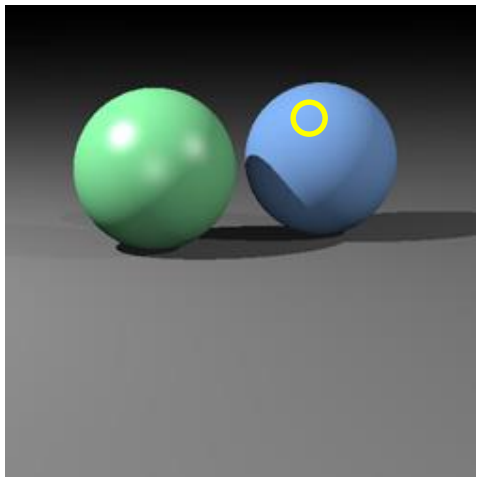
Lights and Shading

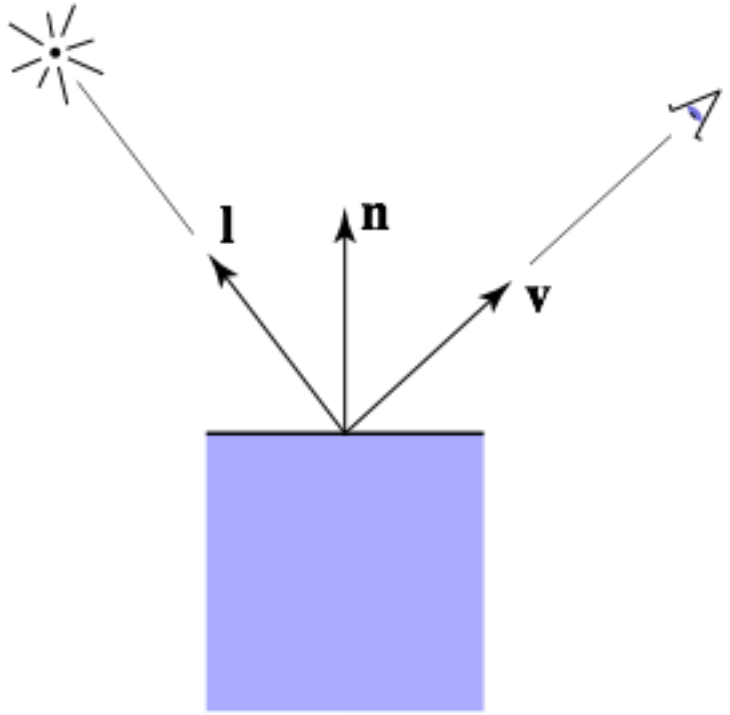
COMP557
Paul Kry



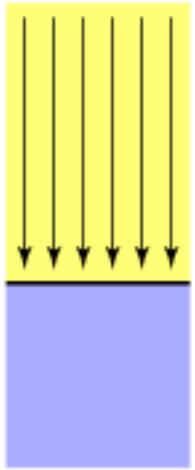
light source
viewer (eye)
Illumination
Projection
direction
objects
in scene

Shading

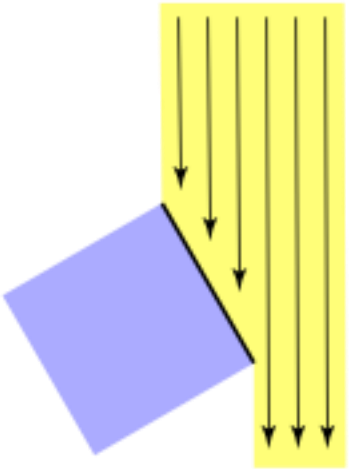
- Compute light reflected (*scattered*) toward camera
- Inputs:
  - eye direction
  - light direction
    (for each of many lights)
  - surface normal
  - surface parameters
    (color, shininess, …)



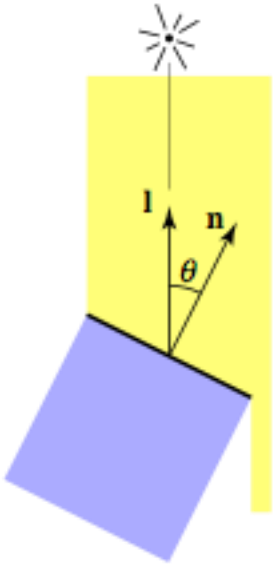
Diffuse reflection



Top face of cube receives a certain amount of light



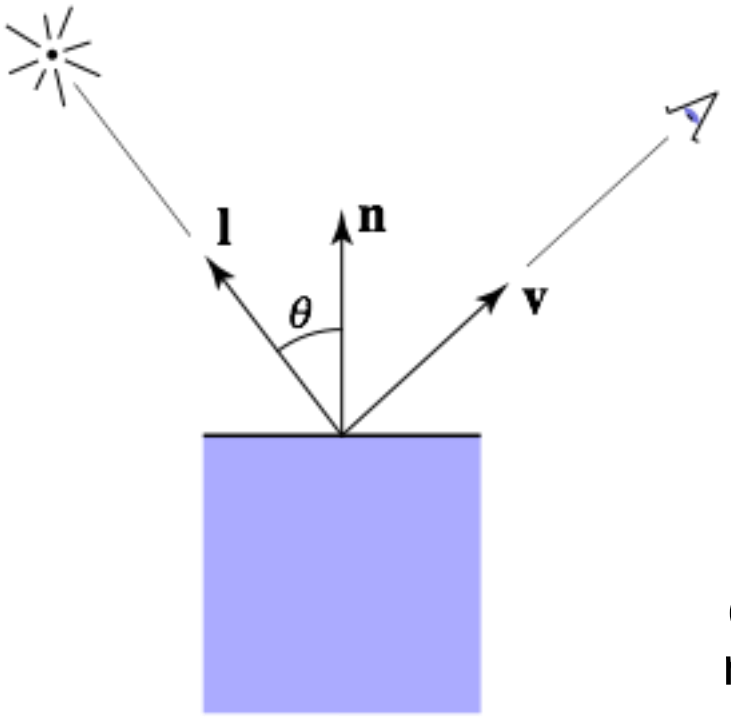
Top face of 60º rotated cube intercepts half the light



In general, light per unit area is proportional to $\cos\theta = \mathbf{l} \bullet \mathbf{n}$

Lambertian shading

A Lambertian surface scatters light equally in all directions



illumination from source

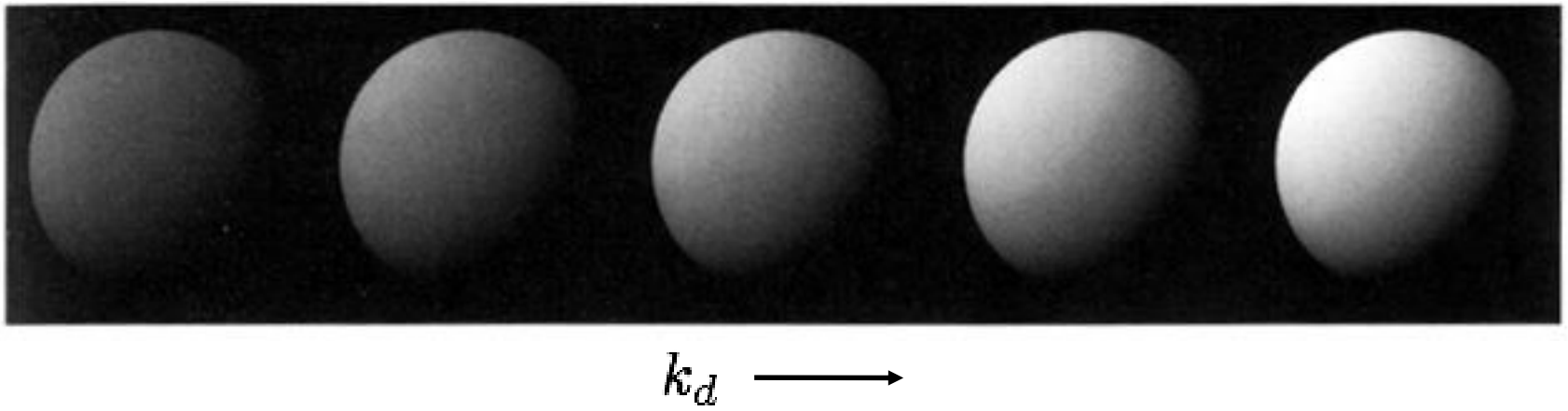
$$L_d = k_d \, I \max(0, \mathbf{n} \cdot \mathbf{l})$$

diffuse coefficient

diffusely reflected light

Lambertian shading

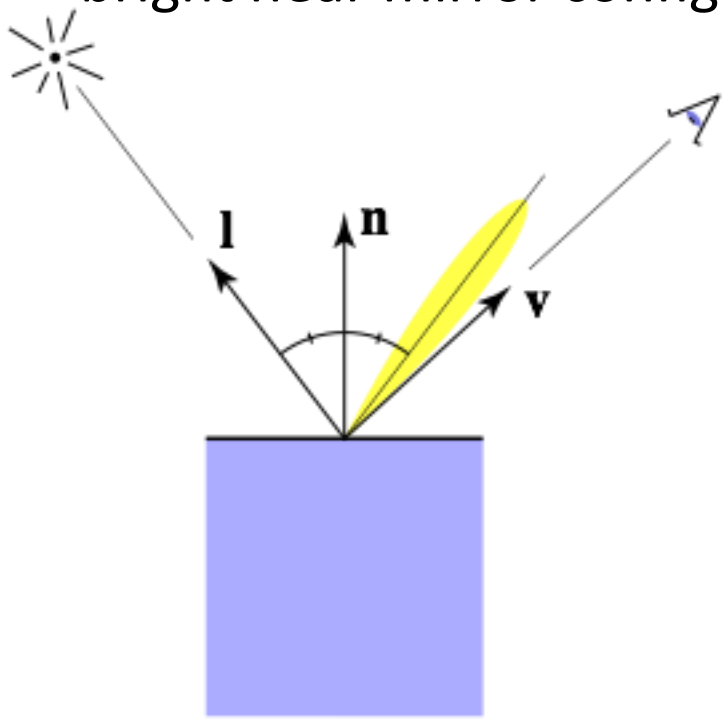
- Produces matte appearance



[Foley et al.]

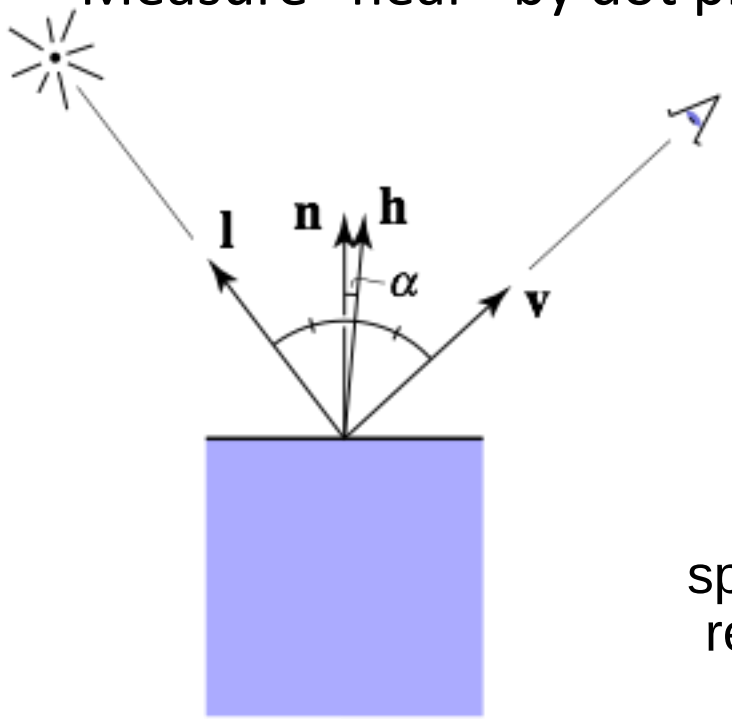
Specular shading (Blinn-Phong)

- Intensity depends on view direction
  - bright near mirror configuration



Specular shading (Blinn-Phong)

- Close to mirror ⇔ half vector near normal
  - Measure “near” by dot product of unit vectors



$$\mathbf{h} = \text{bisector}(\mathbf{v}, \mathbf{l}) = \frac{\mathbf{v}+\mathbf{l}}{\|\mathbf{v}+\mathbf{l}\|}$$

$$L_s = k_s\, I \max(0, \cos\alpha)^p = k_s\, I \max(0, \mathbf{n}\cdot\mathbf{h})^p$$

specularly reflected light

specular coefficient

Phong model—plots

- Increasing specular exponent *p* narrows the lobe

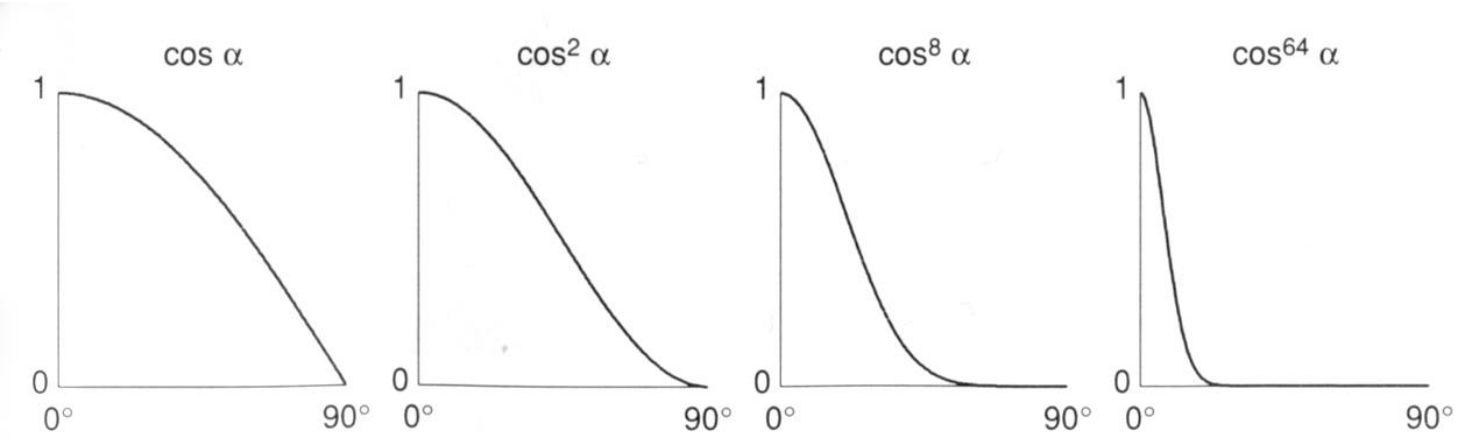
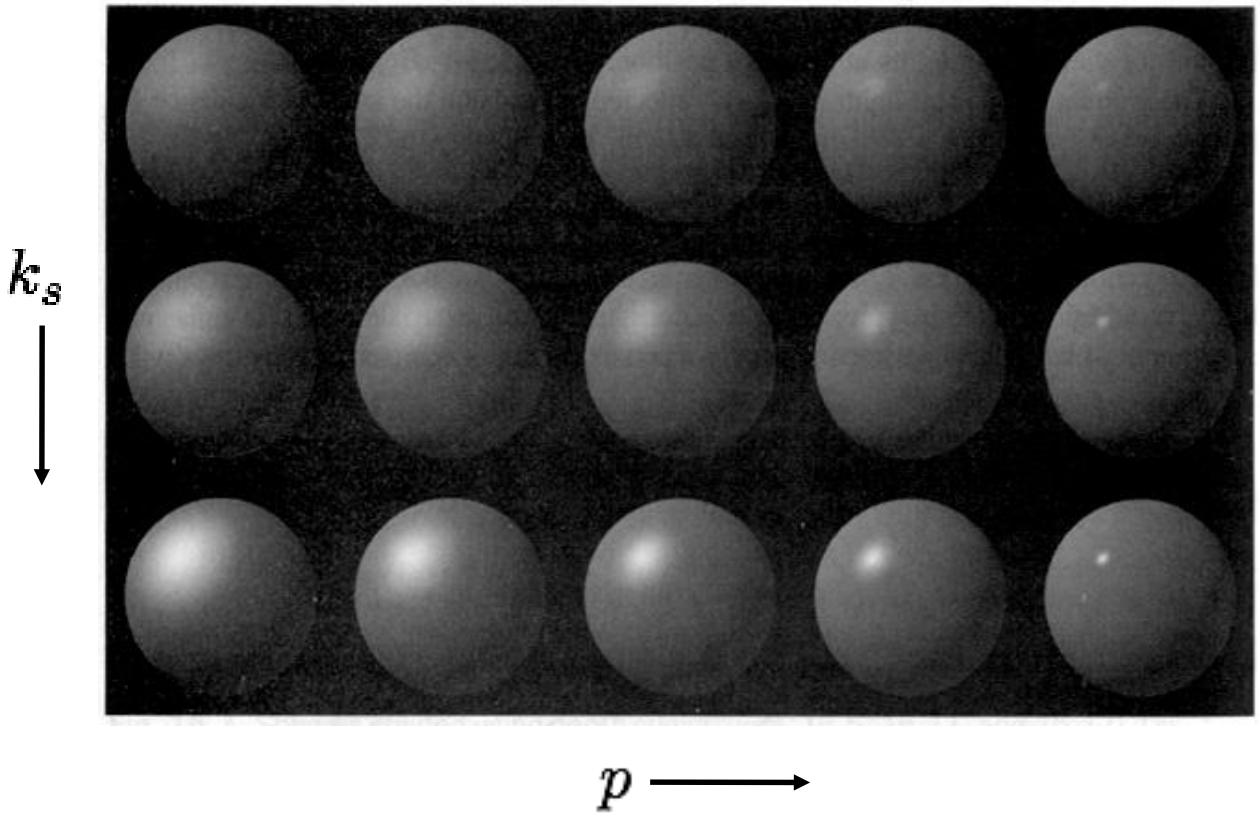


Fig. 16.9 Different values of $\cos^n \alpha$ used in the Phong illumination model.

- What are good values for the exponent?
- Alternatively what are bad values? What can go wrong?

[Foley et al.]

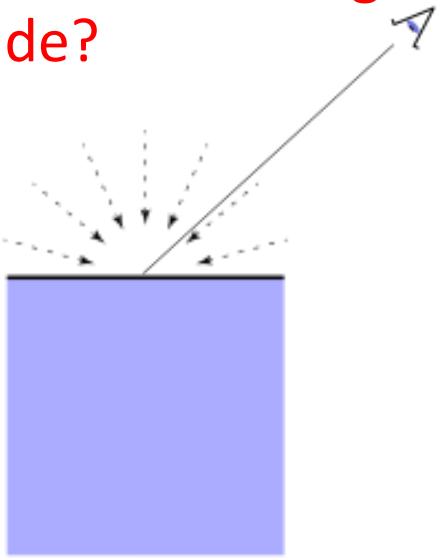
Specular shading



[Foley et al.]

Ambient shading

- Shading that does not depend on anything
  - add constant color to account for ***disregarded illumination*** and fill in black shadows
  - What is this "disregarded illumination" that we didn't include?



$$L_a = k_a I_a$$

ambient coefficient

reflected ambient light

Putting it together

- Usually include ambient, diffuse, Phong in one model

$$L = L_a + L_d + L_s$$
$$= k_a\, I_a + k_d\, I \max(0, \mathbf{n} \cdot \mathbf{l}) + k_s\, I \max(0, \mathbf{n} \cdot \mathbf{h})^p$$

- The final result is the sum over many lights

$$L = L_a + \sum_{i=1}^{N} [(L_d)_i + (L_s)_i]$$

$$L = k_a\, I_a + \sum_{i=1}^{N} [k_d\, I_i \max(0, \mathbf{n} \cdot \mathbf{l}_i) + k_s\, I_i \max(0, \mathbf{n} \cdot \mathbf{h}_i)^p]$$

OpenGL Lighting Functions

chapter 5 of red book

What are all these zeros?

```
gl.glEnable( GL2.GL_LIGHTING );
gl.glEnable( GL2.GL_LIGHT0 );
```

Need to enable lighting, otherwise OpenGL does not perform lighting computations. You also need to "turn on" each light you want to use (typically 8 available)

```
gl.glLightfv( GL2.GL_LIGHT0, GL2.GL_POSITION, position, 0 );
```

Position is a float[] with 4 components, homogeneous coordinates. Directional lights are set by providing a vector instead of a position.

USE SAME

```
gl.glLightfv( GL2.GL_LIGHT0, GL2.GL_SPECULAR, Is, 0 );
gl.glLightfv( GL2.GL_LIGHT0, GL2.GL_DIFFUSE, Id, 0 );
gl.glLightfv( GL2.GL_LIGHT0, GL2.GL_AMBIENT, Ia, 0 );
```

Typically we provide the same colour for specular and diffuse. The 4th component of the float[] should be 1 (no transparency). Each light contributes an ambient amount (use zero or a *small* value of the same colour).

SUMMED

```
gl.glLightModelfv( GL2.GL_LIGHT_MODEL_AMBIENT, Ia, 0);
```

There is also an overall ambient amount (non zero by default). glLightModelfv is also used to set GL_LIGHT_MODEL_TWO_SIDE (off by default), which is useful for drawing surfaces with different front and back materials (the front surface has the normal pointing towards you).

Lights also have spot settings (direction, exponent, cutoff, disabled by default), and also an attenuation model settings…

OpenGL Material Functions

chapter 5 of red book

gl.glMaterialfv(GL.GL_FRONT, GL2.GL_DIFFUSE, kd, 0);

gl.glMaterialfv(GL.GL_FRONT, GL2.GL_AMBIENT, ka, 0);

We often want the diffuse and ambient reflectance properties to be the same, thus we can pass the parameter GL2.GL_AMBIENT_AND_DIFFUSE. If using two sided lighting, we may also want to quickly set material properties of both sides with GL.GL_FRONT_AND_BACK

gl.glMaterialfv(GL.GL_FRONT_AND_BACK, GL2.GL_SPECULAR, ks, 0);

Specular highlights commonly act like glossy reflections, and directly scatter the colour of the light, so a common value for ks is {1,1,1,1}. Otherwise, the light colour components will be component-wise modulated by what you provide here.

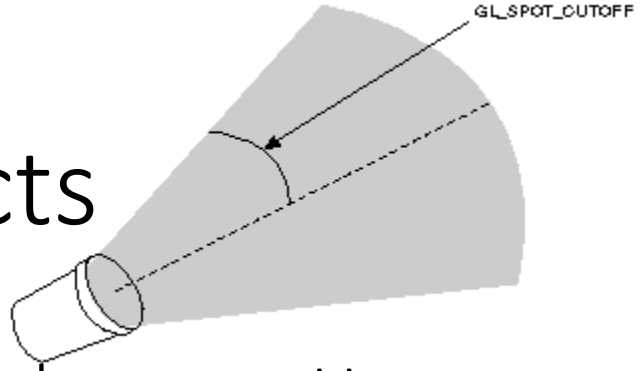
Can you think of a real world material that has coloured specular highlights?

gl.glMaterialf(GL.GL_FRONT_AND_BACK, GL2.GL_SHININESS, p);

The higher the number on the specular exponent p, the sharper the highlight. Note that 128 is typically the maximum value, while low values can look very strange!

Materials can also be emissive (glow in the dark), and this is set with the GL_EMISSION material parameter (which is {0,0,0,1} by default)

Other effects



- Spot lights
  - Direction, cutoff, and an exponent to control how concentrated.
- Attenuation
  - Intensity of light decreases as distance from the light increases.
  - Directional light is “infinitely far away”, so attenuation disabled.
  - Constant and Linear term helps us capture the attenuation behaviour of area lights.
    http://imdoingitwrong.wordpress.com/2011/01/31/light-attenuation/

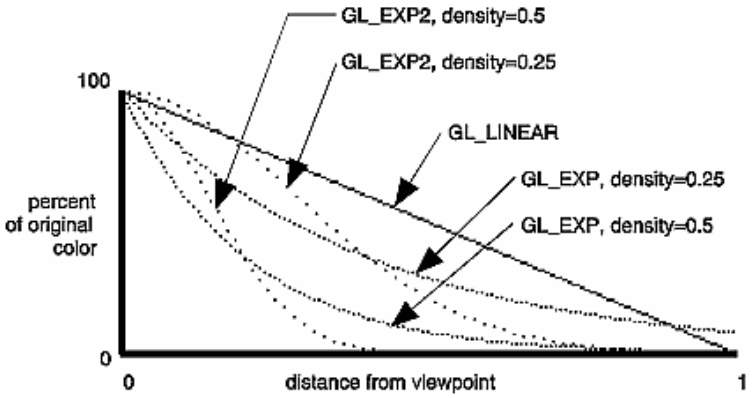
$$\frac{1}{k_c + k_l d + k_q d^2}$$

- Fog
  - Colour, exponential with density, or linear with start and end.

$$f = e^{-(density \cdot z)} \text{ (GL\_EXP)}$$

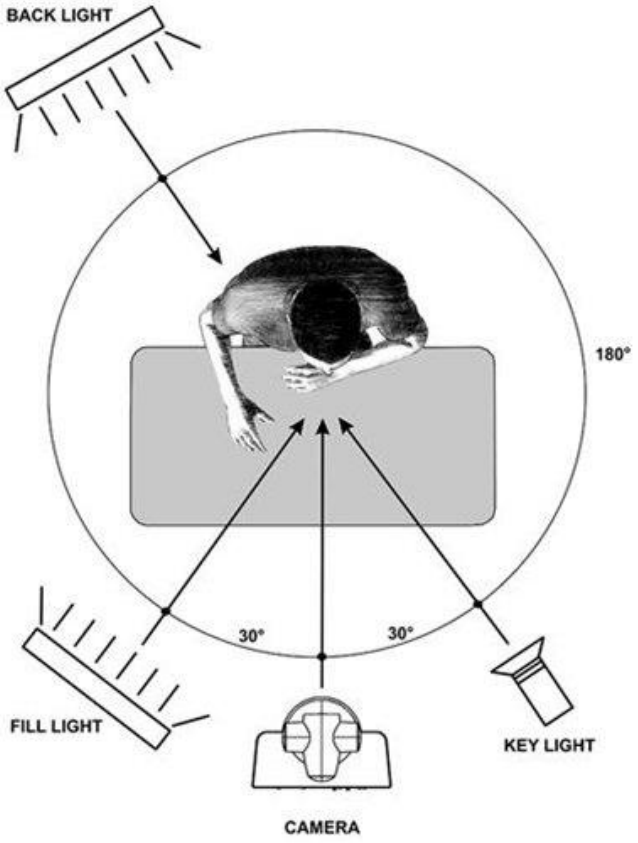
$$f = e^{-(density \cdot z)^2} \text{ (GL\_EXP2)}$$

$$f = \frac{end - z}{end - start} \text{ (GL\_LINEAR)}$$



Multiple lights

- Completely black shadows are not so pleasing to the eye, so add additional lights to fill in the shadows.
  - Three lights often using in television: key, fill, back
  - Fill and back can be “area” lights, harder to deal with in object order (and also image order).
    Why? Soft shadows!
- Loop over lights, add contributions
- Ambient shading
  - black shadows are not really right
  - one solution: dim light at camera
  - alternative: add a constant “ambient” color to the shading



Review and more information

- Fundamentals of Computer Graphics
  - Chapter 10, but not Section 10.3 on artistic shading
  - Ambient, Diffuse, Specular (Phong)
  - This is also covered in Chapter 4 Section 4.5 on the basic shading equations in the context of ray tracing
- OpenGL RedBook Chapter 5
  - Defining light parameters
  - Defining material parameters
  - Attenuation, fog, spots
  - Lighting models (more on this soon)