9. Shading

Reading

- Angel, sections 6.1 - 6.4

Introduction

So far, we've talked exclusively about geometry.

- What is the shape of an object?
- How do I place it in a virtual 3D space?
- How do I know which pixels it covers?
- How do I know which of the pixels I should actually draw?

Once we've answered all those, we have to ask one more important question:

- What value do I set each pixel to?

Answering this question is the job of the **shading model**.

(Of course, people also call it a lighting model, a light reflection model, a local illumination model, a reflectance model, etc., etc.)

Tedious reality

Properly determining the right color is *really hard.*

Look around the room. Each light source has different characteristics. Trillions of photons are pouring out every second.

These photons can:

- interact with the atmosphere, or with things in the atmosphere
- strike a surface and
    - be absorbed
    - be reflected
    - cause fluorescence or phosphorescence
- of course, none of the surfaces in here are perfect spheres or cylinders. At some microscopic level (very important for photons) they're all really bumpy.
- also, everything depends on wavelength.

Our problem

We're going to build up to an *approximation* of reality called the **Phong illumination model**.

It has the following characteristics:

- *not* physically based
- gives a first-order *approximation* to physical light reflection
- very fast
- widely used

Iteration zero

Given:

- a point P on a surface
- visible through pixel *p*

Assign each polygon a single color:

$$I = k_i$$

where

- I is the resulting intensity
- k_i is the intrinsic shade associated with the object

This has some special-purpose uses, but not really good for drawing a scene.

Iteration one

Let's make the color at least dependent on the overall quantity of light available in the scene:

$$I = k_a I_a$$

- k_a is the **ambient reflection coefficient.**
  - really the reflectance of ambient light
  - "ambient" light is assumed to be equal in all directions
- I_a is the **ambient intensity**.

Physically, what is "ambient" light?

Wavelength dependence

Really, k_a and I_a are functions over all wavelengths λ.

Ideally, we would do the calculation on these functions:

$$I(\lambda) = k_a(\lambda) I_a(\lambda)$$

then we would find good RGB values to represent the spectrum $I_a(\lambda)$.

Traditionally, though, k_a and I_a are represented as RGB triples, and the computation is performed on each color channel separately.

Diffuse reflection

Let's examine the ambient shading model:

- objects have different colors
- we can control the overall light intensity
  - what happens when we turn off the lights?
  - what happens as the light intensity increases?
  - what happens if we change the color of the lights?

So far, objects are uniformly lit.

- not the way things really appear
- in reality, light sources are directional

Diffuse, or **Lambertian** reflection will allow reflected intensity to vary with the direction of the light.

Diffuse reflectors

Diffuse reflection occurs from dull, matte surfaces, like latex paint, or chalk.

These **diffuse** or **Lambertian** reflectors reradiate light equally in all directions.

Picture a rough surface with lots of tiny **microfacets**.

The microfacets have the effect of distributing light rays in all directions.

Note:

- Light may actually penetrate the surface, bounce around, and then reflect back out.
- Accounts for colorization of diffusely reflected light by plastics.

Diffuse reflectors, cont.

The reflected intensity from a diffuse surface does not depend on the direction of the viewer. The incoming light, though, does depend on the direction of the light source.

Q: Why is the North Pole cold? Why is winter cold?

Iteration two

The incoming energy is proportional to cos θ, giving the diffuse reflection equations:

$$\begin{aligned} I &= k_a I_a + k_d I_l \cos\theta \\ &= k_a I_a + k_d I_l (\mathbf{N}\cdot\mathbf{L})_+ \end{aligned}$$

where:

- k_d is the **diffuse reflection coefficient**
- I_l is the intensity of the light source
- $\boldsymbol{N}$ is the normal to the surface (unit vector)
- $\boldsymbol{L}$ is the direction to the light source (unit vector)
- $(x)_+$ means max $\{0,x\}$

OpenGL supports different kinds of lights: point, directional, and spot. How do these work?

Intensity drop-off with distance

The laws of physics state that the intensity of a point light source must drop off with its distance squared.

We can incorporate this effect by multiplying I_l by $1/d^2$.

Sometimes, this distance-squared dropoff is considered too "harsh." Angel suggests using

$$\frac{1}{a+bd+cd^2}$$

with user-supplied constants for *a*, *b*, and *c*.

Specular reflection

Specular reflection accounts for the highlight that you see on some objects.

It is particularly important for *smooth, shiny* surfaces, such as:

- metal
- polished stone
- plastics
- Safeway apples

Specular reflection depends on the viewing direction ***V***. The color is often determined solely by the color of the light.

- corresponds to absence of internal reflections

Specular reflection derivation

For a perfect mirror reflector, light is reflected about ***N***, so

$$I = \begin{cases} I_l & \text{if } \mathbf{V} = \mathbf{R} \\ 0 & \text{otherwise} \end{cases}$$

For a near-perfect reflector, you might expect the highlight to fall off quickly with increasing angle ϕ.

Also known as:

- **"rough specular" reflection**
- **"directional diffuse" reflection**
- **"glossy" reflection**

Derivation, cont.

One way to get this effect is to take $(\boldsymbol{R}\cdot\boldsymbol{V})$, raised to a power n_s.

As *ns* gets larger,

- the dropoff becomes {more,less} gradual
- gives a {larger,smaller} highlight
- simulates a {more,less} glossy surface

Iteration three

Since light is additive, we can handle multiple lights by taking the sum over every light.

Our equation is now:

$$I = k_a I_a + \sum_i f(d_i) I_{li} \left[k_d (\mathbf{N} \cdot \mathbf{L}_i)_+ + k_s (\mathbf{V} \cdot \mathbf{R})_+^{n_s} \right]$$

This is the Phong illumination model.

Which quantities are spatial vectors?

Which are RGB triples?

Which are scalars?

Choosing the parameters

How would I model…

- polished copper?
- blue plastic?
- lunar dust?

Summary

The most important thing to take away from this lecture is the final equation for the Phong model.

- What is the physical meaning of each variable?
- How are the terms computed?
- What effect does each term contribute to the image?
- What does varying the parameters do?