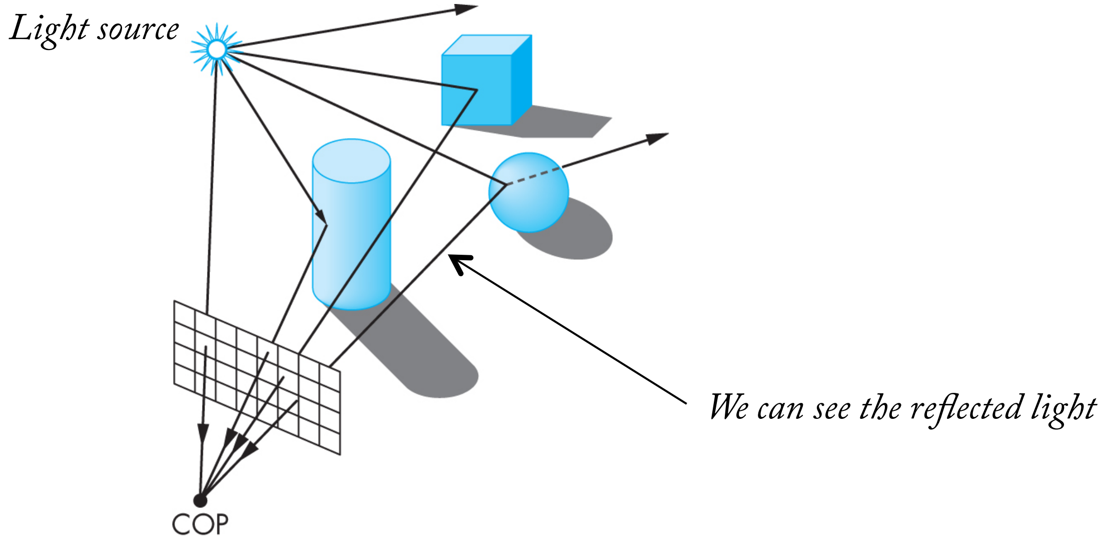
Lighting and Shading

- First, what is going on?
  - We can see the light of the source.
    - Which would be the color of the source itself.



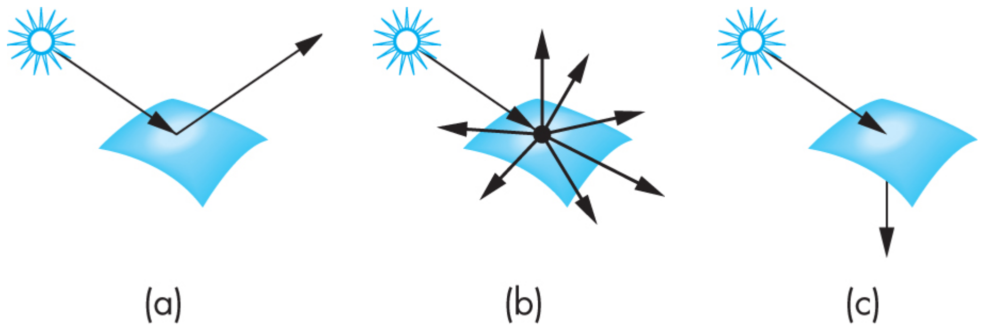
Lighting and Shading

- First, what is going on?
  - We can also see the reflection of a light source.
    - The reflected light off a surface after some of the light has been absorbed or affected in some way by the material.



Lighting and Shading

- Types of surfaces
  - Specular (a)
    - Most light is reflected in a narrow range (like a mirror)
  - Diffuse (b)
    - Light is reflected is all directions (very flat appearance)
  - Translucent (c)
    - Light is refracted, some may be reflected



Lighting and Shading

- Light sources types (models)
    - Ambient
    - Point
    - Spotlight
    - Distant / Infinite

    - They each have color and luminance (brightness)
        - Both are represented with an RGB value.

Lighting and Shading

- Light sources – Ambient
  - Uniform illumination.
    - Think of a cloudy day.
    - Or, indirect lighting in your house.
  - Light is scattered in all directions.
  - We model our ambient light with a luminance, I_a.
  - Every point in our scene will receive the same illumination from I_a.
    - Every surface *may* reflect ambient light differently, however.

UCLA

Lighting and Shading

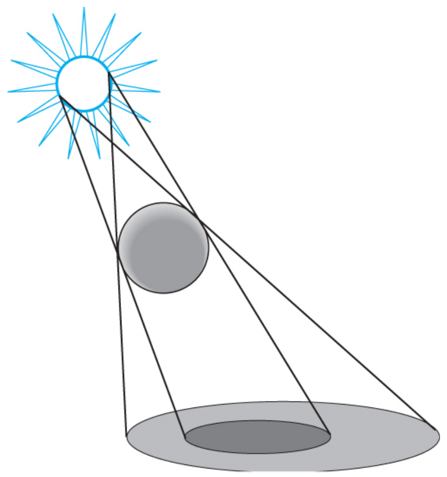
- Light sources – Point
    - Emits light equally in all directions.
    - Has a luminance like ambient light
    - Has a location as well, $I(p_0)$.
    - The amount of light received by a point in our scene is proportional to the distance from the point light source.
        - The terms ***a***, ***b*** and ***c*** are used to soften the light.

$$d = |p - p_0|^2$$

$$i(p, p_0) = \frac{1}{a + bd + cd^2} I(p_0)$$

Lighting and Shading

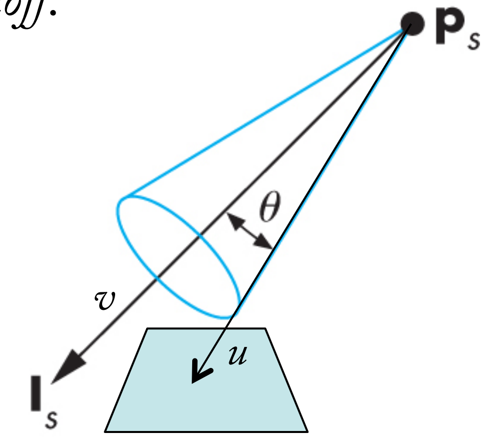
- Light sources – Point
  - Most *real* point sources are not points.
  - Most have a *finite* size
    - An object's shadow casts an *umbra* and *penumbra.*
    - Our idealized point source model does not automatically cause an umbra or penumbra.



UCLA

Lighting and Shading

- Light sources – Spotlight
    - Like point lights, they have a position
    - Unlike point lights, are *directional*.
    - Have a distribution of light defined by an angle.
        - The intensity of the light is attenuated by the magnitude of the angle between the direction vector and a point on the surface.
        - Called the *falloff*.

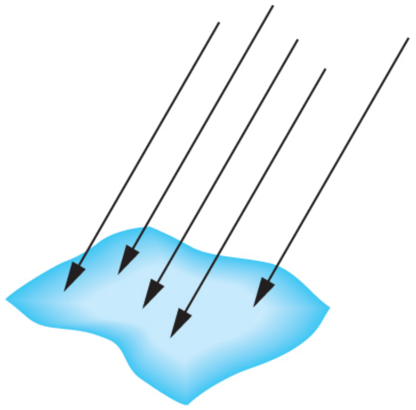


$$\cos\theta = u \bullet v$$

UCLA

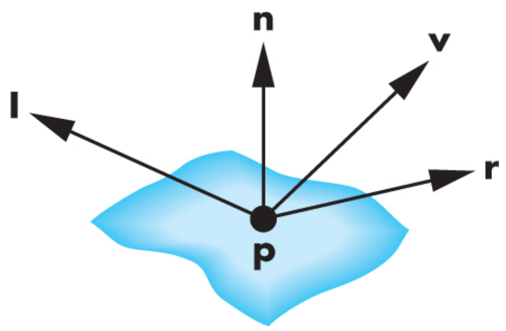
Lighting and Shading

- Light sources – Distant / Infinite
  - When a light source is far enough away. (Sun)
  - All light rays are effectively parallel to each other.
  - We use a direction vector in place of p_0.
  - All surface points use that same vector.
    - No need to recompute the incidence vector for each point on a surface using p_0.



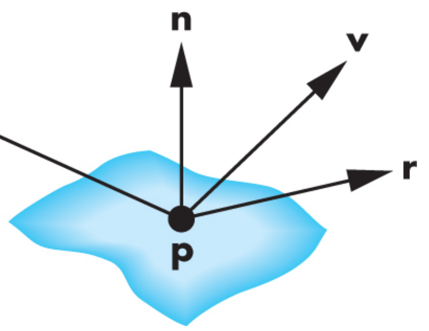
Lighting and Shading

- Shading – The Phong Reflection Model
  - Rather than an exact physical model
  - We are using an efficient approximation.
  - The Phong model uses four vectors.
    - These enable the calculation of a color value for any point on a surface.



Lighting and Shading

- Shading – The Phong Reflection Model
  - The Phong model's four vectors are
    - **n**, normal at point **P** on the surface.
    - **v**, vector from point **P** to the eye.
    - ***I***, vector from the point **P** to the light source (direction)
    - **r**, vector of the perfectly reflected ray from ***I***.
      - » **r** is computed from **n** and ***I***.



Lighting and Shading

- Shading – The Phong Reflection Model
    - The Phong supports three types of light interaction
        - Ambient
        - Diffuse
        - Specular
    - Just like the light sources
        - We store these three values (colors) for a surface.
        - More generally we define a *material* and associate it with a surface.
        - So, a surface can have more than one color.

Lighting and Shading

- Shading – The Phong Reflection Model
  - To compute the light (color) value at a point on a surface
  - We sum contributions from
    - Each light source interacting with the surface, *R* value
    - and the global ambient term.

$$I = \sum_i (I_{a_i} + I_{d_i} + I_{s_i}) + I_a \quad \text{where}$$

$$(I_{a_i} + I_{d_i} + I_{s_i}) = (L_{a_i} R_a + L_{d_i} R_d + L_{s_i} R_s)$$

Lighting and Shading

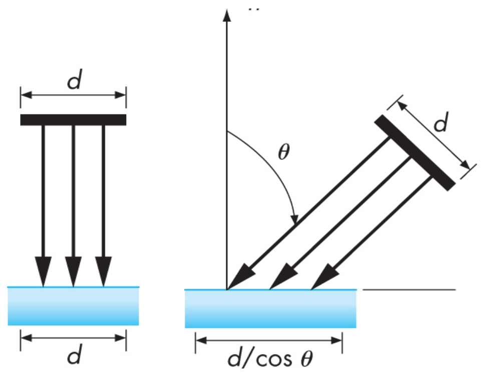
- Shading – The Phong Reflection Model
  - Ambient Reflection
    - The intensity of ambient light at each point on a surface is the same.
    - The amount of ambient light that is *reflected* is given by the coefficient k_a.
    - For any light source the resulting reflected light intensity is

$$I_a = k_a L_a \quad \text{where} \quad 0 \leq \mathrm{k}_a \leq 1$$

UCLA

Lighting and Shading

- Shading – The Phong Reflection Model
  - Diffuse Reflection
    - There is no preferred direction of reflection
    - Lambert's law is useful to model the behavior.
    - We only see the vertical component of the reflected light.
    - But as the angle becomes larger the light gets dimmer.
    - This happens because the light is spread out over a larger area.



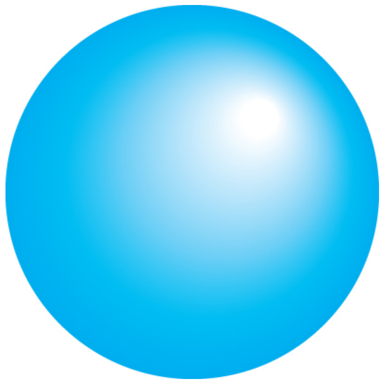
Lighting and Shading

- Shading – The Phong Reflection Model
  - Diffuse Reflection
    - The direction of the light source **l** and the surface normal **n** determine the amount of reflected light.
    - Once again a reflection coefficient k_d can be factored in.
    - We also want to include the quadratic attenuation term to account for the surface's distance from the light.
    - The max() is to handle when the light is below the horizon.

$$I_d = \frac{k_d}{a + bd + cd^2} \max((\mathbf{l} \bullet \mathbf{n}) L_d, 0)$$

Lighting and Shading

- Shading – The Phong Reflection Model
    - Specular Reflection
        - Adding some "shininess" to a smooth surface.
        - Reflection of part of the light source itself.
            - The smoother the surface the more concentrated it is
            - A mirror is a perfectly specular surface.



Lighting and Shading

- Shading – The Phong Reflection Model
  - Specular Reflection
    - – Phong only approximates the effect (can't do a mirror)
    - – Depends on the angle between the perfect reflection vector **r** and the direction to the viewer **v**.
    - – k_s, is the reflection coefficient of the specular reflection.
    - – $\propto$, is the shininess coefficient which controls how narrow or broad the reflection appears.

$$I_s = k_s L_s \max((\mathbf{r} \bullet \mathbf{v})^{\alpha}, 0)$$

Lighting and Shading

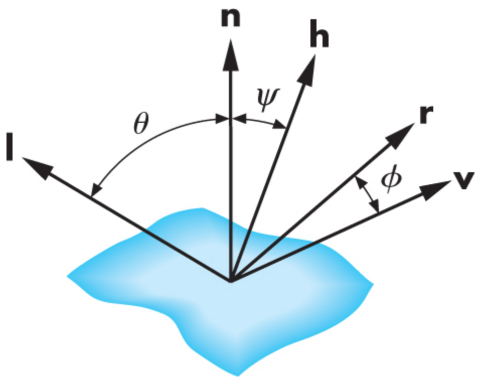
- Shading – The Phong Reflection Model
  - Full Phong Model
  - Combining the diffuse, specular and ambient terms.
  - Computed for each light source in the scene.

$$I = \frac{1}{a + bd + cd^2}(k_d L_d \max(\mathbf{l} \bullet \mathbf{n}, 0) + k_s L_s \max((\mathbf{r} \bullet \mathbf{v})^\alpha, 0)) + k_a L_a$$

$$\mathbf{l} = 2(\mathbf{l} \bullet \mathbf{n})\mathbf{n} - 1.$$

Lighting and Shading

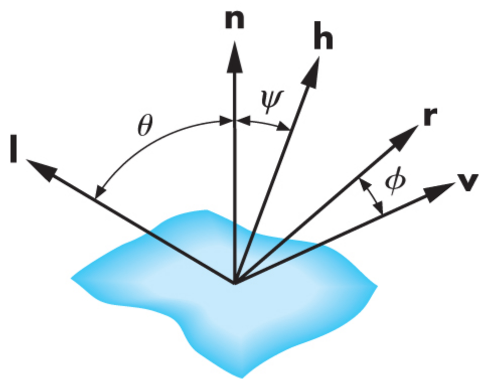
- Shading – The Phong Reflection Model
  - Blinn-Phong Model
    - Normally we have to *compute* **r**, the reflection vector, for every point on the surface.
    - When **l, n, r** and **v** are in the same plane the half-way vector **h** can be used to avoid computing **r**.



$$\mathbf{h} = \frac{\mathbf{l} + \mathbf{v}}{|\mathbf{l} + \mathbf{v}|}$$

Lighting and Shading

- Shading – The Phong Reflection Model
  - Blinn-Phong Model
    - The angle between **n** and **h** is smaller than the angle between **r** and **v**, however.
    - This means the shininess coefficient, ∝, has to be raised to get approximately the same result as the pure Phong.



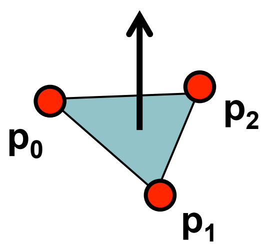
$$\phi = 2\psi$$

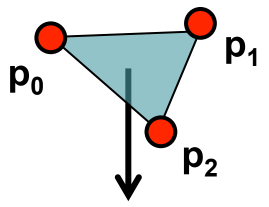
Lighting and Shading

- Shading – The Normal
  - For flat surfaces we mean the normal to a plane.
  - This is a vector we have seen already.
  - For three non-collinear points, the normal is

$$\mathbf{n} = (\mathbf{p}_2 - \mathbf{p}_0) \times (\mathbf{p}_1 - \mathbf{p}_0).$$

  - The order we consider the points *matters*.
  - It will *determine* the direction of the normal **n**.





Lighting and Shading

- Shading – The Normal
  - For curved surfaces, how the normal is computed depends on how the surface is represented.
  - Take the unit sphere at the origin.
  - The normal at any point on the sphere is simply

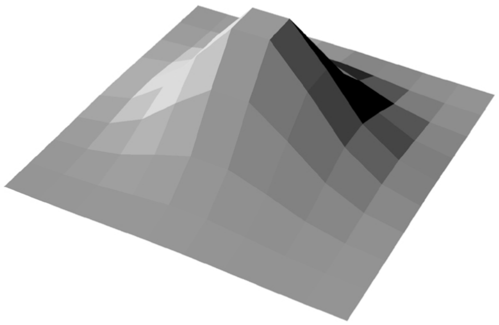
$$\mathbf{n} = \mathbf{p}$$

  - Often, in OpenGL we are approximating the normal using nearby points and planes.
  - This is because we can only represent objects, including surfaces, with these basic primitives.

UCLA

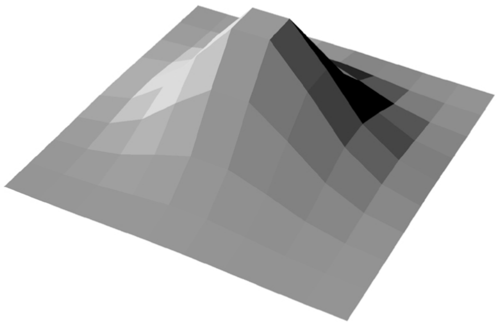
Lighting and Shading

- Shading – flat
  - Vectors **l**, **n** and **v** vary as we move across a surface.
  - However, if the surface is flat, and the light source and viewer are far these vectors can be held constant.
  - Then we have *flat shading* where every point on a surface primitive receives the same shading.



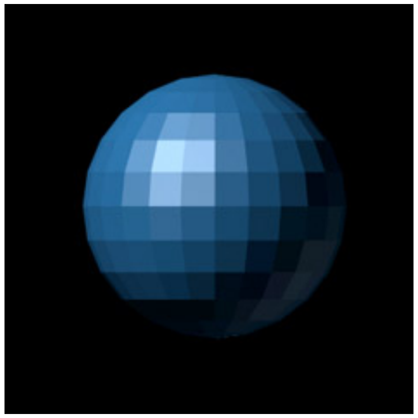
Lighting and Shading

- Shading – flat
  - This can look funny for a non-flat surface or if the light source or viewer are not far away.
  - But, depending on how small the elements that make up the surface are it may not be as visible.
  - Squint your eyes and look at this surface.



Lighting and Shading

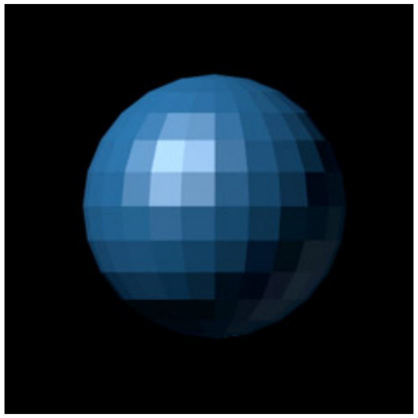
- Shading – flat
  - Your brain has an amazing ability to fill in what *should* be there – including interpolating the colors.
  - Flat shading is also called *per primitive* in OpenGL since we are applying a single color (light) value to the entire primitive.



UCLA

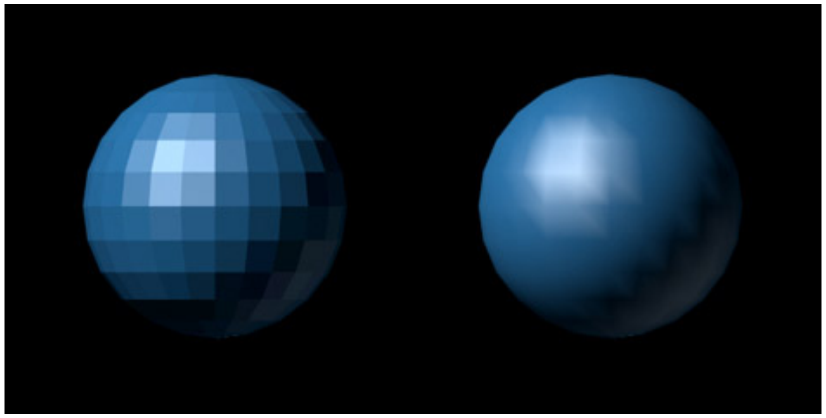
Lighting and Shading

- Shading – smooth
  - By providing color and normal information *per vertex* we can achieve a much smoother shading result.
  - Here we let the rasterizer interpolate the color values across the surface of the primitive.
  - This requires us to think about how to compute the normal for a surface.



Lighting and Shading

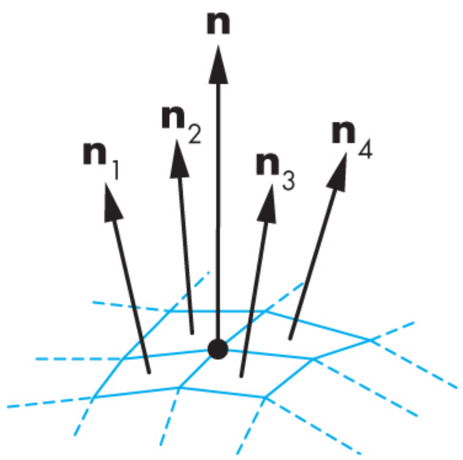
- Shading – smooth
  - *Gouraud* shading informs how we might define the normals we use.
  - Gouraud shading uses the average of the surface normals of the primitive they are part of.

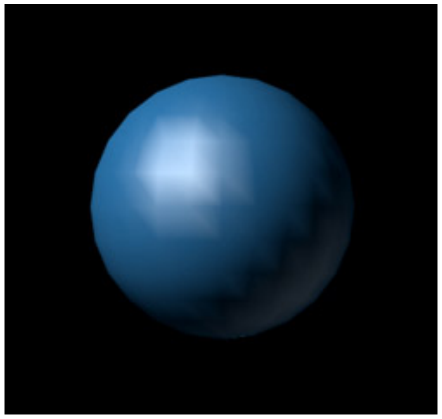


Lighting and Shading

- Shading – smooth
  - For a vertex that is shared by four surfaces we would compute the vertex normal as

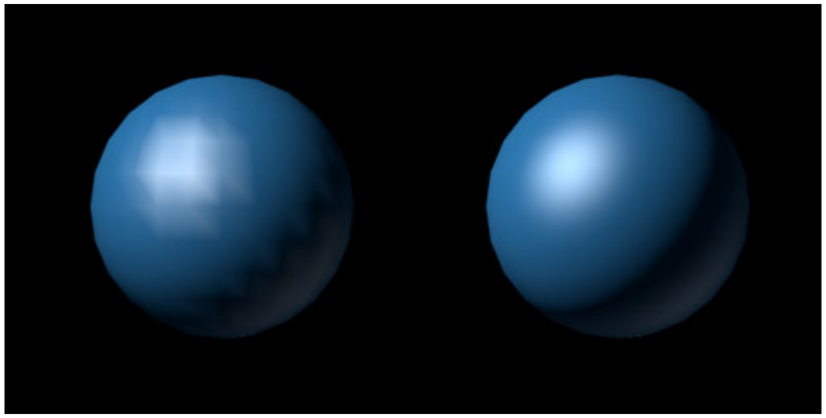
$$\mathbf{n} = \frac{\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4}{|\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4|}.$$





Lighting and Shading

- Shading – smooth
  - A variation takes Gouraud shading one step further.
  - *Phong shading* interpolates the normals themselves.
  - Phong shading requires us to compute lighting at the vertex or *fragment* level.



Lighting and Shading

- Shading – smooth