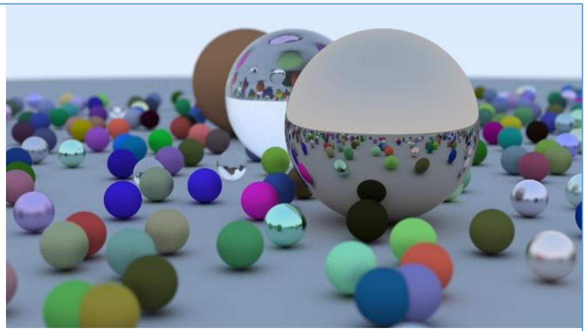
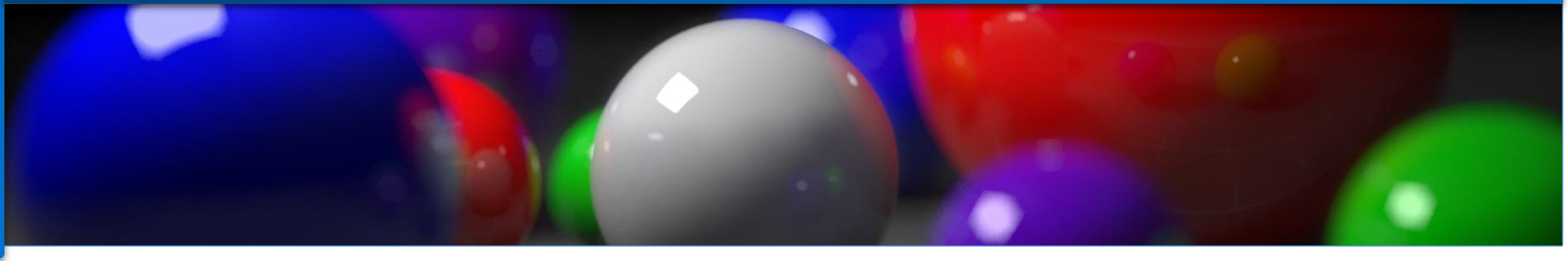
Comp4422



Computer Graphics

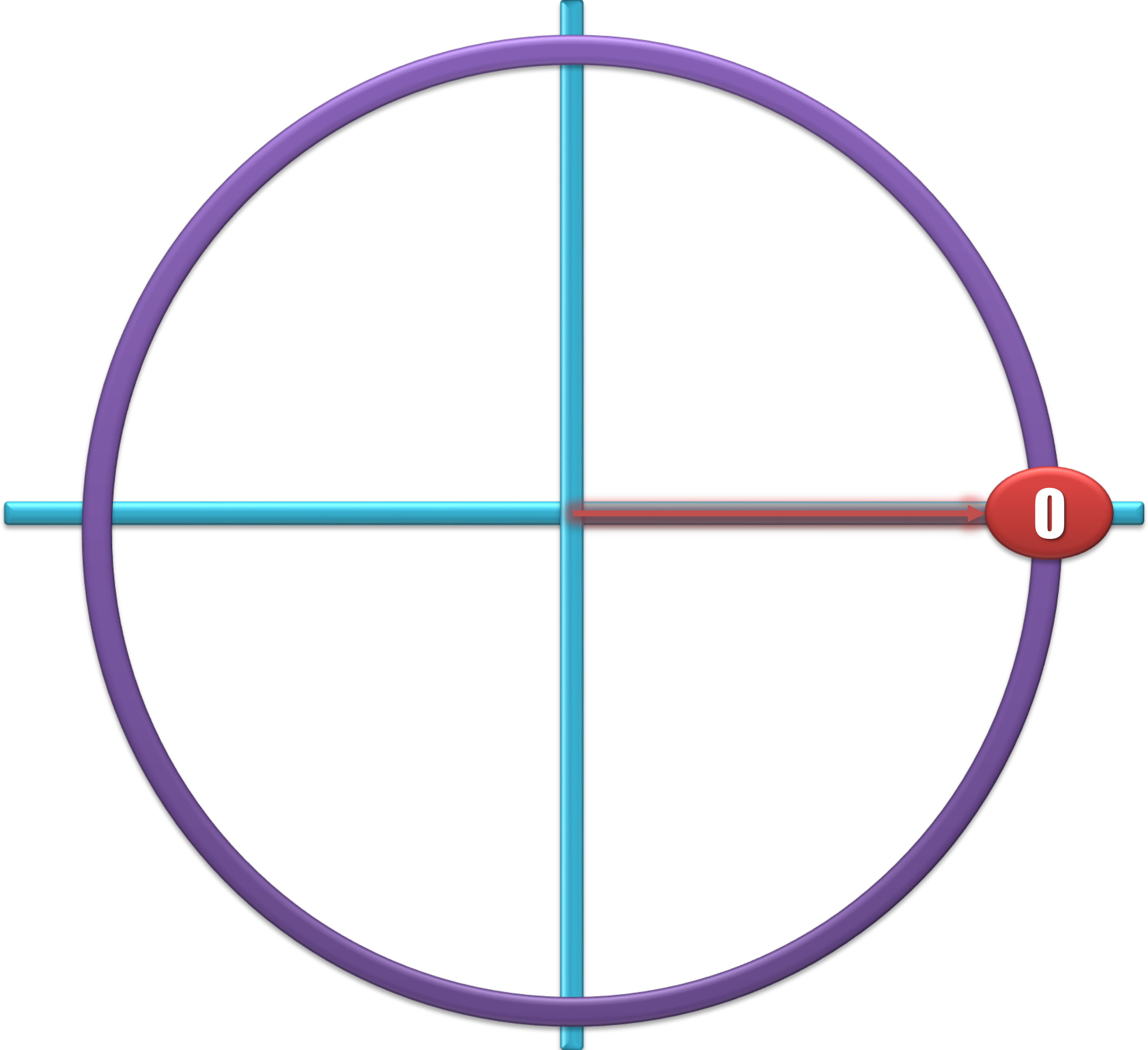
Lecture 07:

Light



What you will learn…

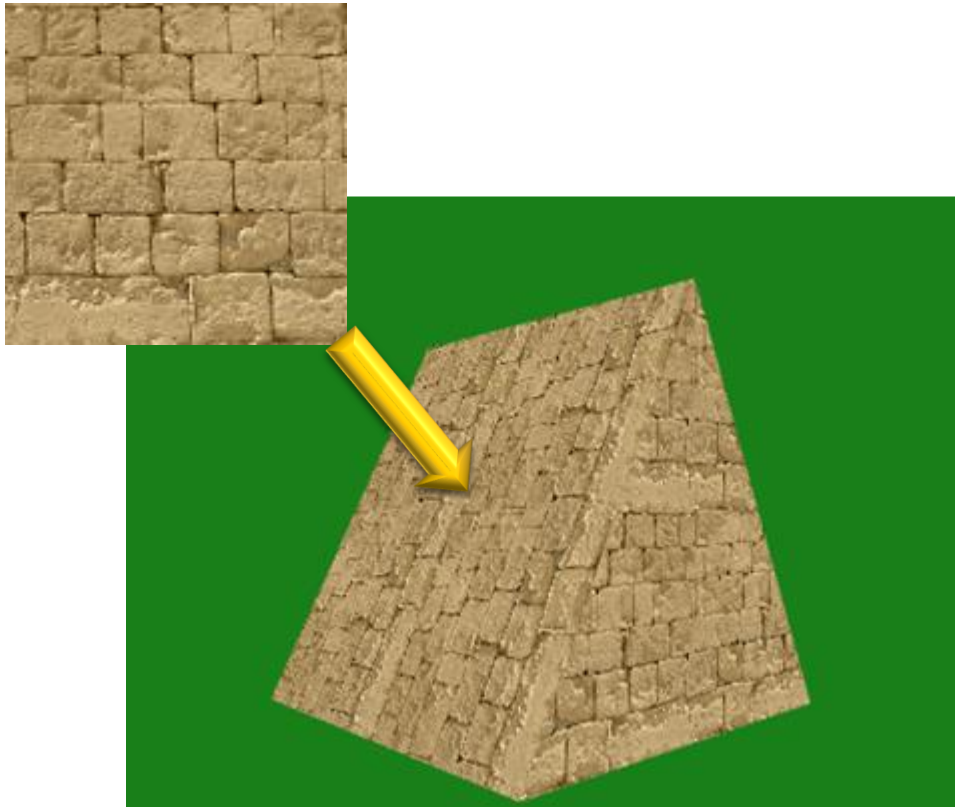
- Multiple textures
- Basic light modeling
- Basic Shading
- Lighting vs Shading Differences


0

Multiple Textures

- Single Texture

1



- Double Texture

1 2

Multiple Texture

- You can use multi-textures to produce special effects such as:
  - light or height maps
  - bumpmapping (simulating bumps and wrinkles).
- We will just take the WebGL logo as a texture and mix it with the stone texture.

Two Textures

- We assign constant values to the variables:
  - STONE_TEXTURE
  - WEBGL_LOGO
- We will use them as our new array indices.
- Then, we change our texture variable declarations to be arrays
- Adjust the LoadTexture and SetupTexture functions to handle multiple textures.

Preparing for MultiTexture

```
// Global settings:
STONE_TEXTURE                = 0,
WEBGL_LOGO                   = 1,
texture                      = [],
textureImage                 = [];
textureFileName[]            = [
          "./texture/stone-128px.jpg“,
          “./texture/webgl-logo-512px.png”
          ];
```

Texture i = 0..1

```
function wglLoadTexture(i)
{       textureImage[i] = new Image();
        textureImage[i].onload = function() {
                setupTexture(i);
                gl.uniform1i(glProgram.samplerUniform, i);
        }
        textureImage[i].src = textureFileName[i];
}
```

Set up the uniform

```
glProgram.uDoTexturing =
gl.getUniformLocation(glProgram, "uDoTexturing");
gl.uniform1i(glProgram.uDoTexturing, 1);
```

Set up texture buffers

```
function wglSetupTexture(i) {
        gl.activeTexture(gl.TEXTURE0 + i);        // must specify the active texture
        texture[i] = gl.createTexture();
        gl.bindTexture(gl.TEXTURE_2D, texture[i]);
        gl.pixelStorei(gl.UNPACK_FLIP_Y_WEBGL, true);
        gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, gl.RGBA,
                gl.UNSIGNED_BYTE, textureImage[i]);
        gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MAG_FILTER, gl.NEAREST);
        gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.NEAREST);
        if( !gl.isTexture(texture[i]) ) {
                console.error("Error: Texture is invalid"); }
}
```

Get Sampler Locations

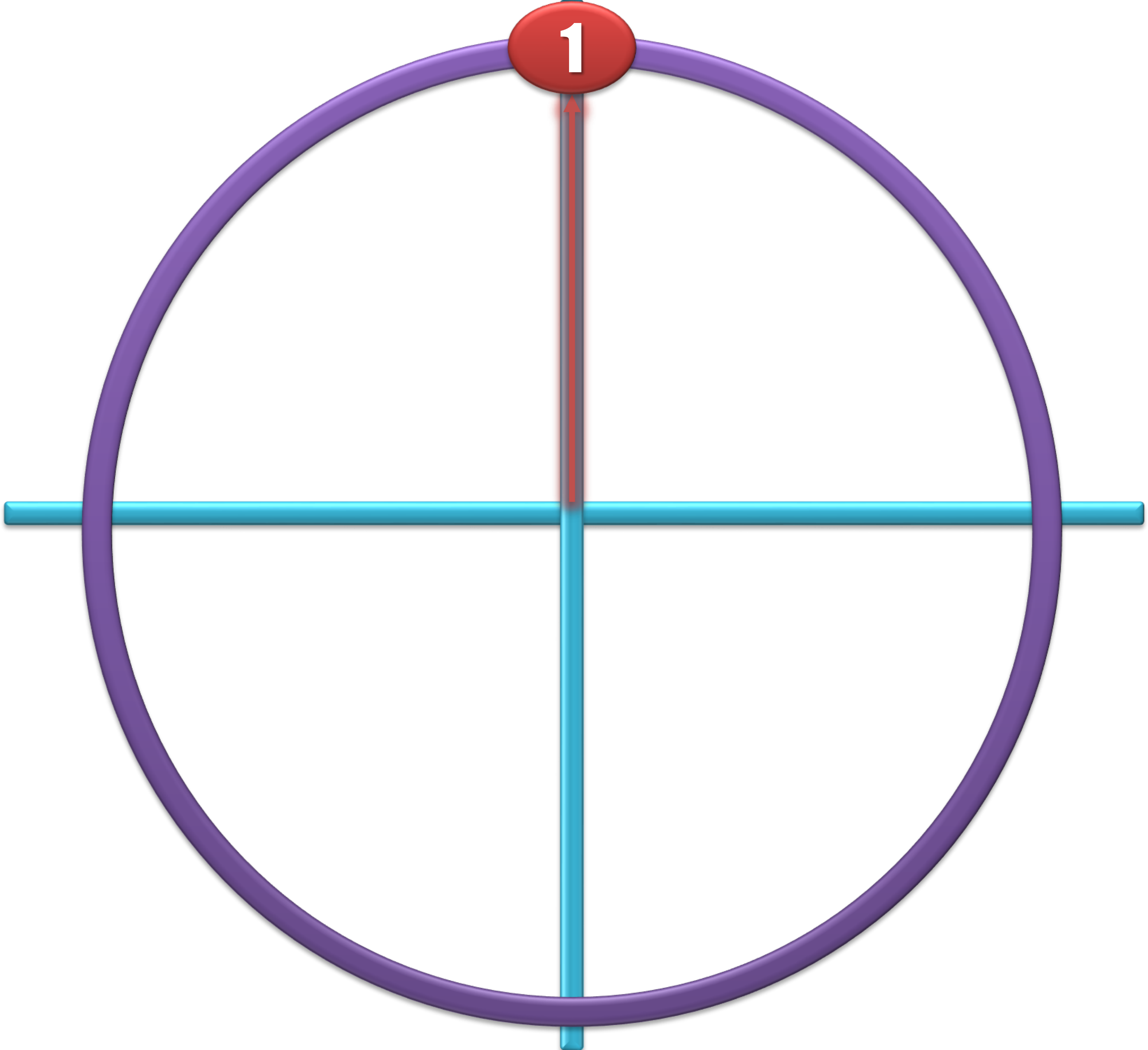
```
function getMatrixUniforms(){
	glProgram.pMatrixUniform = gl.getUniformLocation(glProgram, "uPMatrix");
	glProgram.mvMatrixUniform = gl.getUniformLocation(glProgram, "uMVMatrix");
	glProgram.samplerUniform = gl.getUniformLocation(glProgram, "uSampler");
	glProgram.samplerUniform2 = gl.getUniformLocation(glProgram, "uSampler2");
}
```

Finally, the Frag. Shader

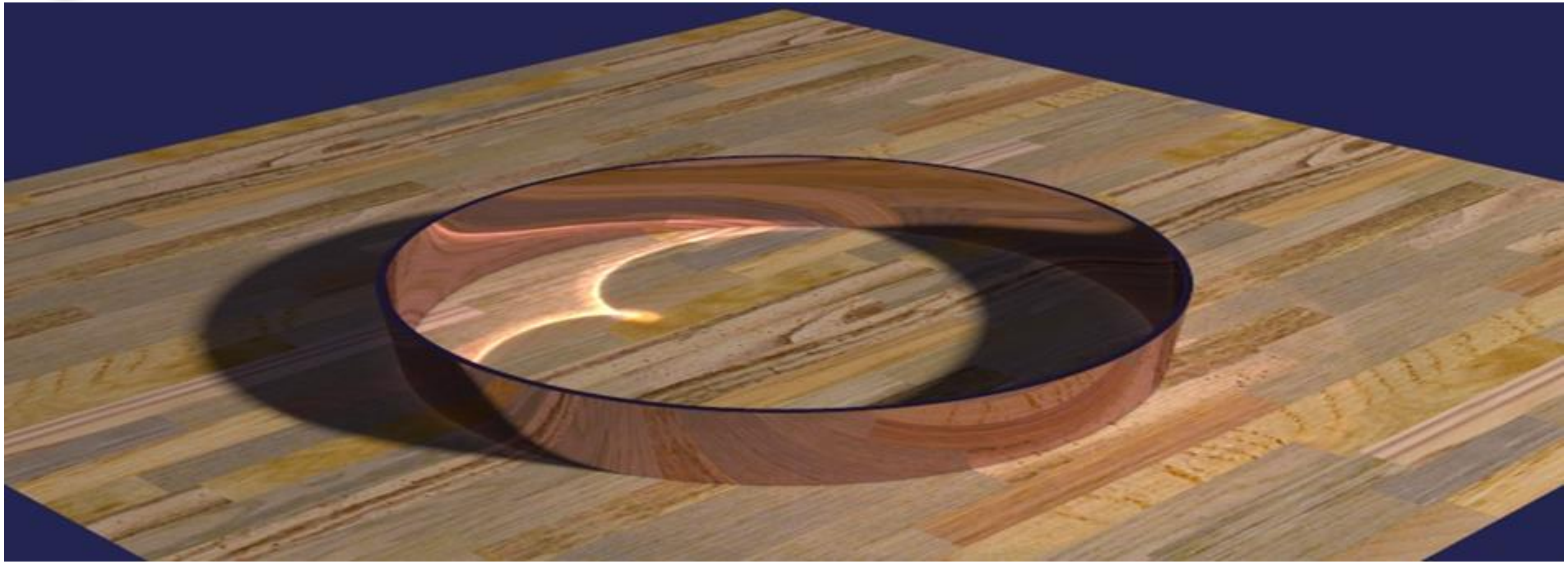
```
<script id="shader-fs" type="x-shader/x-fragment">
	varying highp vec2 vTextureCoord;
	uniform sampler2D uSampler;
	uniform sampler2D uSampler2;
	uniform int uDoTexturing;
// ...
```

Fragment Shader

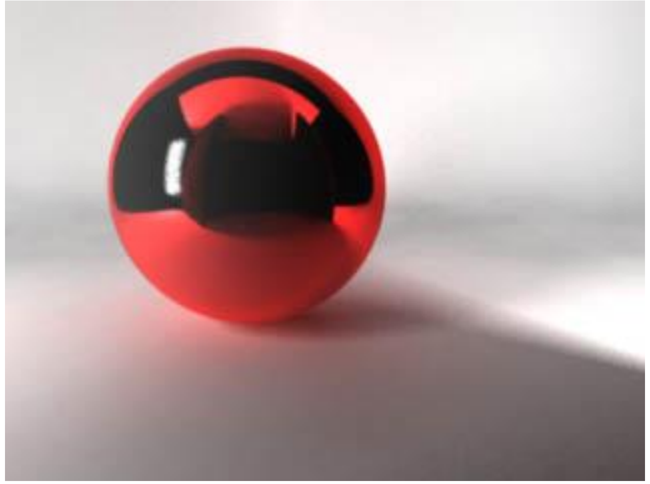
```
void main(void) {
        if(uDoTexturing == 1){ // check for texturing flag
                highp vec4 stoneColor = texture2D(uSampler,
                                                   vec2(vTextureCoord.st));
                highp vec4 webglLogoColor = texture2D(uSampler2,
                                                   vec2(vTextureCoord.st));
                gl_FragColor = mix(stoneColor, webglLogoColor, 0.5);
        }else{
                gl_FragColor = vec4(1.0, 0.1, 0.1, 1.0);
        }
}
</script>
```


1

Lights



1. Learn about transmission of light
2. Learn about light-object phenomena
3. Learn about illumination models



The Illumination Problem

Illumination Models

- Specify light propagation
- Specify light-surface interactions
- Note:

Illumination → Shading

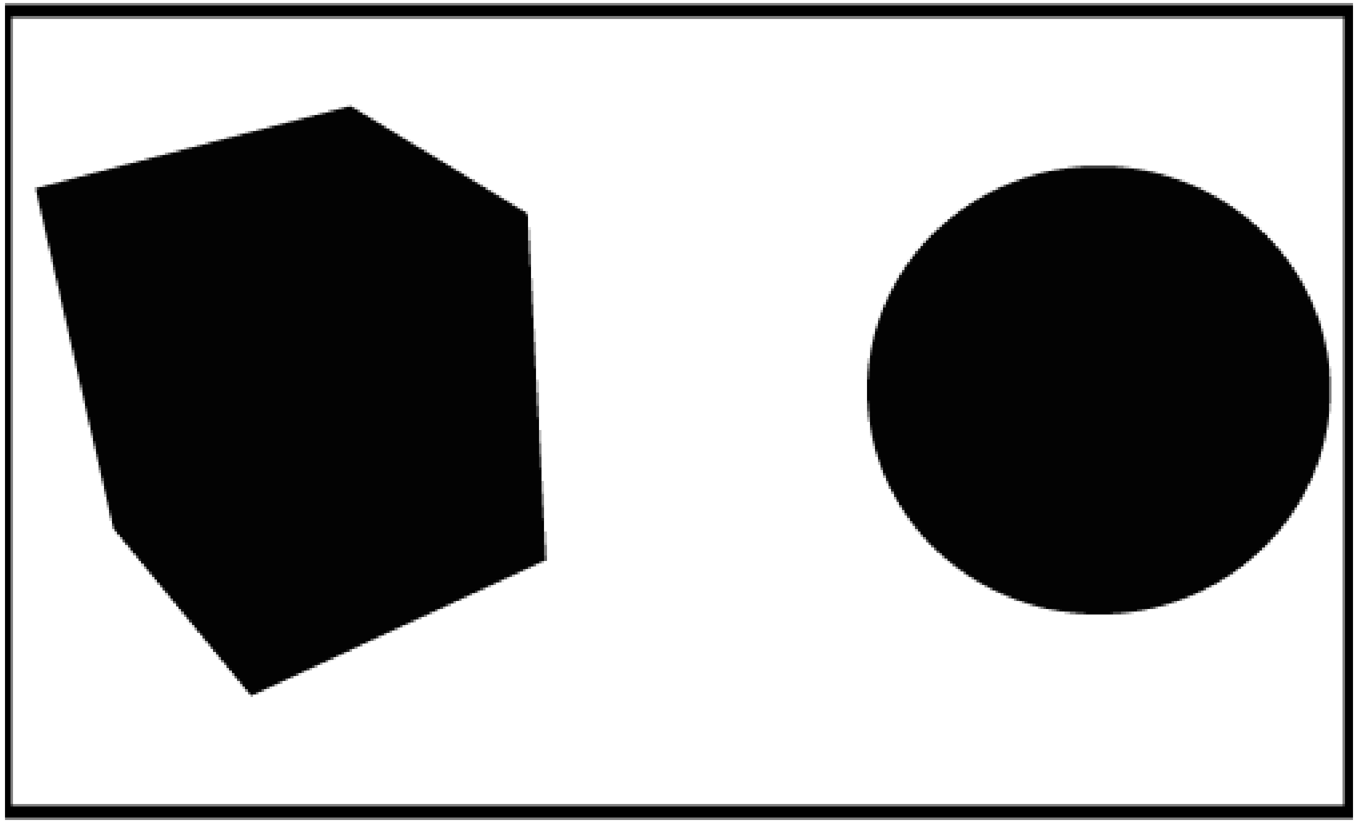
Shading Models

- Specify the color of each surface point P(x,y,z)

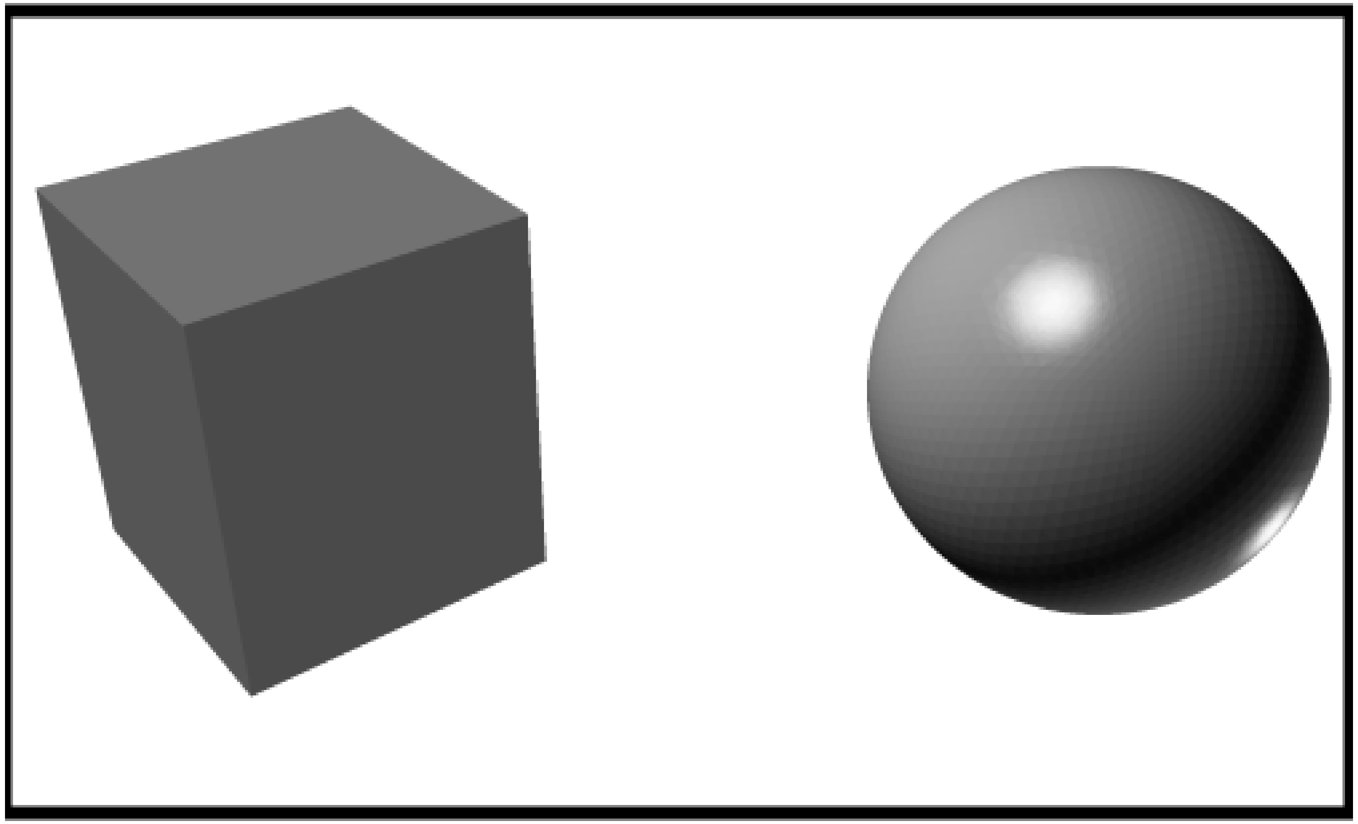
- Employ an illumination model

- Determine when to compute illumination

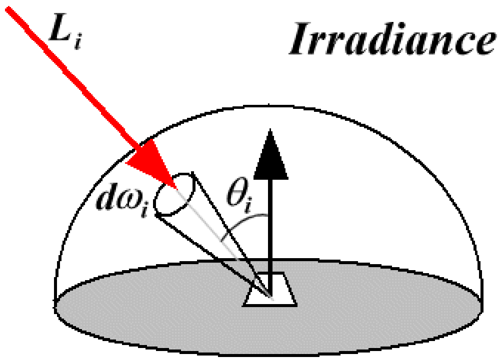
Example: No lights



Example: Two Lights



Part 2: Complete Model



$$E_i = \int_{\Omega_i} L_i \cos\theta_i \, d\omega_i$$

Light Models

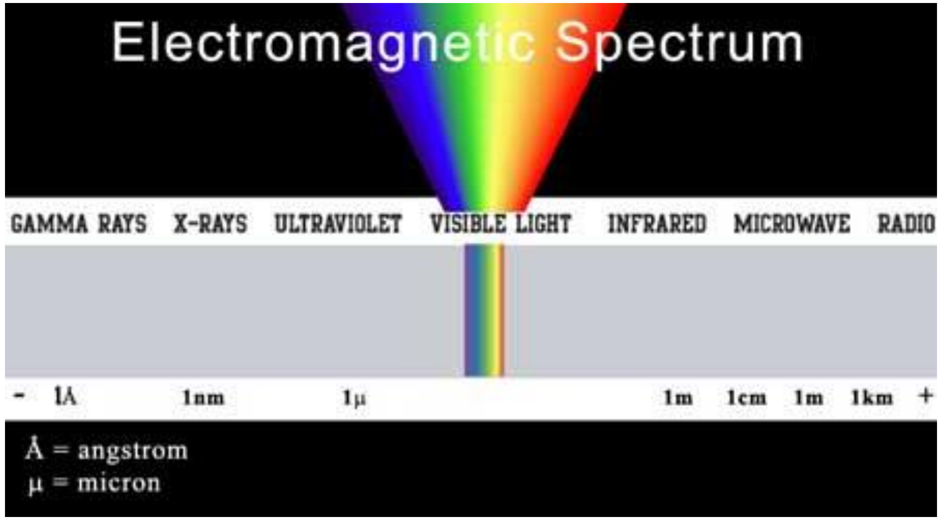
Reflection Function

$$I_r(x, y, z) = \int_{t=-\infty}^{\infty} \int_{\lambda=400}^{700} \int_{\varphi=0}^{\pi/2} \int_{\theta=0}^{\pi} L(t, x, y, z, \varphi, \theta, \lambda)\, R(t, \varphi, \theta, \lambda)\, d\theta\, d\varphi\, d\lambda\, dt$$

Irradiance at Point (x,y,z)

where:
x, y, z = the coordinates of the point on the surface
t = time
λ = wavelength
φ = azimuthal angle (from Z axis)
θ = angle about Z axis

Light



- Light is electromagnetic radiation.
- At any **point**, at any **time**, one can measure the "flow" of light through that point in a given **direction:**
  - The plenoptic function describes the light in a region:
    - $\rho(\mathbf{x}, \theta, \phi, t)$;
  - The plenoptic function over an area defines
    - the *light field* in that region;

Light as Flux

- Radiance:

The radiant flux	*per unit wavelength* ($d\,\lambda$)
	per unit solid angle ($d\,\omega$)
	per unit of projected area of the source ($d\,A$)

- Units:

watts	*per nanometer*
	per steradian
	per square meter

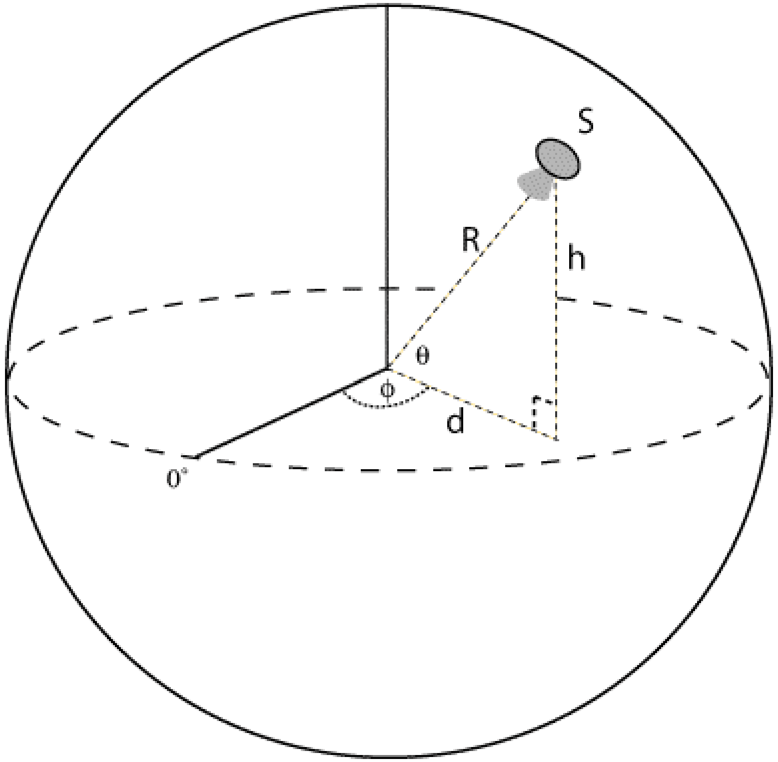
Energy per unit time = power

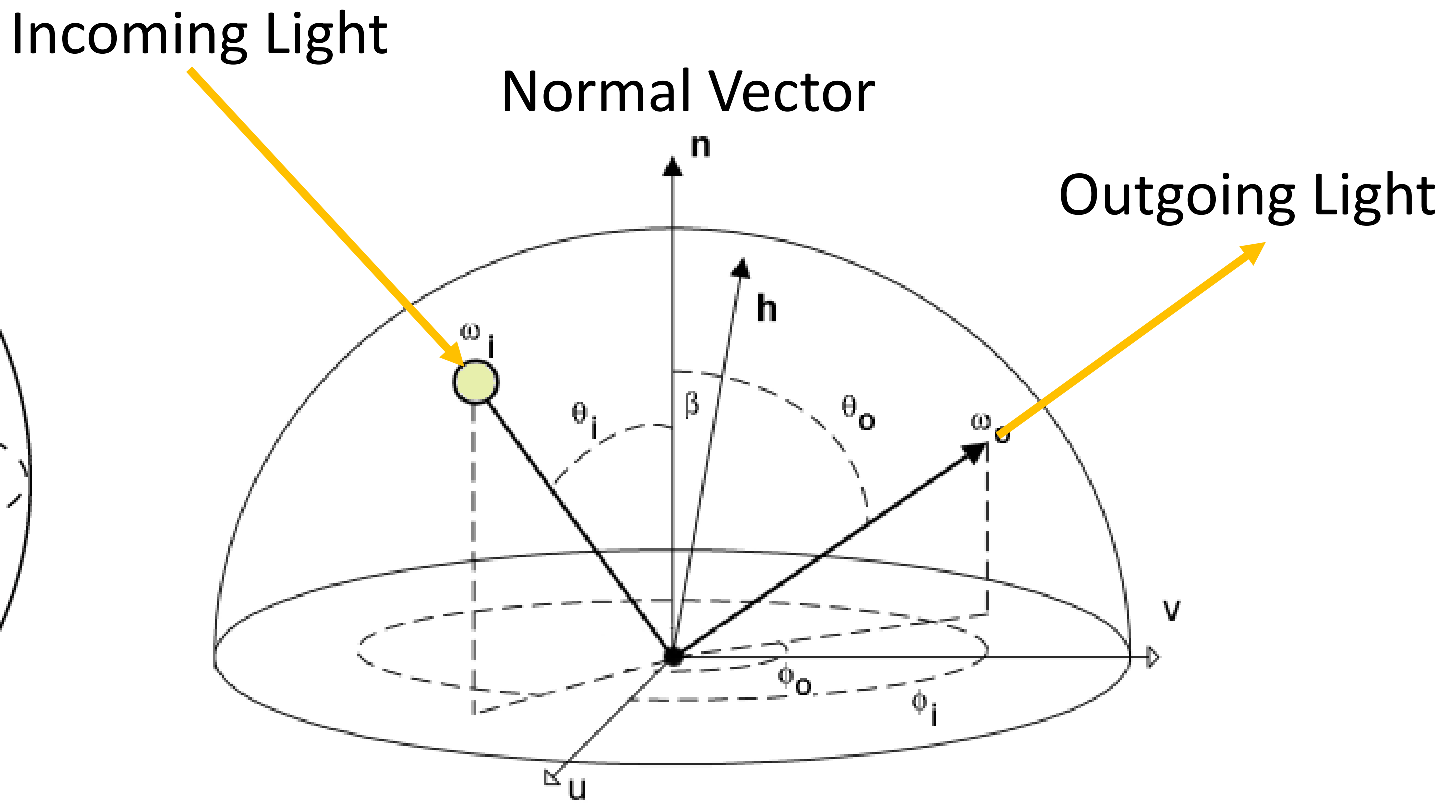
Irradiance

- Irradiance:
  The radiance flux emitted by a light source.

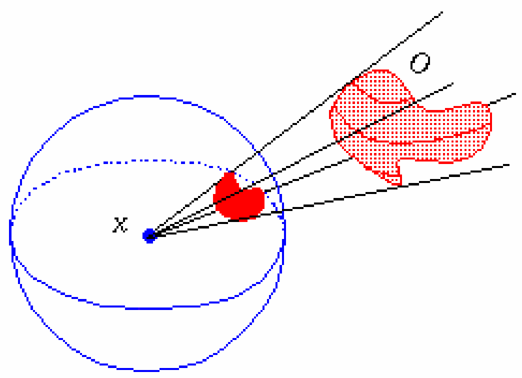
- Units: the same as radiance
  watts *per nanometer*
  per steradian
  per square meter

Spherical Geometry





Solid Angle



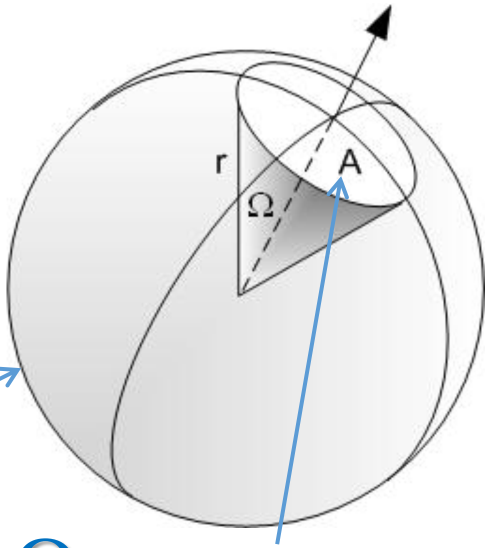
- Solid Angle Ω:

  The solid angle subtended by a surface S, is the surface area Ω of a unit sphere covered by the surface projection onto the sphere;

  infinitesimal solid angle is ($d\omega$);

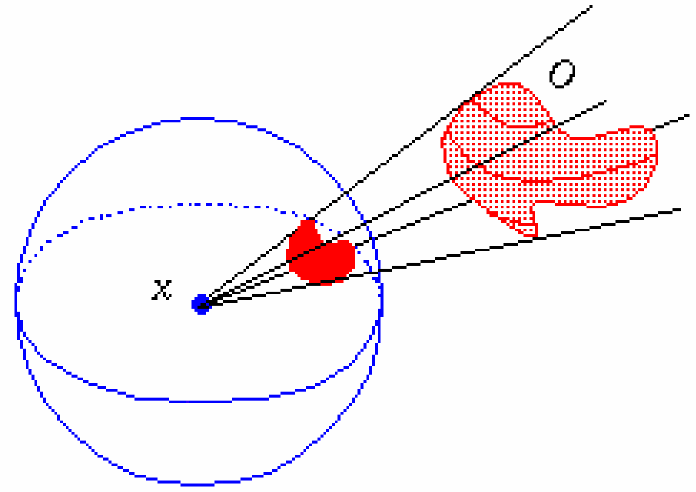
- What does this mean?:

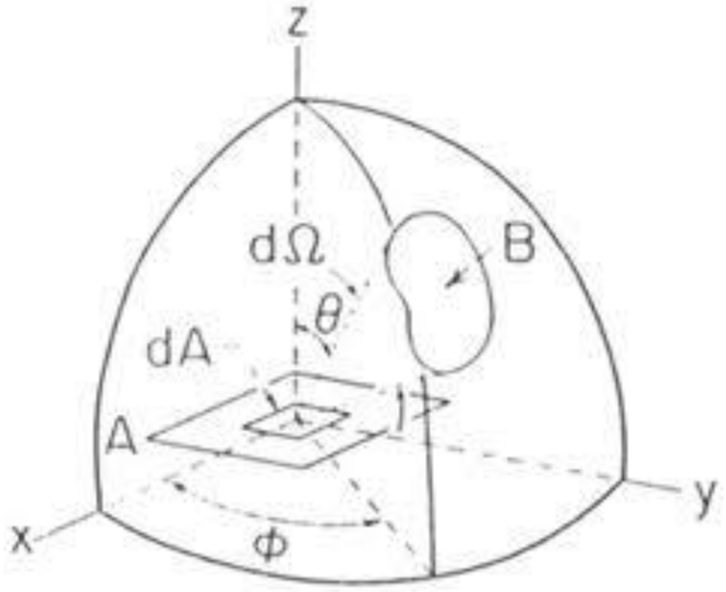
  1. Take an arbitrary surface
  2. Project it onto a unit sphere of radius 1
  3. Calculate the surface area of the projection: Ω



Solid Angle Geometry

- Solid Angle Ω:





- Geometrically:

Simplified Light Models

- Purpose:

  to simulate certain aspects of physical light transmission and reflection

- Note:

  NOT a full imitation of reality

Light-Object Interface

Reflection

Transmission

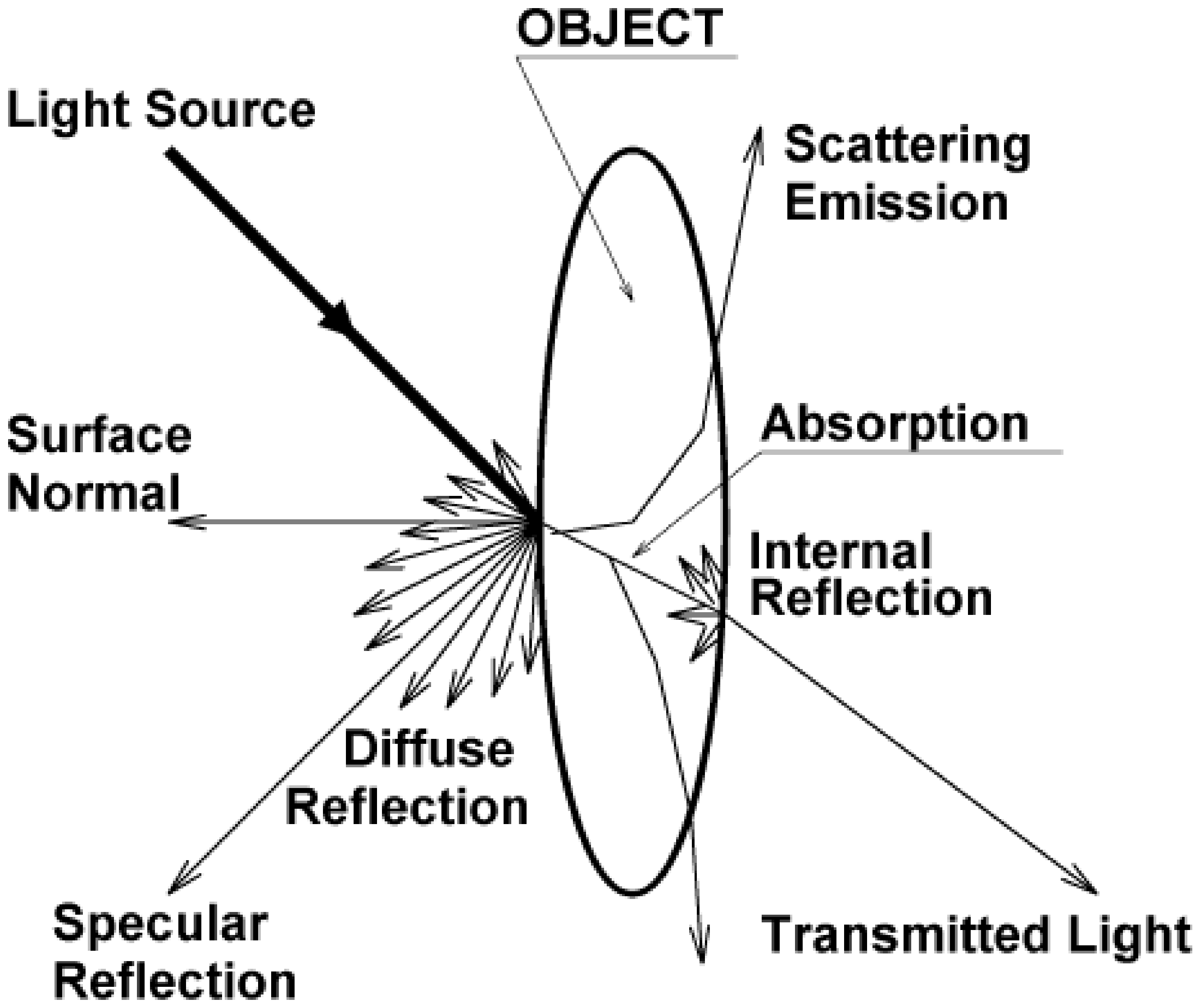
Absorption

Diffraction

Refraction

Interference

Light-Object Interactions



Other Interactions

- Diffraction:

  altering the path of light without any collision in a magnetic field.

- Absorption:

  energy diminishes as light penetrates an object.

- Interference:

  Canceling or amplification of coexisting waves producing interference patterns.

Light Behavior

- Intensity and wavelength of reflected light depend on:

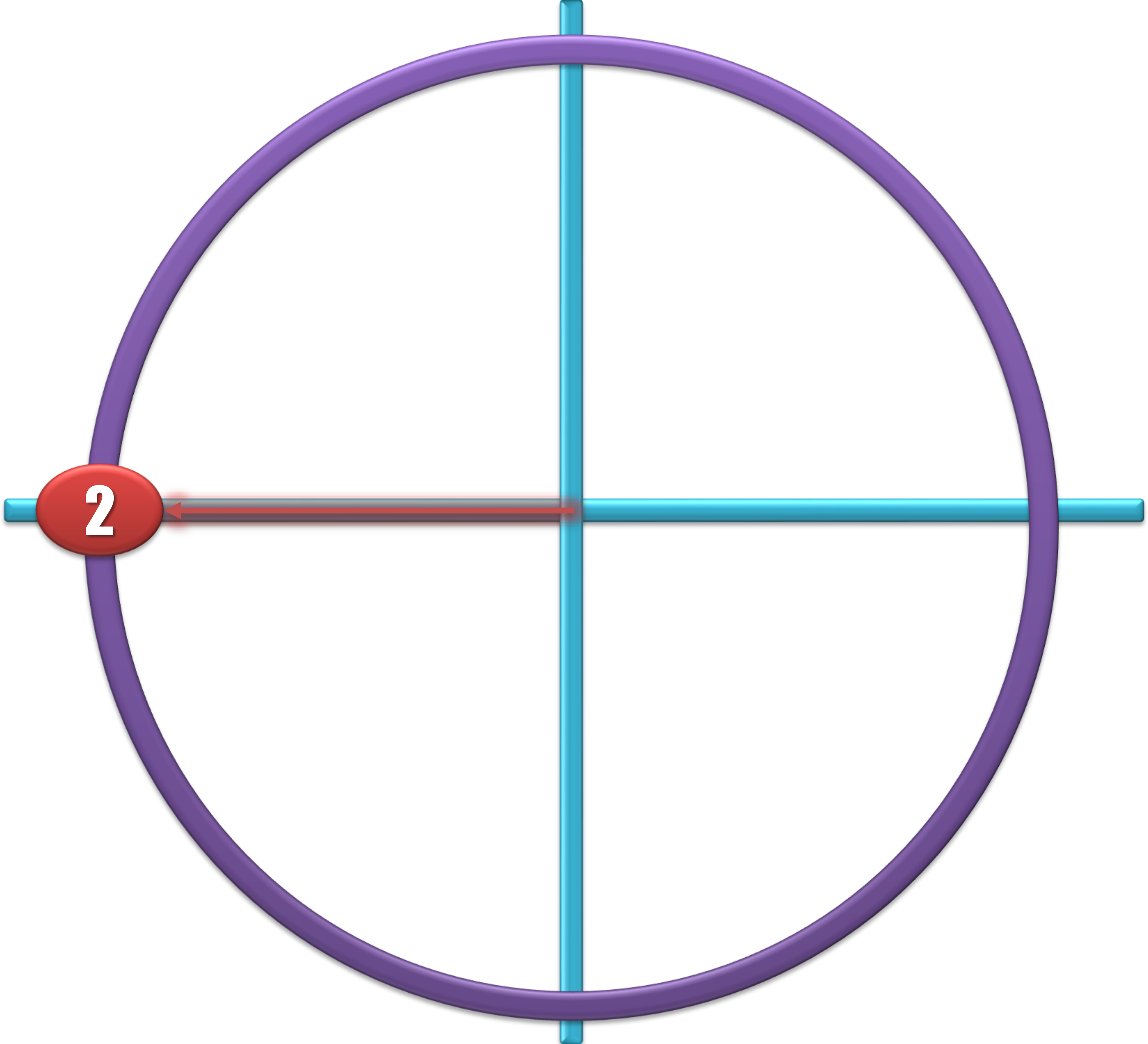
  1. angle of incidence
  2. surface geometry
  3. material properties

     Eg. Permeability, conductivity, temperature

In general...

Concentrate on

- Incident light = reflected light
  + scattered light
  + absorbed light
  + transmitted light


2

Light Models

- Local models:
  1. Lambert
  2. Phong
  3. Cook and Torrance
- Global models:
  1. Ray tracing (Appel 1968)
  2. Radiosity (Goral 1984)

Local Models

- Two distinct approximations:
  1. Surface geometry (normals)
  2. Transmission of light (instant)
- Reflected light is the sum of:
  1. Ambient term
  2. Diffuse term
  3. Specular term

Ambient Light

- Result of multiple reflections
- Incident on a surface from all directions
- Modeled as a constant term:

$$\boxed{I_a \, k_a}$$

  - First global diffuse approximation
  - No light-object interactions

Diffuse Surfaces

- A perfect diffuser:

1. Scatters light equally in all directions
2. Reflected light does NOT depend on the viewer's position
3. Effect: dull or matte tone

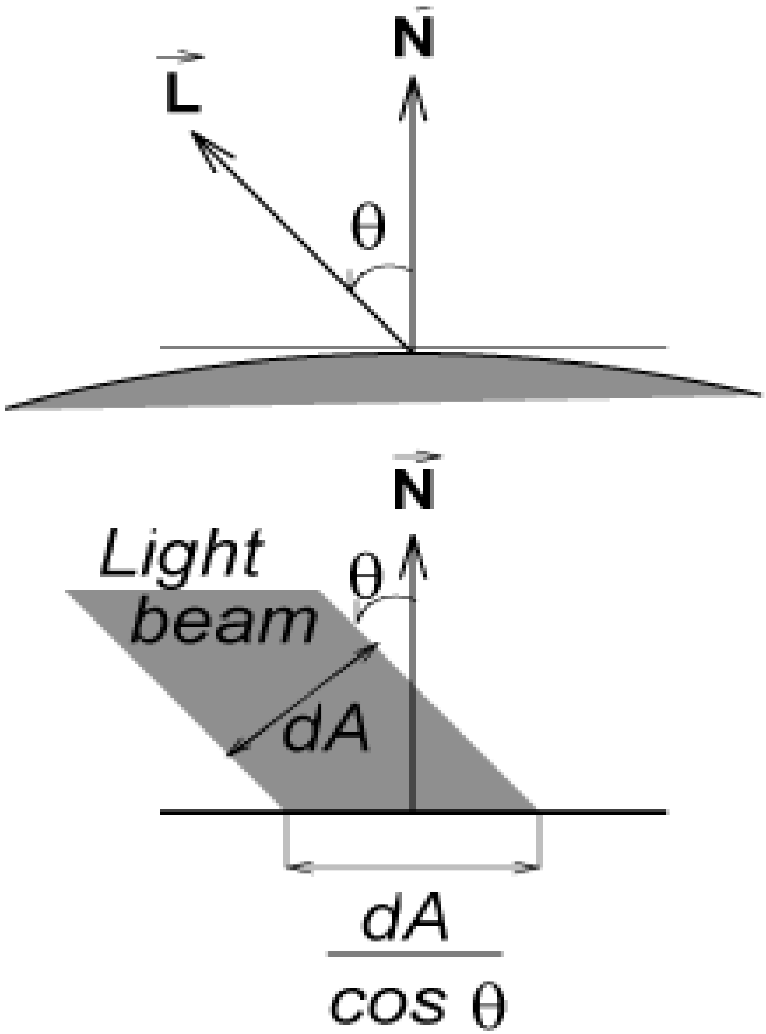
Diffuse Reflection

- A surface reflects colored light when illuminated by white light due to absorption of some wavelengths and reflection of others.

- Modeled from Lambert's Law:

$$\boxed{I_d = I_p\, k_d \cos(\theta)} \qquad 0 \le \theta \le \frac{\pi}{2}$$

Lambert's Law



Lambert's Law

The amount of light energy that falls on infinitesimal area dA is proportional to $\cos(\theta)$

Diffuse Model

$$\boxed{I_d = I_p\, k_d \cos(\theta)}$$

$$0 \le \theta \le \frac{\pi}{2}$$

I_p = intensity of incident light

k_d = diffuse reflectivity constant

θ = angle between:

1. surface normal
2. direction of light source

Multiple Diffuse Reflections

$$I_d = I_p\, k_d \cos(\theta)$$

- We can also re-write it as:

$$\boxed{I_d = I_p\, k_d (L \cdot N)}$$

- Multiple light sources:

$$I_d = k_d \sum_{i=1}^{n} I_{pi} \left(L_i \bullet N \right)$$

Total Diffuse+Ambient

$$\boxed{I = I_a k_a + I_p k_d (L \cdot N)}$$

Attenuation

- Introduce a factor which modifies the light intensity depending on the distance from the viewer:

$$I_d = I_a k_a + f_{att} I_p k_d (L \cdot N)$$

Attenuation (1)

- Inverse square law:

$$f_{att} = \frac{1}{d^2}$$

- Note: in practice it does not work

1. If d is too large then f does not vary much
2. If d is too small then f varies too much

Attenuation (2)

- Modified inverse square law:

$$f_{att} = \min\left(\frac{1}{c_1 + c_2 d + c_3 d^2},\ 1\right)$$

- c_1, c_2, c_3 are associated with light

  c_1: keeps denominator from being small

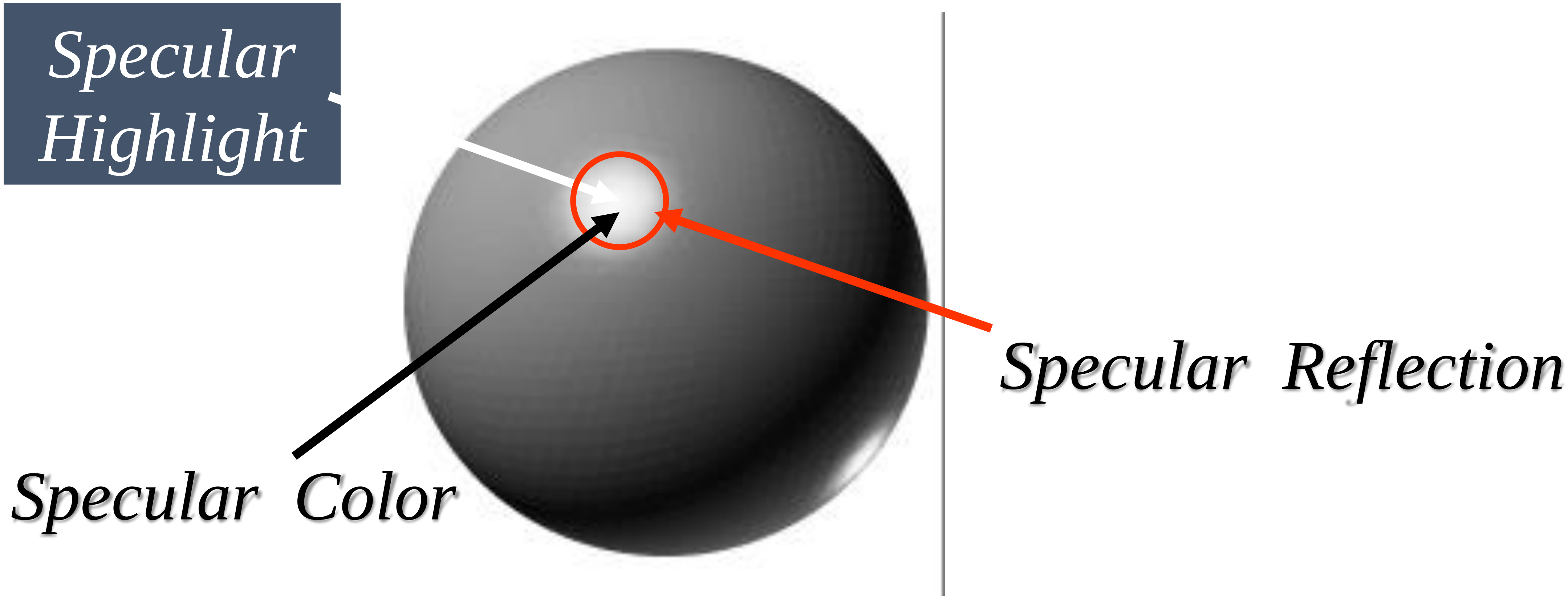
  c_2: controls d

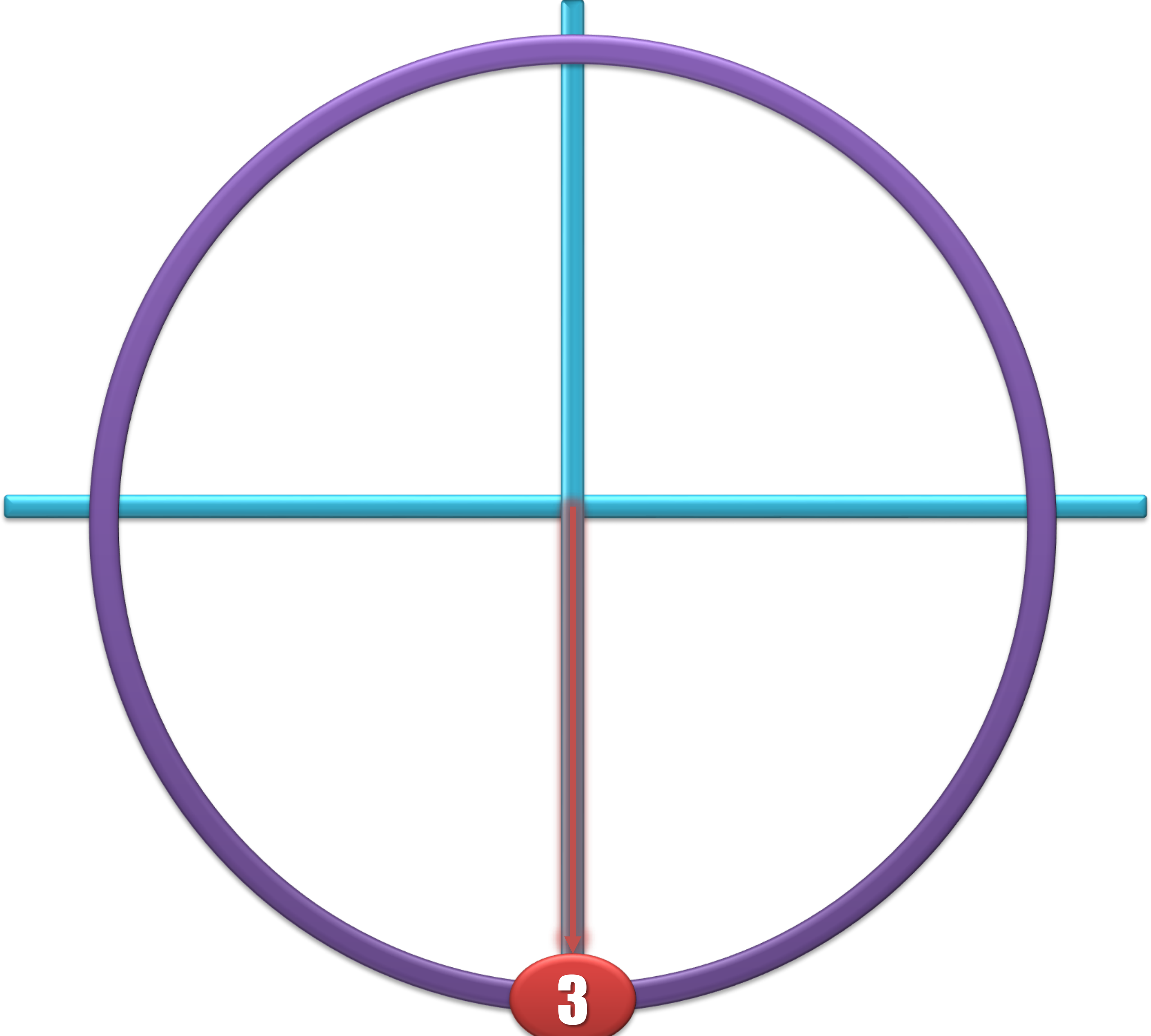
  c_3: controls d^2

Specular Reflection

- Models light reflected off a glossy surface.
- Distribution of reflected light is NOT the same in all directions
- Mirrors : perfect specular reflection

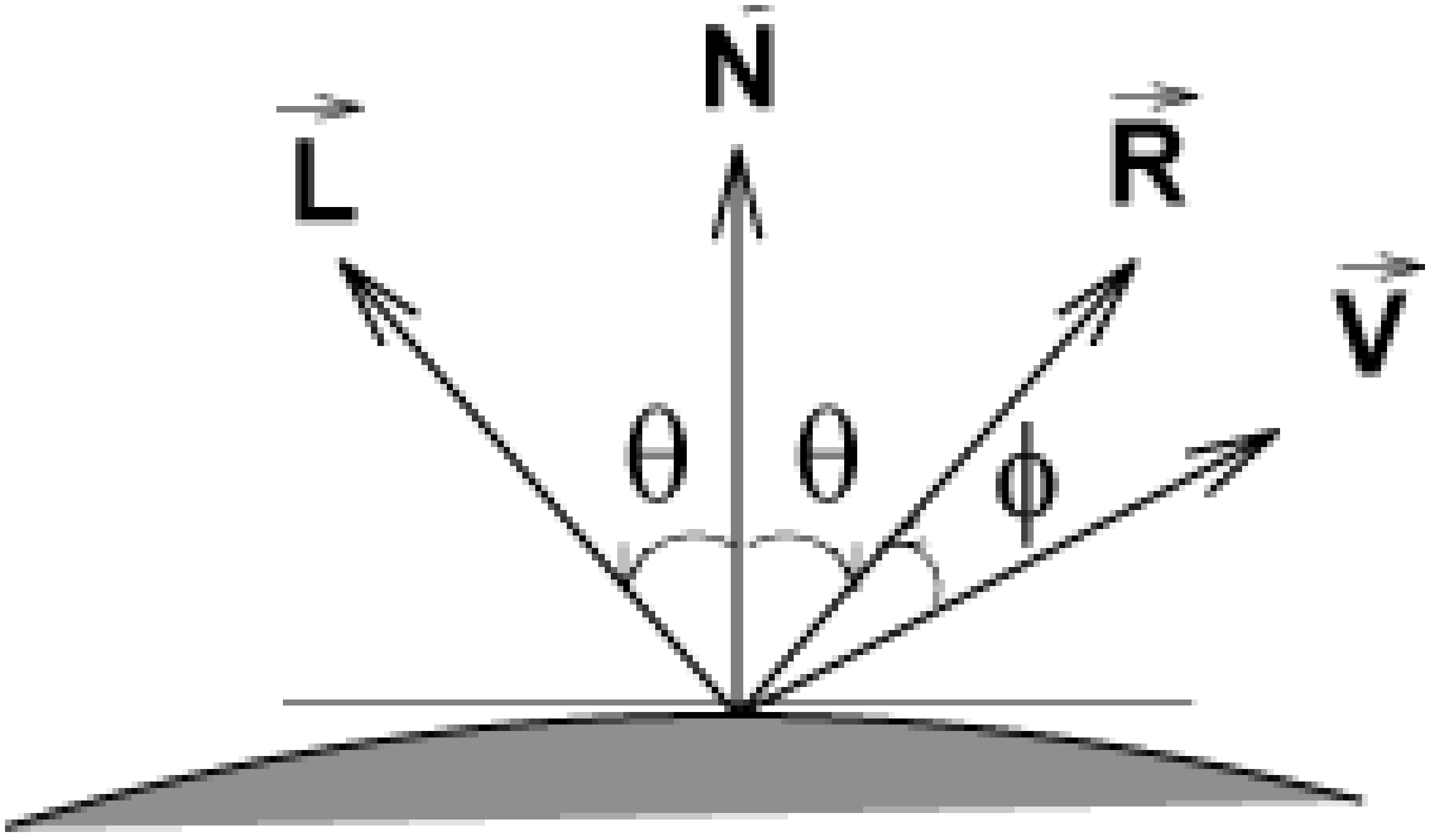
Specular Reflection




3

Phong Illumination Model

Phong Specular Model



Bui-Tong Fong (1975)

Phong Model

- Approximate specularity by:

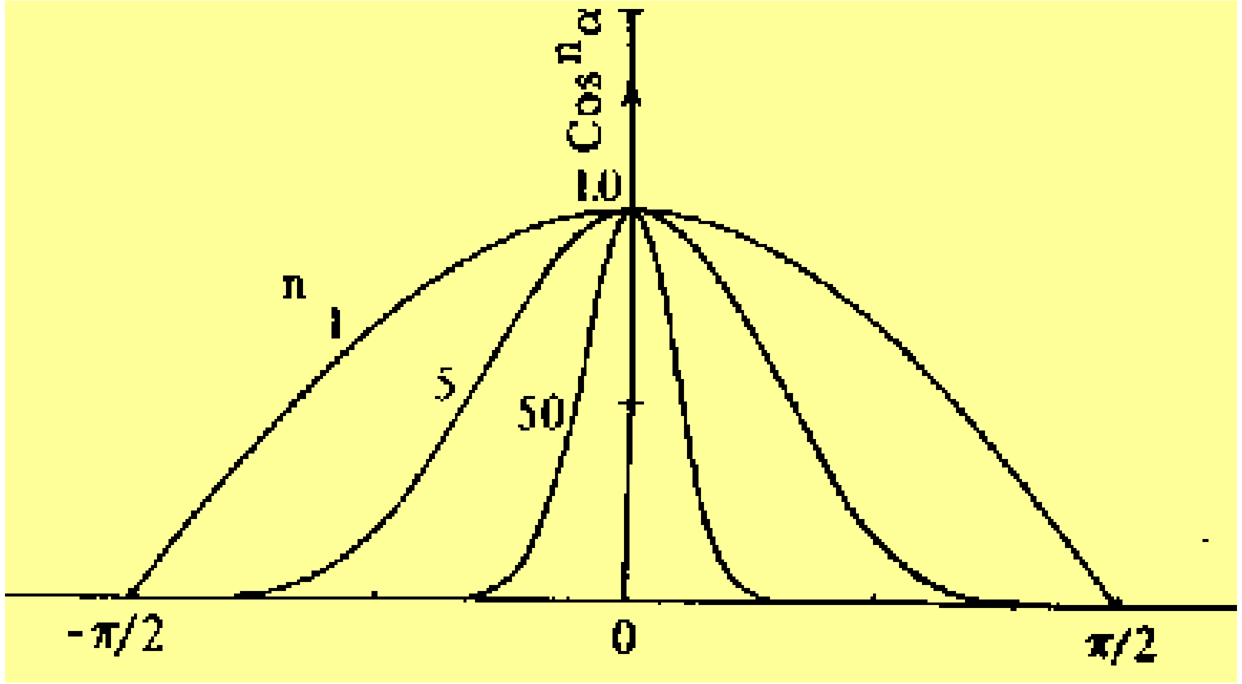
$$I_p k_s \cos^n(\varphi)$$

- ϕ = angle between R & V
- n = controls the spread

n is large $\Rightarrow$ glossy reflector
n is small $\Rightarrow$ Matte/dull

Specular Coefficient

- Light intensity as a function of specular coefficient – as n goes to infinity we get perfect mirrors.



Total Intensity

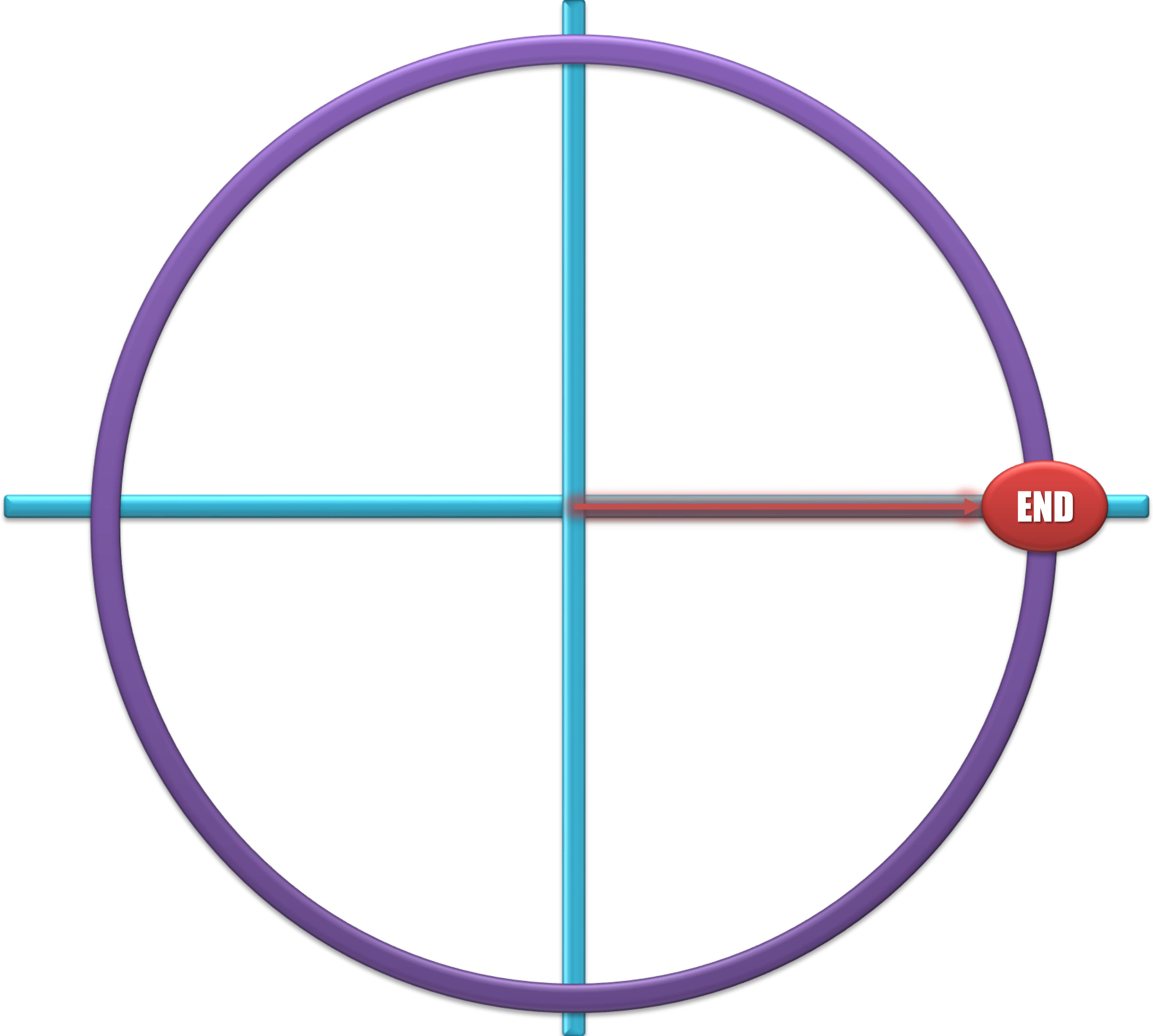
$$\boxed{I = I_a\, k_a + I_p\, (k_d\, (L \cdot N) + k_s \cos^n(\varphi))}$$

OR

$$I = I_a\, k_a + I_p\, (k_d\, (L \cdot N) + k_s\, (R \cdot V)^n)$$

Summary

- Covered light transmission
- Identify light-object phenomena
- Two major light effects:
  - Reflection: diffuse and specular
  - Transmission
- Phong Illumination Model


END