CptS 442/542 (Computer Graphics)
Unit 8: Lighting, Shading, and Local Illumination

Bob Lewis

School of Engineering and Applied Sciences
Washington State University

Fall, 2017

Motivation

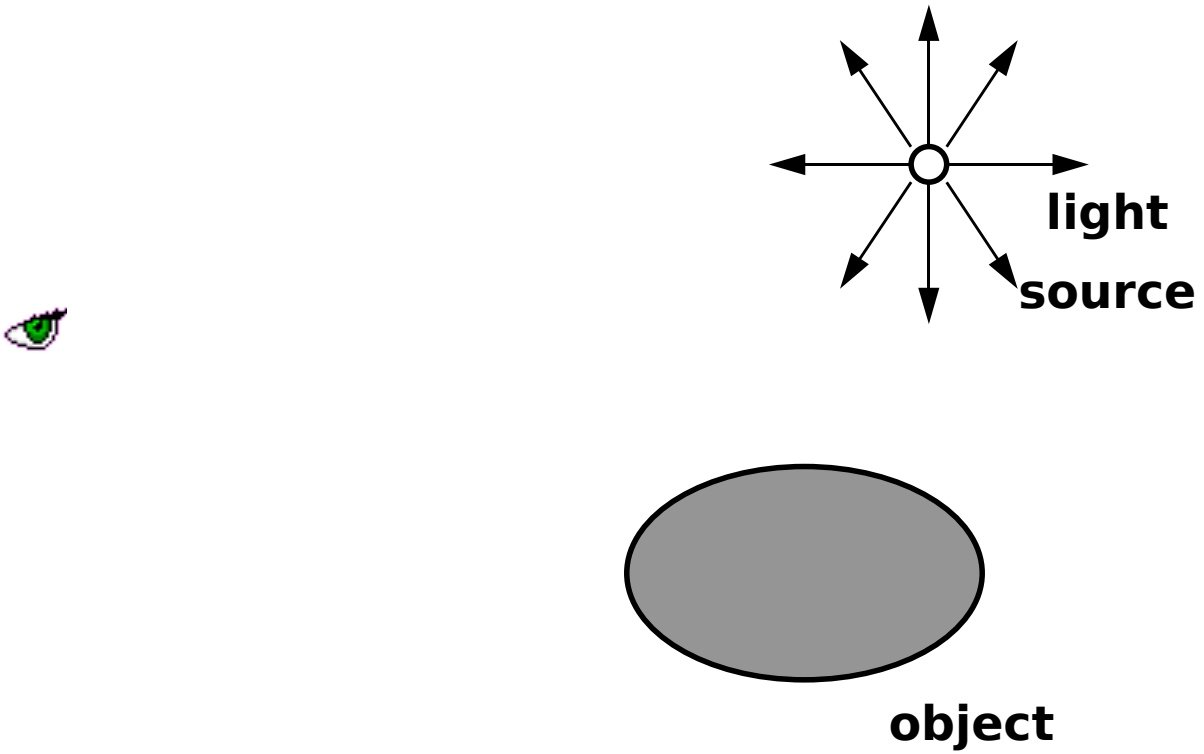
- lighting effects convey three-dimensionality
- to model the real world, we need to understand how light interacts with matter
- to write a custom shader on a GPU, we need to know how conventional shaders work

Shaded Images

- Allow color to vary from triangle to triangle or even vertex to vertex (interpolation).
- Enabled by raster display technology.
- Improvement on use of color and texture.
- Material properties (glossy vs. dull, etc.) can be modelled.
- Texture can be applied to simplify rendering.
- Shadows help with spatial cueing.
- Transparency and reflection can be added.
- Can render views with something like a real camera model.

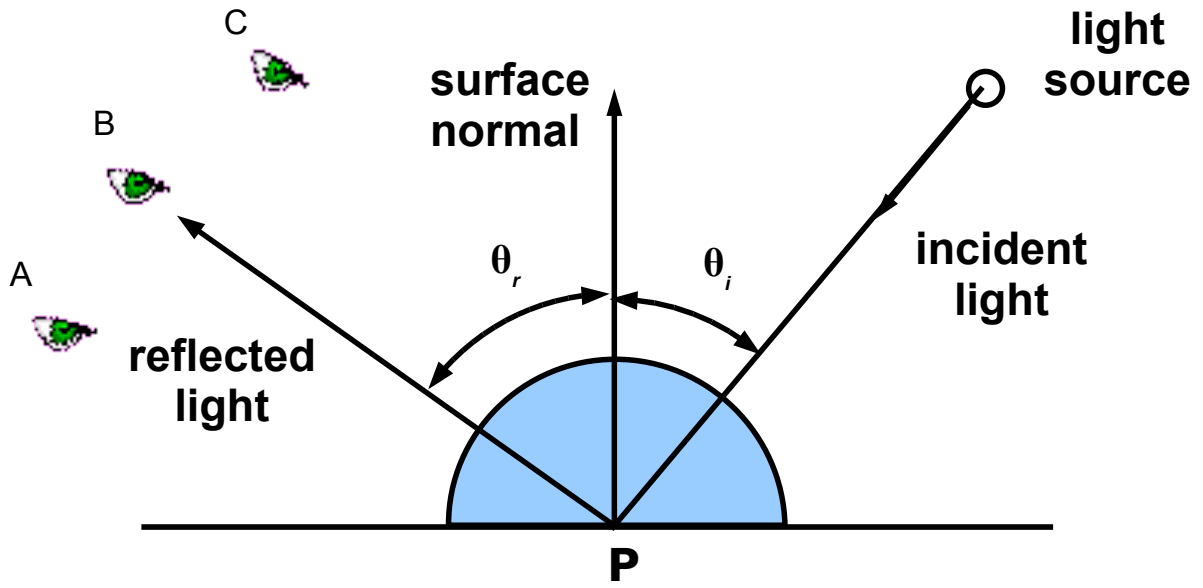
The Interaction of Light with Matter

Answer: One way is by studying a little physics.



(We'll *start* with physics, but add a few hacks.)

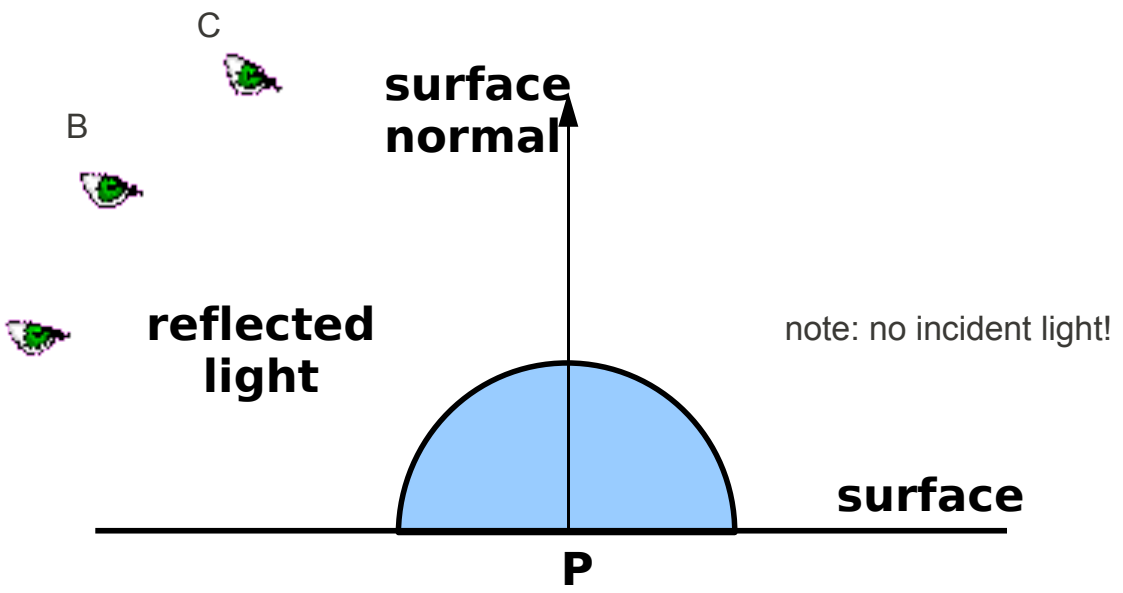
Polar Plot Reflection Terminology



- The blue solid is a polar plot of the amount of reflected light as a (constant, in this case) function of light (incident) and observer (reflected) directions $L(\theta_i, \phi_i, \theta_r, \phi_r)$.
- This function is actually a 4D function, but we take ($\phi_i = 0°$ and $\phi_r == 180°$).

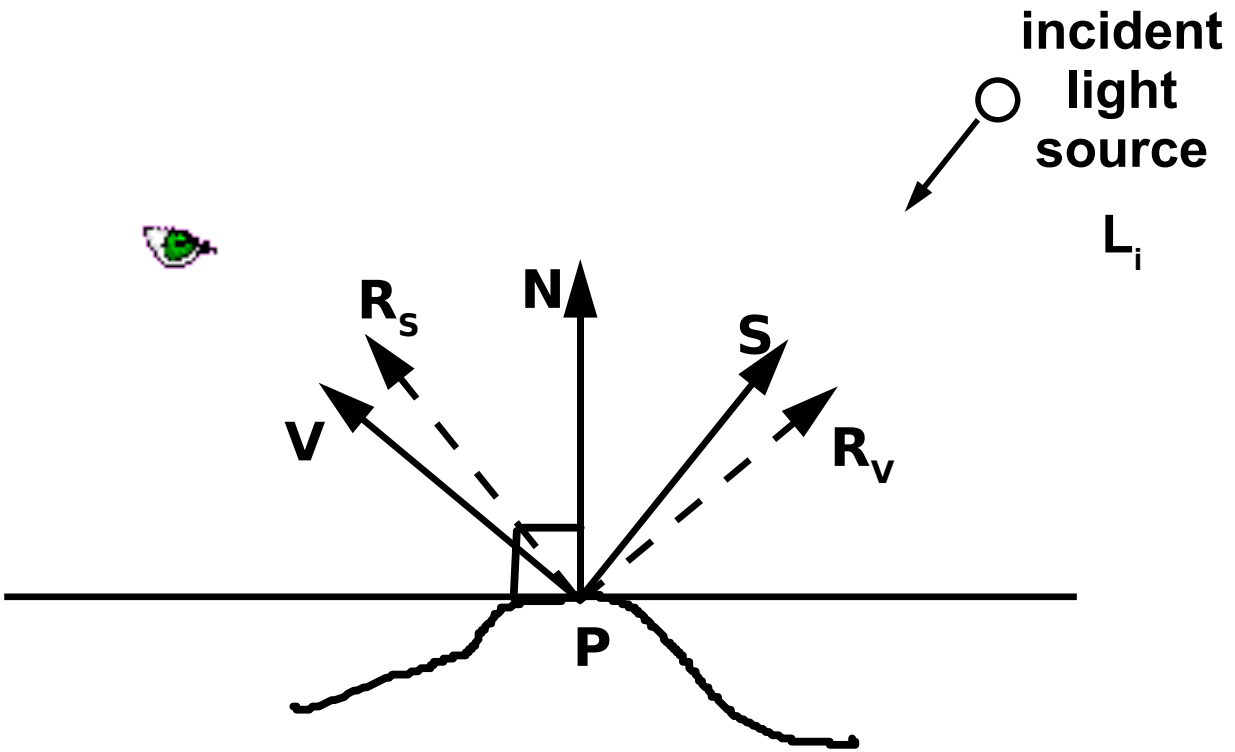
An Emissive "Shader"

An emissive shader has a constant color. It models a light source omnidirectional light.



Computing Reflection: Geometry

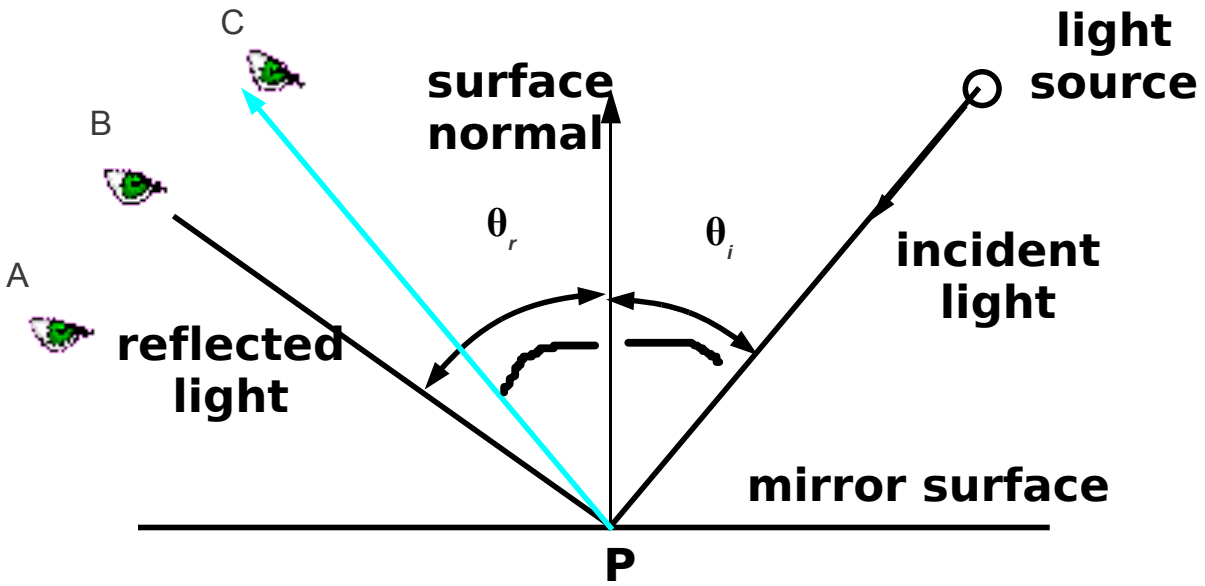
To mathematically model reflection, we need to compute the geometry of incident and reflected light at a surface:



How can we compute $\widehat{\mathbf{R}_{\vec{V}}}$?

Special Case: A Mirrored Surface

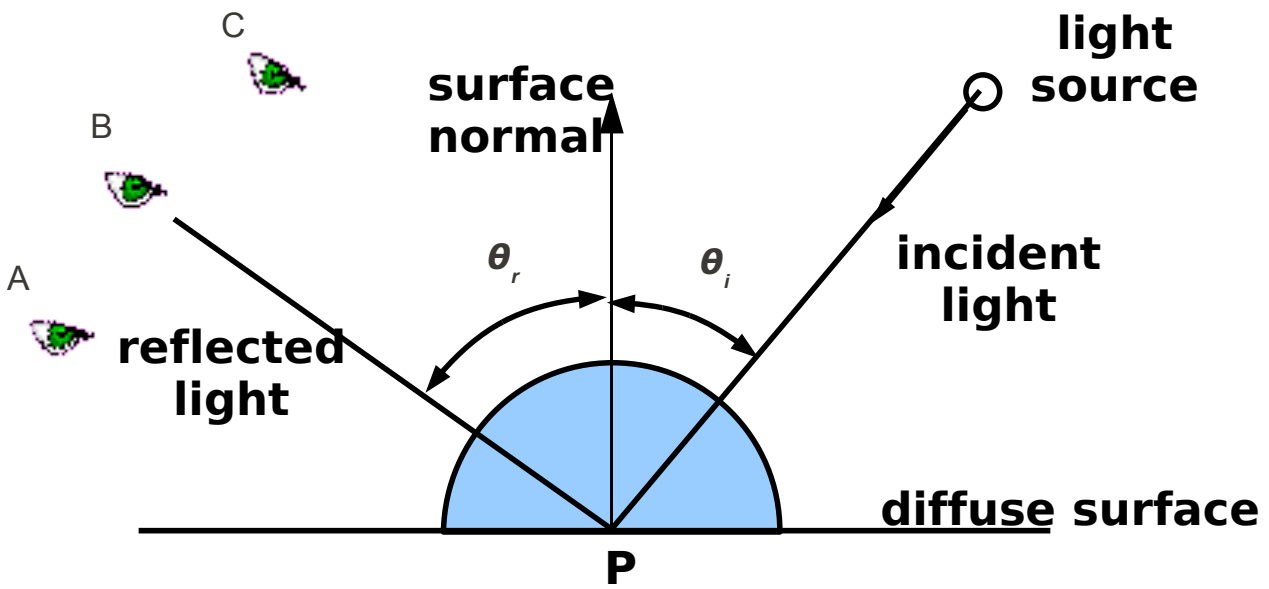
If our surface is a mirror, all light is reflected in a single direction! What is that direction? (Remember high school physics.)



This is not supported in the OpenGL *lighting* model, but there are ways, which we'll see in a later unit.

A Diffuse Shader

For a diffuse surface, the amount of light reflected depends on incident direction, but is independent of reflected direction:



The Diffuse Illumination Model

Dating back to Johann Heinrich Lambert (1728-1777), a Swiss mathematician, physicist, and philosopher.
This assumes that light is reflected *isotropically* (uniformly in all directions), but the amount of light reflected still depends on the incident direction, so

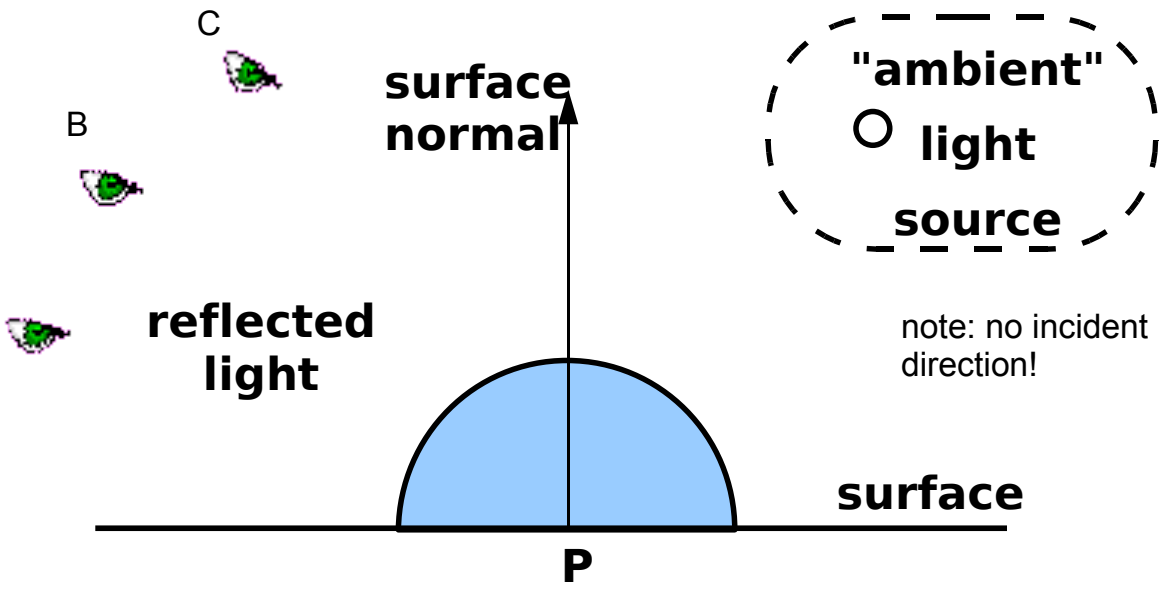
reflected radiance incident radiance

$$L_r = k_d \left(\widehat{\mathbf{S}} \cdot \widehat{\mathbf{N}} \right) L_i$$

where k_d is the *diffuse reflectivity*.
If the dot product is ever negative, the result is taken to be zero: the light is "below the horizon".

An Ambient Shader

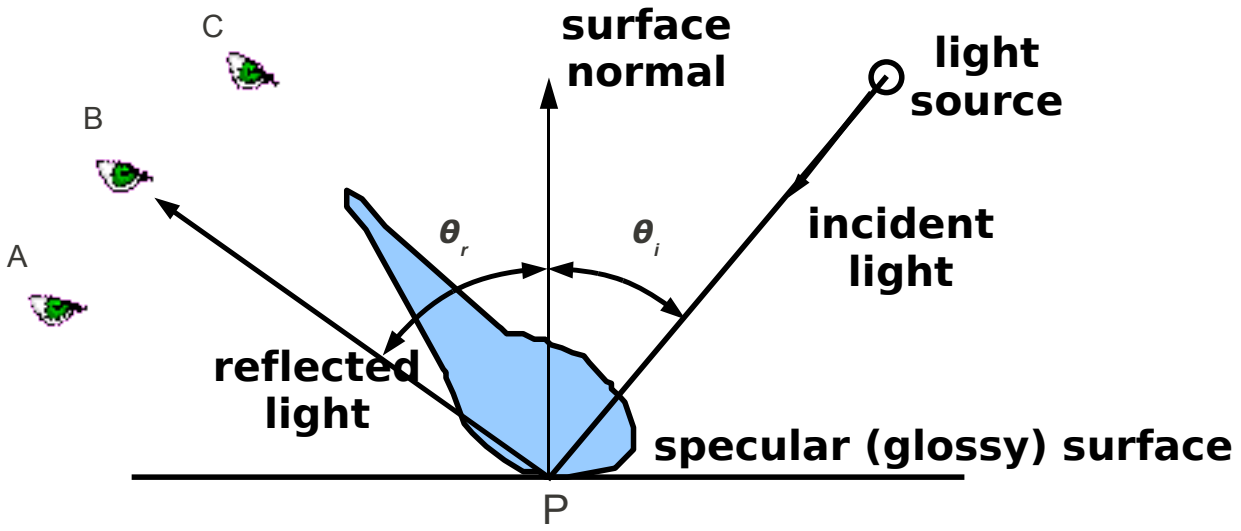
Amount of light reflected is independent of incident and reflected directions.



This, too, is bogus physically, but is often necessary. Why? Take a look around the room you're sitting in.

A Specular Shader

For a specular surface, the amount of light reflected depends on *both* the incident direction and the reflected direction.



Mathematically (but not in OpenGL), in the extreme case (an vanishingly narrow reflection peak of constant volume), you get a mirror (L. speculum means "mirror").

The Phong Illumination Model

It's a "power law". No physics involved, but based on a rough observational model. The original model for reflected light L_r is:

$$L_r = k_s \left(\widehat{\mathbf{R}}_{\widehat{\mathbf{V}}} \cdot \widehat{\mathbf{S}} \right)^{n_p} L_i$$

where k_s is the *specular reflectivity* and L_i is the incident light (coming from direction $\widehat{\mathbf{S}}$). but these days, the "Phong-Blinn" model is more common:

$$L_r = k_s \left(\widehat{\mathbf{N}} \cdot \widehat{\mathbf{H}} \right)^{n_b} L_i$$

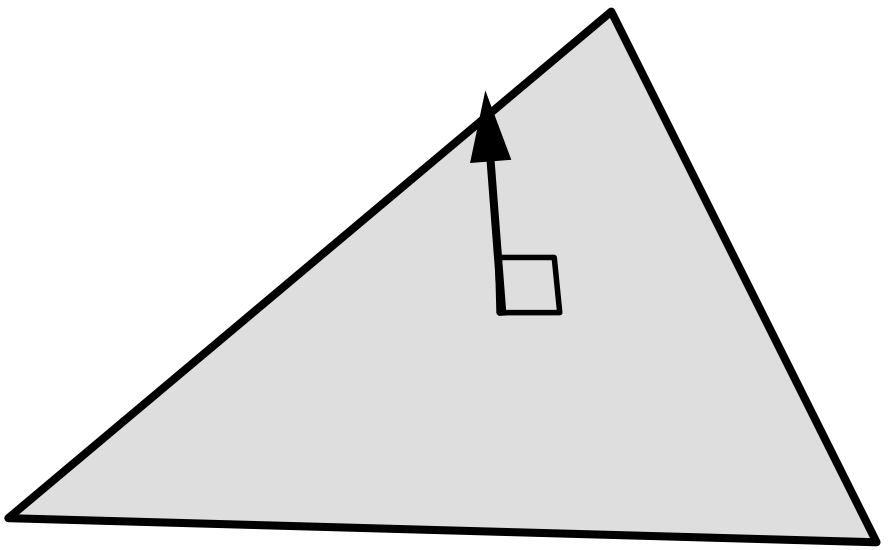
where $\vec{\mathbf{H}} \equiv \widehat{\mathbf{S}} + \widehat{\mathbf{V}}$ is the "halfway" vector between $\widehat{\mathbf{S}}$ and $\widehat{\mathbf{V}}$. (Implement the last one in PA04, the $\widehat{\mathbf{S}}$ is `towardsLight` and $\widehat{\mathbf{V}}$ is `towardsCamera`.)

The Phong Illumination Model Comments

- Neither the exponents nor the specular reflectivities are comparable between models.
- If the dot product is ever negative, the result is taken to be zero. (Specular – like diffuse – lighting is never negative.)
- The dot product (in both cases) is a cosine, so it's < 1, hence increasing the exponent makes the highlight smaller and "sharper".
- Reflected light is usually calculated (independently) for R, G, and B channels.
- This is the main "shader" used for specular surfaces in OpenGL and elsewhere.
- It's supported in virtually all 3D hardware.

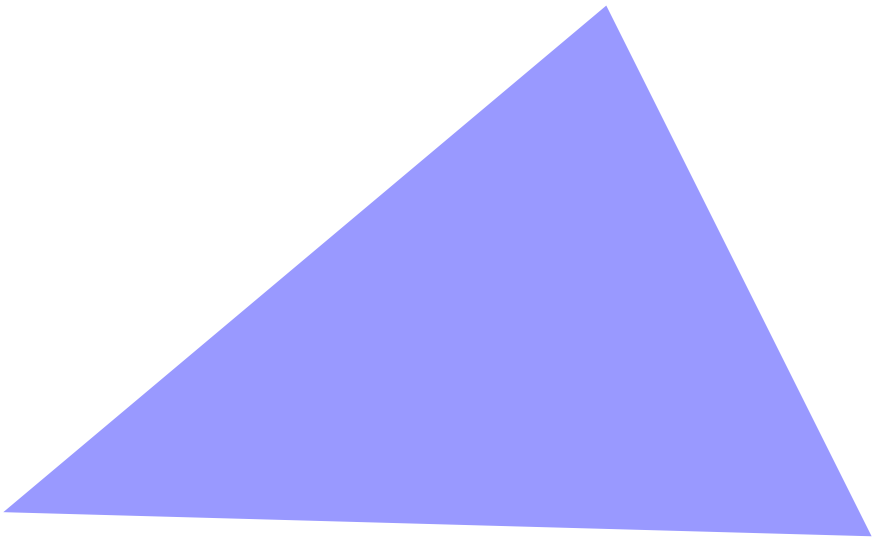
Flat Shading I

Given a triangle with its face normal:



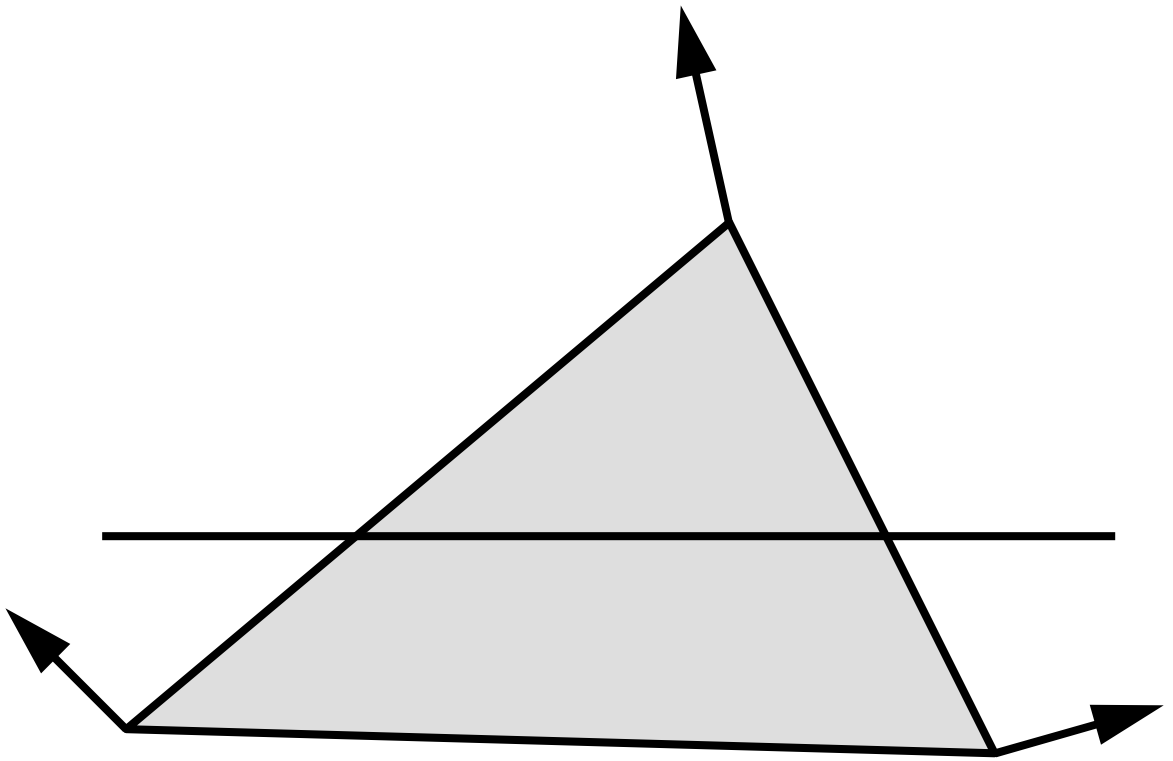
Flat Shading II

Shade the entire triangle based on the face normal:



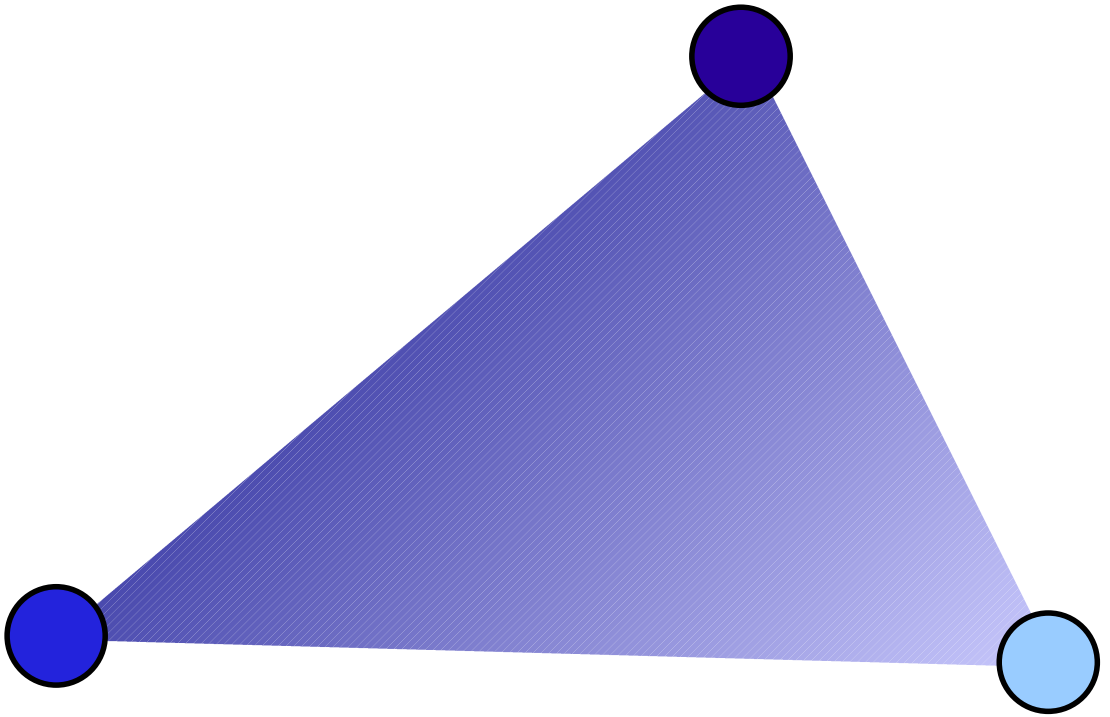
Gouraud Shading I

Given a triangle with three vertex normals:



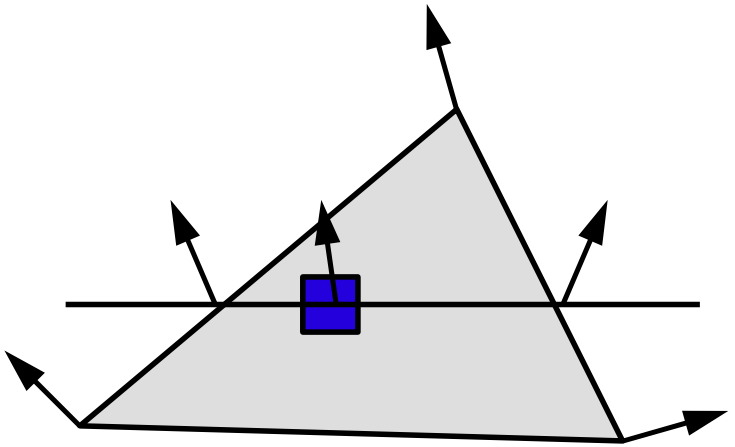
Gouraud Shading II

Compute vertex colors with the illumination model, then use smooth shading during fill.



Phong Shading

(not to be confused with the Phong illumination model – Phong was a clever guy!)



Given a triangle mesh,

- compute vertex normals
- interpolate edge normals
- interpolate normals along scan line
- use illumination model for *each* pixel