



Visual Realism Requirements

- Light Sources
- Materials (e.g., plastic, metal)
- Shading Models
- Depth Buffer Hidden Surface Removal
- Textures
- · Reflections
- Shadows

A CARL TOWN CASE 11-1-1-1



Rendering Objects

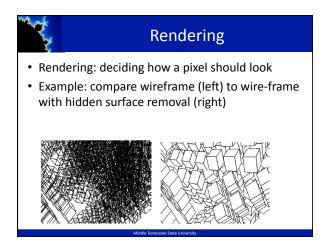
- We know how to model mesh objects, manipulate a jib camera, view objects, and make pictures.
- Now we want to make these objects look visually interesting, realistic, or both.
- We want to develop methods of rendering a picture of the objects of interest: computing how each pixel of a picture should look.

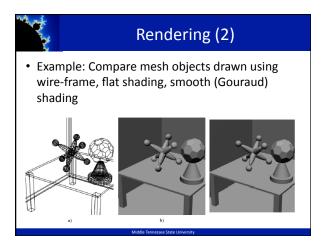
fiddle Tennessee State University

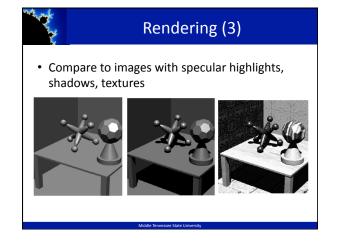


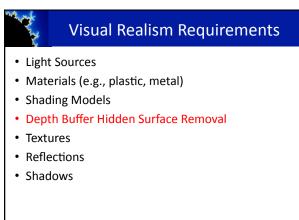
Rendering Objects (2)

- Much of rendering is based on different shading models, which describe how light from light sources interacts with objects in a scene.
 - It is impractical to simulate all of the physical principles of light scattering and reflection.
 - A number of approximate models have been invented that do a good job and produce various levels of realism.











Hidden Surface

- Hidden Surface Removal is very important in 3D scenes
- Only surfaces closest to the eye should be seen and objects that are hidden by others should be eliminated.
- The use of a depth buffer facilitates hidden surface removal.

Middle Tennessee State University



Hidden Surface Removal

- To use depth-buffering, you need to enable it: glutInitDisplayMode (GLUT_DEPTH | ..); glEnable (GL_DEPTH_TEST);
- Initialize the depth buffer and color buffer by using:

glClear(GL_DEPTH_BUFFER_BIT |
GL_COLOR_BUFFER_BIT);

Park Toronto Control (1971)



Hidden Surface Removal

• Put them together:

glutInitDisplayMode(GLUT_DEPTH|..);
glEnable(GL_DEPTH_TEST);

// in display function

glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
draw3D0bjectA();

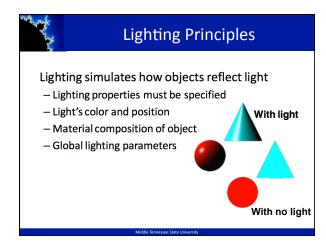
draw3DObjectB();

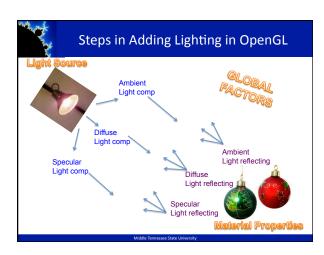
Middle Tennessee State University

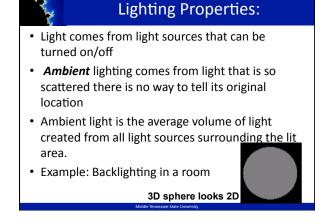


Visual Realism Requirements

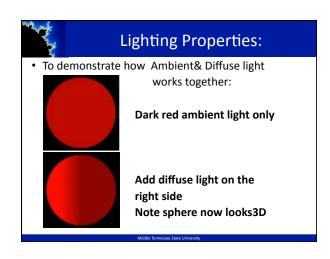
- Light Sources
- Materials (e.g., plastic, metal)
- Shading Models
- Depth Buffer Hidden Surface Removal
- Textures
- Reflections
- Shadows

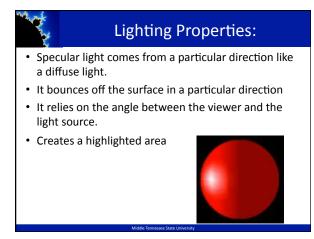




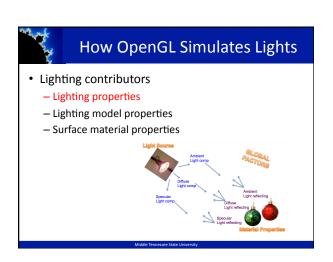








RGB Values for Lights A light source is characterized by the amount of red, green, & blue light it emits. Examples: If R=G=B=1.0, the light is the brightest possible white. If R=G=B=.5, the color is still white, but only at half intensity, so it appears gray If R=G=1.0 and B=0.0, the light appears yellow.





Setting Lighting Properties

glLightfv(light, property, value);

- · light specifies which light
 - multiple lights (at least 8), starting with GL_LIGHT0
- Properties
 - Colors for ambient, diffuse, & specular component
 - position and type
 - attenuation

Middle Tennessee State Heiserits



Types of Lights

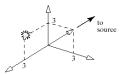
- · OpenGL supports two types of Lights
 - Local (Point) light sourcess of Light
 - Local light positioned at (x, y, z, 1)
 - Infinite (Directional) light sources
 - Infinite light directed along (x, y, z, 0)

ARIAN Toronto Cara Indiana



Point and Vector Light Locations

- The figure shows a local source at (0, 3, 3, 1) and a remote source "located" along vector (3, 3, 0, 0).
- Infinitely remote light sources are often called "directional". There are computational advantages to using directional light sources, since direction s in the calculations of diffuse and specular reflections is constant for all vertices in the scene.
- But directional light sources are not always the correct choice: some visual effects are properly achieved only when a light source is close to an object.





Creating and Using Light Sources in Open-GL

- Each light has a position specified in homogeneous coordinates using a GLfloat array named, for example, litePos.
 - Glfloat litePos[4]={3, 3, 1, 1};
- The light is created using glLightfv (GL_LIGHTO, GL_POSITION, litePos);
- If the position is a vector (4th component = 0), the source is infinitely remote (like the sun).



Creating and Using Light Sources in OpenGL (2)

- The light color is specified by a 4-component array [R, G, B, A] of GLfloat, named (e.g.) amb0.
 The A value can be set to 1.0 for now:
 Glfloat amb0={0.2, 0.8, 0.0, 1.0};
- The light color is specified by glLightfv (GL_LIGHT_0, GL_AMBIENT, amb0);
- Similar statements specify GL_DIFFUSE and GL_SPECULAR.

Middle Tennessee State University



Creating and Using Light Sources in OpenGL (3)

- Lights do not work unless you turn them on.
 - In your main program, add the statements

glEnable (GL_LIGHTING);
glEnable (GL_LIGHT0);

- If you are using other lights, you will need to enable them also.
- To turn off a light

glDisable (GL_LIGHT0);

To turn them all off,

glDisable (GL_LIGHTING);

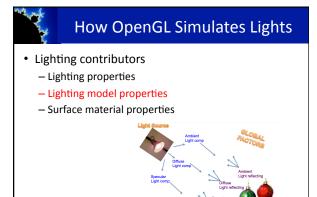
A CARL TOWN CASE 11-1-1-1



Creating an Entire Light

// define some colors GLfloat amb0[] = {0.2, 0.4, 0.6, 1.0}; GLfloat diff0[] = {0.8, 0.9, 0.5, 1.0}; GLfloat spec0[] = {1.0, 0.8, 1.0, 1.0};

// attach them to LIGHT0
glLightfv(GL_LIGHT0, GL_AMBIENT, amb0);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diff0);
glLightfv(GL_LIGHT0, GL_SPECULAR, spec0);





Creating and Using Light Sources in OpenGL (4)

- Global ambient light is present even if no lights are created. Its default color is {0.2, 0.2, 0.2, 1.0}.
- To change this value, create a GLfloat array of values newambient and use the statement

glLightModelfv (GL_LIGHT_MODEL_AMBIENT, newambient);

- Default light values are:
 - {0, 0, 0, 1} for ambient for all lights,
 - {1, 1, 1, 1} for diffuse and specular light for LIGHT_0, and
 - {0, 0, 0, 1} for diffuse and specular light for lights LIGHT_1 to LIGHT_7

Middle Tennessee State University



Spotlights in Open-GL

• A spotlight emits light only in a cone of directions; there is no light outside the cone. Inside the cone, $I = I_s(\cos \beta)^{\epsilon}$, where $\cos \beta$ uses the angle between **d** and a line from the source to *P*.

TIB P

Middle Tennessee State Heiserits



Spotlights in OpenGL (2)

- To create the spotlight, create a GLfloat array for d. Default values are d = $\{0, 0, 0, 1\}$, $\alpha = 180^{\circ}$, $\epsilon = 0$: a point source.
- Then add the statements:

Glfloat d={2, 2, 2, 1};
glLightf (GL_LIGHTO, GL_SPOT_CUTOFF, 45.0); //45.0 is α in degrees)
glLightf (GL_LIGHTO, GL_SPOT_EXPONENT, 4.0); //4.0 is ε)
glLightfv (GL_LIGHTO, GL_SPOT_DIRECTION, d);

Middle Tennessee State University



Attenuation of Light with Distance

- OpenGL also allows you to specify how rapidly light diminishes with distance from a source.
- OpenGL attenuates the strength of a positional light source by the following attenuation factor:

$$atten = \frac{1}{k_c + k_l D + k_q D^2}$$

where k_c , $k_{\rm p}$ and $k_{\rm g}$ are coefficients and D is the distance between the light s position and the vertex in question.



Attenuation of Light with Distance (2)

$$atten = \frac{1}{k_c + k_l D + k_q D^2}$$

• These parameters are controlled by calling:

glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 2.0); glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, 2.0); glLightf(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, 2.0);

The default values are $k_{\rm c}$ = 1, $k_{\rm l}$ = 0, and $k_{\rm q}$ = 0 (no attenuation).

Middle Tennessee State University



Moving Light Sources in OpenGL

- To move a light source independently of the camera:
 - set its position array,
 - clear the color and depth buffers,
 - set up the ModelView matrix to use for everything except the light source and push the matrix
 - move the light source and set its position
 - pop the matrix
 - set up the camera, and draw the objects.

Middle Tennessee State University



Changing the OpenGL Light Model

- The color of global ambient light: specify its color using: GLfloat amb[] = {0.2, 0.3, 0.1, 1.0}; glLightModelfv(GL_LIGHT_MODEL_AMBIENT, amb);
- · Is the viewpoint local or remote?
 - OpenGL computes specular reflections using the "halfway vector" h = s + v . The true directions s and v are normally different at each vertex in a mesh.
 - OpenGL uses ${\bf v}$ = (0, 0, 1), along the positive z-axis, to increase rendering speed. To use the true value of ${\bf v}$ for each vertex, execute

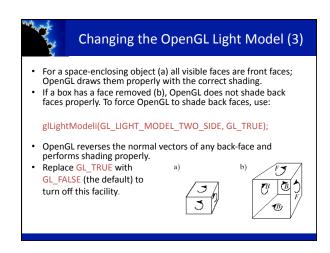
glLightModeli(GL_LIGHT_MODEL_LOCAL_VIEWER, GL_TRUE);

Middle Tennessee State University



Changing the OpenGL Light Model (2)

- Are both sides of a polygon shaded properly?
 Each polygonal face in a model has two sides. When modeling, we tend to think of them as the "inside" and "outside" surfaces. The convention is to list the vertices of a face in counter-clockwise (CCW) order as seen from outside the object.
- OpenGL has no notion of inside and outside. It can only distinguish between "front faces" and "back faces". A face is a front face if its vertices are listed in counterclockwise (CCW) order as seen by the eye.







Steps in Adding Lighting in OpenGL (1)

Step 1: Establish vertex normal

- 1. Make sure the normal for each vertex has been defined
- 2. Draw with:

glNormal3f(0.0, 0.0, 1.0); glVertex3f(-50.0, 50.0, 10.0);

3. One normal for one vertex, or one normal for a group of vertices

Middle Tennessee State University



Steps in Adding Lighting in OpenGL (2)

Step 2: Enable/Define the necessaries:

- glEnable(GL_LIGHTING);
- glEnable(GL_DEPTH_TEST); // perform hidden depth removal
- glEnable(GL_NORMALIZE); // !!! important to get the right shading !!!
- glShadeModel (GL_SMOOTH);
- Optional:
 - glFrontFace(GL_CCW);



Steps in Adding Lighting in OpenGL (3)

Step 3. Setup the Light source

```
\label{eq:GLfloat_sourceLight[] = { 0.25, 0.25, 0.25, 1.0 }; \\ Glfloat specularLight[] = { 0.8, 0.8, 0.8, 1.0 }; \\ GLfloat lightPos[] = { -50.0, 25.0, 250.0, 0.0 }; \\ Glfloat shininess[] = { 50.0 }; \\ \\ glLightfv(GL_LIGHT0,GL_AMBIENT,sourceLight); \\ \\ glLightfv(GL_LIGHT0,GL_DIFFUSE,sourceLight); \\ \end{aligned}
```

glLightfv(GL_LIGHT0,GL_DIFFUSE,specularLight); glLightfv(GL_LIGHT0,GL_POSITION,lightPos);

glEnable(GL LIGHT0);

A STATUTE TO STATE OF THE STATE



Steps in Adding Lighting in OpenGL (4)

Step 3. Setup the Light source continued ...

- Optional:
 - Setup multiple light sources (at different locations)
 - Specify certain light source as spot light (α and ϵ)

Eiddle Tennessee State University



Steps in Adding Lighting in OpenGL (5)

Step 4. Setup the material properties of each object

```
\begin{split} & \text{GLfloat mat\_color[]} = \{0.23, 0.23, 0.23, 1.0 \}; \\ & \text{GLfloat mat\_specular[]} = \{0.5, 0.5, 0.5, 1.0 \}; \\ & \text{GLfloat mat\_shininess[]} = \{100.0 \}; \end{split}
```

glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular); glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess); glMaterialfv(GL_FRONT, GL_AMBIENT, mat_color); glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_color);

Middle Tennessee State University



Steps in Adding Lighting in OpenGL (6)

Step 5. Define various options in the global shading model

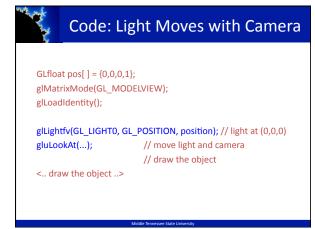
```
GLfloat amb[] = {0.2, 0.3, 0.1, 1.0};
glLightModelfv(GL_LIGHT_MODEL_AMBIENT, amb);
```

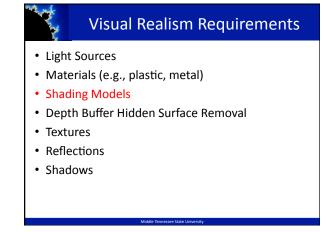
glLightModeli(GL_LIGHT_MODEL_TWO_SIDE, GL_TRUE);

glLightf(GL_LIGHTO, GL_CONSTANT_ATTENUATION, 2.0); glLightf(GL_LIGHTO, GL_LINEAR_ATTENUATION, 2.0); glLightf(GL_LIGHTO, GL_QUADRATIC_ATTENUATION, 2.0);

•••









Shading Models: Introduction

- Assume to start with that light has no color, only brightness: R = G = B
- Assume we also have a point source of light (sun or lamp) and general ambient light

Middle Tennessee State University



Shading Models: Introduction (2)

- When light hits an object, some light is absorbed (and turns into heat), some is reflected, and some may penetrate the interior (e.g., of a clear glass object).
- If all the light is absorbed, the object appears black and is called a blackbody.
- If all the light is transmitted, the object is visible only through refraction.

Middle Tonnersee State Heimstitu



Shading Models: Introduction (3)

- When light is reflected from an object, some of the reflected light reaches our eyes, and we see the object.
 - Diffuse reflection: some of the light slightly penetrates the surface and is re-radiated uniformly in all directions. The light takes on some fraction of the color of the surface.
 - Specular reflection: more mirror-like. Light is reflected directly from the object's outer surface, giving rise to highlights of approximately the same color as the source. The surface looks shiny.

Middle Tennessee State University



Shading Models: Introduction (4)

- In the simplest model, specular reflected light has the same color as the incident light. This tends to make the material look like plastic.
- In a more complex model, the color of the specular light varies over the highlight, providing a better approximation to the shininess of metal surfaces.



Shading Models: Introduction (5)

- Most surfaces produce some combination of diffuse and specular reflection, depending on surface characteristics such as roughness and type of material.
- The total light reflected from the surface in a certain direction is the sum of the diffuse component and the specular component.
 - For each surface point of interest, we compute the size of each component that reaches the eye.

Middle Tennessee State Heiserits



Reflected Light Model

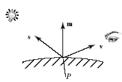
- Finding Reflected Light: a model
 - Model is not completely physically correct, but it provides fast and relatively good results on the screen
 - Intensity of a light is related to its brightness. We will use $\rm I_s$ for intensity, where s is R or G or B.

Edd Toron Control



Calculating Reflected Light

To compute reflected light at point P, we need 3 vectors: normal m to the surface at P and vectors s from P to the source and v from P to the eye.
 We use world coordinates.

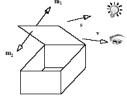


Middle Tennessee State University



Calculating Reflected Light (2)

- Each face of a mesh object has an inside and an outside.
- Normally the eye sees only the outside (front, in Open-GL), and we calculate only light reflected from the outside.





Calculating Reflected Light (3)

- If the eye can see inside, we must also compute reflections from the inside (back, in OpenGL).
 - If $\mathbf{v} \cdot \mathbf{m} > 0$, the eye can see the face and lighting must be calculated.

Middle Tennessee State Heiserits



Calculating Diffuse Light

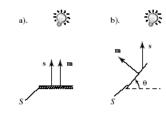
- Diffuse scattering is assumed to be independent of the direction from the point, P, to the location of the viewer's eye.
- Because the scattering is uniform in all directions, the orientation of the facet F relative to the eye is not significant, and I_d is independent of the angle between m and v (unless v · m < 0, making I_d =0.)
- The amount of light that illuminates the facet does depend on the orientation of the facet relative to the point source: the amount of light is proportional to the area of the facet that it sees: the area subtended by a facet.

A CARL TOWN CASE 11-1-1-1



Calculating Diffuse Light (2)

• The intensity depends on the projection of the face perpendicular to $\bf s$ (Lambert's law): $I_d \cos\theta$



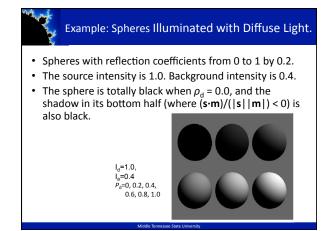
a Podallo Warrance and Parker Marker State of the



Calculating Diffuse Light (3)

- For ϑ near 0° , brightness varies only slightly with angle, because the cosine changes slowly there. As ϑ approaches 90° , the brightness falls rapidly to 0.
- We know $\cos \vartheta = (\mathbf{s} \cdot \mathbf{m})/(|\mathbf{s}| |\mathbf{m}|)$.
- $I_d = I_d \rho_d (s \cdot m) / (|s| |m|).$
 - I_d is the intensity of the source.
 - $-\ \rho_{d}$ is the diffuse reflection coefficient and depends on the material the object is made of.
- **s·m** < 0 implies I_d = 0.
- · So to take all cases into account, we use

 $I_d = I_d \rho_d \max [(s \cdot m)/(|s||m|), 0].$





Calculating the Specular Component

- Real objects do not scatter light uniformly in all directions; a specular component is added to the shading model.
- Specular reflection causes highlights, which can add significantly to realism of a picture when objects are shiny.

tiddle Tennersee State University



Calculating the Specular Component (2)

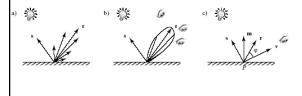
- A simple model for specular light was developed by Phong. It is easy to apply.
 - The highlights generated by the Phong model give an object a plastic-like or glass-like appearance.
 - The Phong model is less successful with objects that are supposed to have a shiny metallic surface, although you can roughly approximate them with OpenGL by careful choices of certain color parameters.

fiddle Tennessee State University



Calculating the Specular Component (3)

 Most of the light reflects at equal angles from the (smooth and/or shiny) surface, along direction r, the reflected direction.



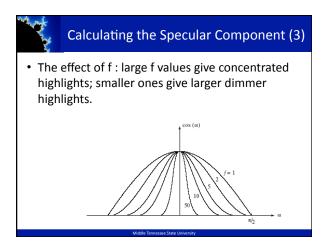
made remedee state onversey



Calculating the Specular Component (2)

- In Ch. 4, we found that $r = -s + 2 m (s \cdot m)/(|m|^2)$ (mirror reflection direction).
- For surfaces that are not mirrors, the amount of reflected light decreases as the angle \$\phi\$ between \$\mathbf{r}\$ and \$\mathbf{v}\$ (vector from reflection point to eye) increases.
- For a simplified model, we say the intensity decreases as cosf ф, where f is chosen experimentally between 1 and 200.

Middle Tennessee State Heiseriks





Calculating the Specular Component (4)

- $\cos \phi = \mathbf{r} \cdot \mathbf{v}/(|\mathbf{r}||\mathbf{v}|)$
- $I_{sp} = I_{s} \rho_{s} (\mathbf{r} \cdot \mathbf{v} / (|\mathbf{r}| |\mathbf{v}|))^{f}$.
 - $-\,\rho_s$ is the specular reflection coefficient, which depends on the material.
- If **r·v** < 0, there is no reflected specular light.
- $I_{sp} = I_s \rho_s \max[(\mathbf{r} \cdot \mathbf{v}/(|\mathbf{r}| |\mathbf{v}|))^f, 0].$

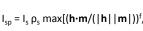
liddle Tennessee State University

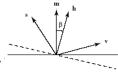


Speeding up Calculations for Specular Light

- Find the halfway vector **h** = **s** + **v**.
- Then the angle β between **h** and **m** approximately measures the falloff of intensity. To take care of errors, we use a

different f value, and write

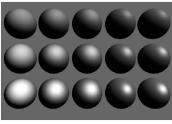






Calculating the Specular Component (5)

- From bottom to top, ρ_s = 0.75, 0.5, 0.25. From left to right, f = 3, 6, 9, 25, 200.
- ρ_a= 0.1
- $\rho_d = 0.4$



Middle Tennessee State Heisesite



Ambient Light

- Our desire for a simple reflection model leaves us with far from perfect renderings of a scene.
 - E.g., shadows appear to be unrealistically deep and harsh.
- To soften these shadows, we can add a third light component called *ambient light*.

fiddle Tennessee State University



Calculating Ambient Light

- The source is assigned an intensity, I_a.
- Each face in the model is assigned a value for its ambient reflection coefficient, $\rho_{\rm a}$ (often this is the same as the diffuse reflection coefficient, $\rho_{\rm d}$), and the term $I_{\rm a}$ $\rho_{\rm a}$ is simply added to whatever diffuse and specular light is reaching the eye from each point P on that face.
- I_a and ρ_a are usually arrived at experimentally, by trying various values and seeing what looks best.



Adding Ambient Light to Diffuse Reflection

- The diffuse and ambient sources have intensity 1.0, and ρ_d = 0.4. ρ_a = 0, 0.1, 0.3, 0.5, 0.7 (left to right).
- Modest ambient light softens shadows; too much ambient light washes out shadows.



Middle Tennessee State University



Combining Light Contributions and Adding Color

• $I = I_a \rho_a + I_s \rho_d lambert + I_{sp} \rho_s x phong^f$

where Lambert = $\max[(\mathbf{s} \cdot \mathbf{m})/(|\mathbf{s}||\mathbf{m}|), 0]$ and Phong = $\max[(\mathbf{h} \cdot \mathbf{m}/(|\mathbf{h}||\mathbf{m}|), 0]$

- To add color, we use 3 separate total intensities like that above, one each for Red, Green, and Blue, which combine to give any desired color of light.
- We say the light sources have three types of color: ambient = (I_{ar}, I_{ag}, I_{ab}) , diffuse = (I_{dr}, I_{dg}, I_{db}) , and specular = $(I_{spr}, I_{spg}, I_{spb})$.

Address Towns of the Control of the



Combining Light Contributions and Adding Color (2)

- Generally the diffuse light and the specular light have the same intensity.
- ρ_s is the same for R, G, and B, so specular light is the color of the light source.
- An object's color (in white light) is specified by 9 coefficients (ambient and diffuse are color of object):
- ambient reflection coefficients: $\rho_{\text{ar}},\,\rho_{\text{ag}},$ and $\rho_{\text{ab}},$
- diffuse reflection coefficients: $\rho_{\text{dr}},\,\rho_{\text{dg}}$ and ρ_{db}
- specular reflection coefficients: ρ_{sr} , ρ_{sg} , and ρ_{sb}

liddle Tennessee State University



Example

- If the color of a sphere is 30% red, 45% green, and 25% blue, it makes sense to set its ambient and diffuse reflection coefficients to (0.3K, 0.45K, 0.25K) respectively, where K is some scaling value that determines the overall fraction of incident light that is reflected from the sphere.
- Now if it is bathed in white light having equal amounts of red, green, and blue $(I_{sr} = I_{sg} = I_{sb} = I)$ the individual diffuse components have intensities $I_r = 0.3~K~I$, $I_g = 0.45~K~I$, $I_b = 0.25~K~I$, so as expected we see a color that is 30% red, 45% green, and 25% blue.

Middle Tennessee State University



Example (2)

- Suppose a sphere has ambient and diffuse reflection coefficients (0.8, 0.2, 0.1), so it appears mostly red when bathed in white light.
- We illuminate it with a greenish light I_s = (0.15, 0.7, 0.15)
- The reflected light is then given by (0.12, 0.14, 0.015), which is a fairly even mix of red and green, and would appear yellowish.
 - $-0.12 = 0.8 \times 0.15, 0.14 = 0.2 \times 0.7, 0.015 = 0.1 \times 0.15$



Example

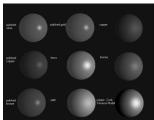
- Because specular light is mirror-like, the color of the specular component is often the same as that of the light source.
 - E.g., the specular highlight seen on a glossy red apple when illuminated by a yellow light is yellow rather than red.
- To create specular highlights for a plastic-like surface, set the specular reflection coefficients $\rho_{sr} = \rho_{sg} = \rho_{sb} = \rho_{s}$, so that the reflection coefficients are 'gray' in nature and do not alter the color of the incident light.
- The designer might choose $\rho_s = 0.5$ for a slightly shiny plastic surface, or $\rho_s = 0.9$ for a highly shiny surface.

Middle Tennessee State Universit



Combining Light Contributions and Adding Color (3)

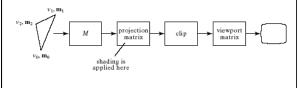
- A list of ρ and f values for various materials for ambient, diffuse, and specular light is given in Fig. 8.17.
- Spheres of different materials (mostly metallic, but one jade at bottom center) are shown at right (Fig. 8.18).





Shading and the Graphics Pipeline

 Shading is applied to a vertex at the point in the pipeline where the projection matrix is applied.
 We specify a normal and a position for each vertex.



Middle Tennessee State University



Shading and the Graphics Pipeline (2)

- glNormal3f (norm[i].x, norm[i].y, norm[i].z)
 specifies a normal for each vertex that follows it.
- The modelview matrix transforms both vertices and normals (m), the latter by M^{-T}m. M^{-T} is the transpose of the inverse matrix M⁻¹.
- The positions of lights are also transformed.



Shading and the Graphics Pipeline (3)

- Then a color is applied to each vertex, the perspective transformation is applied, and clipping is done.
- Clipping may create new vertices which need to have colors attached, usually by linear interpolation of initial vertex colors.
- If the new point a is 40% of the way from v_0 to v_1 , the color associated with a is a blend of 60% of (r_0, g_0, b_0) and 40% of (r_1, g_1, b_1) :

 color at point a = ($lerp(r_0, r_1, 0.4)$, $lerp(g_0, g_1, 0.4)$, $lerp(b_0, b_1, 0.4)$)

A STATUTE TO STATE OF THE STATE



Shading and the Graphics Pipeline (4)

- The vertices are finally passed through the viewport transformation where they are mapped into screen coordinates (along with pseudodepth, which now varies between 0 and 1).
- The quadrilateral is then rendered (with hidden surface removal).