Basic 3D graphics

With affine matrices, we can now transform virtual 3D objects in their local coordinate systems into a global (world) coordinate system:

To synthesize an image of the scene, we also need to add light sources and a viewer/camera:

Pinhole camera

To create an image of a virtual scene, we need to define a camera, and we need to model lighting and shading. For the camera, we use a pinhole camera.

The image is rendered onto an image plane (usually in front of the camera).

Viewing rays emanate from the center of projection (COP) at the center of the pinhole.

The image of an object point $P$ is at the intersection of the viewing ray through $P$ and the image plane.

But is $P$ visible? This the problem of hidden surface removal (a.k.a., visible surface determination). We'll consider this problem later.
Shading

Next, we'll need a model to describe how light interacts with surfaces. Such a model is called a shading model.

Other names:
- Lighting model
- Light reflection model
- Local illumination model
- Reflectance model
- BRDF

An abundance of photons

Given the camera and shading model, properly determining the right color at each pixel is extremely hard.

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

These photons can:
- interact with molecules and particles in the air ("participating media")
- strike a surface and
  - be absorbed
  - be reflected (scattered)
  - cause fluorescence or phosphorescence.
- interact in a wavelength-dependent manner
- generally bounce around and around

Our problem

We're going to build up to a approximations of reality called the Phong and Blinn-Phong illumination models.

They have the following characteristics:
- not physically correct
- gives a “first-order” approximation to physical light reflection
- very fast
- widely used

In addition, we will assume local illumination, i.e., light goes: light source -> surface -> viewer.

No interreflections, no shadows.

Setup…

Given:
- a point $P$ on a surface visible through pixel $p$
- The normal $N$ at $P$
- The lighting direction, $L$, and (color) intensity, $I_L$, at $P$
- The viewing direction, $V$, at $P$
- The shading coefficients at $P$

Compute the color, $I$, of pixel $p$.

Assume that the direction vectors are normalized:

$$\|N\| = \|L\| = \|V\| = 1$$
"Iteration zero"

The simplest thing you can do is...

Assign each polygon a single color:

\[ I = k_e \]

where

- \( I \) is the resulting intensity
- \( k_e \) is the emissivity or intrinsic shade associated with the object

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

[Note: \( k_e \) is omitted in Shirley.]

"Iteration one"

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

\[ I = k_e + k_a I_a \]

- \( k_e \) 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 light intensity.

Physically, what is “ambient” light?

[Note: Shirley uses \( c_a \) instead of \( I_a \).]

Wavelength dependence

Really, \( k_a \) and \( I_a \) are functions over all wavelengths \( \lambda \).

Ideally, we would do the calculation on these functions. For the ambient shading equation, we would start with:

\[ I(\lambda) = k_a(\lambda) I_a(\lambda) \]

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

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

\[ r^a = k_a^r I_a^r \]
\[ g^a = k_a^g I_a^g \]
\[ b^a = k_a^b I_a^b \]

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.
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 localized in position or direction

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

Diffuse reflectors

...or picture a surface with little pigment particles embedded beneath the surface (neglect reflection at the surface for the moment):

The microfacets and pigments distribute light rays in all directions.

Embedded pigments are responsible for the coloration of diffusely reflected light in plastics and paints.

Note: the figures above are intuitive, but not strictly (physically) correct.

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:

“Iteration two”

The incoming energy is proportional to $\gamma$, giving the diffuse reflection equations:

$$I = k_d J_{wa} + k_d L B$$

$$= k_d J_{wa} + k_d L (\begin{cases} 1 & \text{if } \mathbf{N} \cdot \mathbf{L} > 0 \\ 0 & \text{if } \mathbf{N} \cdot \mathbf{L} \leq 0 \end{cases})$$

where:

- $k_d$ is the diffuse reflection coefficient
- $J_{wa}$ is the (color) intensity of the light source
- $\mathbf{N}$ is the normal to the surface (unit vector)
- $\mathbf{L}$ is the direction to the light source (unit vector)
- $B$ prevents contribution of light from below the surface:

$$B = \begin{cases} 1 & \text{if } \mathbf{N} \cdot \mathbf{L} > 0 \\ 0 & \text{if } \mathbf{N} \cdot \mathbf{L} \leq 0 \end{cases}$$

[Note: Shirley uses $c_r$ and $c_l$ instead of $k_d$ and $L$.]
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
- apples
- skin

Properties:
- Specular reflection depends on the viewing direction V.
- For non-metals, the color is determined solely by the color of the light.
- For metals, the color may be altered (e.g., brass)

Phong specular reflection

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

As $n_s$ gets larger,
- the dropoff becomes (more,less) gradual
- gives a (larger,smaller) highlight
- simulates a (more,less) mirror-like surface

Phong specular reflection is proportional to:
\[ I_{\text{specular}} \sim B(R \cdot V)^{n_s} \]
where $(x)_+ = \max(0, x)$.

Blinn-Phong specular reflection

A common alternative for specular reflection is the Blinn-Phong model (sometimes called the modified Phong model).

We compute the vector halfway between L and V as:

Analogous to Phong specular reflection, we can compute the specular contribution in terms of $(N \cdot H)$, raised to a power $n_s$:
\[ I_{\text{specular}} = B(N \cdot H)^{n_s} \]
where, again, $(x)_+ = \max(0, x)$. 
"Iteration three"

The next update to the Blinn-Phong shading model is then:

\[ I = k_s + k_rI_s + I_sB(N \cdot L) + k_dB(N \cdot H)^n_s \]

where:
- \( k_s \) is the **specular reflection coefficient**
- \( n_s \) is the **specular exponent** or shininess
- \( H \) is the unit halfway vector between \( L \) and \( V \), where \( V \) is the viewing direction, and:

\[(N \cdot H)^n_s = \begin{cases} (N \cdot H)^n_s & \text{if } N \cdot H > 0 \\ 0 & \text{if } N \cdot H \leq 0 \end{cases} \]

[Note: Shirley uses \( e, r, h, \) and \( p \) instead of \( V, R, H, \) and \( n_s \).]

**Directional lights**

The simplest form of lights supported by renderers are ambient, directional, and point. Spotlights are also supported often as a special form of point light.

We’ve seen ambient light sources, which are not really geometric.

**Directional light** sources have a single direction and intensity associated with them.

**Point lights**

The direction of a **point light** sources is determined by the vector from the light position to the surface point.

Physics tells us the intensity must drop off inversely with the square of the distance:

\[ f_{\text{atten}} = \frac{1}{r^2} \]

Sometimes, this distance-squared dropoff is considered too "harsh." A common alternative is:

\[ f_{\text{atten}} = \frac{1}{a + br + cr^2} \]

with user-supplied constants for \( a, b, \) and \( c \).

Using affine notation, what is the homogeneous coordinate for a point light?

**Spotlights**

We can also apply a *directional attenuation* of a point light source, giving a **spotlight** effect.

A common choice for the spotlight intensity is:

\[ f_{\text{spot}} = \frac{(L \cdot S)^e}{a + br + cr^2} \quad \text{if } L \cdot S \leq \cos \beta \]

\[ 0 \quad \text{otherwise} \]

where:
- \( L \) is the direction to the point light.
- \( S \) is the center direction of the spotlight.
- \( \beta \) is the cutoff angle for the spotlight
- \( e \) is the angular falloff coefficient
“Iteration four”

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

Our equation is now:

\[ I = k_a + k_d \sum_{j} \left[ \frac{(L_j \cdot S_j)^{\beta_j}}{a_j + b_j \cdot \gamma_j + c_j \cdot \gamma_j^2} \right] \\
= I_k + k_a + \sum_{j} \left[ k_s (N \cdot L_j) + k_s (N \cdot H_j)^{\gamma_j} \right] \\
\]

This is the Blinn-Phong illumination model (for spotlights).

Which quantities are spatial vectors?

Which are RGB triples?

Which are scalars?

Choosing the parameters

Experiment with different parameter settings. To get you started, here are a few suggestions:

- Try \( n_s \) in the range \([0, 100]\)
- Try \( k_d + k_s + k_a < 1 \)
- Use a small \( k_a \) (~0.1)

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

More sophisticated BRDF’s

Anisotropic BRDFs [Westin, Arvo, Torrance 1992]

Artistics BRDFs [Gooch]
More sophisticated BRDF's (cont'd)

Hair illuminated from different angles [Marschner et al., 2003]

Wool and silk cloths [Irawan and Marschner, 2012]

BSSRDFs for subsurface scattering

[Jensen et al. 2001]

Shading the interiors of triangles

We will be computing colors using the Blinn-Phong lighting model.

Let's assume (as graphics hardware does) that we are working with triangles.

How should we shade the interiors of triangles?

Shading with per-face normals

Assume each face has a constant normal:

For a distant viewer and a distant light source and constant material properties over the surface, how will the color of each triangle vary?
**Phong interpolation**

To get smoother results, we can "hallucinate" normals that vary smoothly within a triangle, a process called *Phong interpolation*.

Here's how it works:

1. Compute normals at the vertices.
2. Interpolate normals and normalize.
3. Shade using the interpolated normals.