Light Sources in OpenGL

• Light sources supported

- OpenGL supports *ambient light*, *point*, *spotlight*, and *distant* sources

- It supports up to 8 light sources in a program

• Enabling lighting and light sources

- To use lighting, you must first enable it by calling:

glEnable(GL_LIGHTING);

- You must also enable each light that you want using:

glEnable(source)

where source is:

GL_LIGHT0
GL_LIGHT1
...
GL_LIGHT7

- *Important:* when lighting is enabled, OpenGL ignores *glColor()* calls !!

• Point source

glLightf(source, parameter, value)

parameter:

GL_POSITION (*value*: location of the camera, default: 0.0, 0.0, 1.0, 0.0)
GL_AMBIENT (*value*: ambient components, default: 0.0, 0.0, 0.0, 1.0)
GL_DIFFUSE (*value*: diffuse components, default: 1.0, 1.0, 1.0, 1.0)
GL_SPECULAR (*value*: specular components, default: 1.0, 1.0, 1.0, 1.0)

- Example

```
GLfloat light_0_pos[]={1.0, 2.0, 3.0, 1.0};
GLfloat ambient_0_[]={1.0, 0.0, 0.0, 1.0};
GLfloat diffuse_0_[]={1.0, 0.0, 0.0, 1.0};
GLfloat specular_0_[]={1.0, 1.0, 1.0, 1.0};


glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
glLightfv(GL_LIGHT0, GL_POSITION, light_0_pos);
glLightfv(GL_LIGHT0, GL_AMBIENT, ambient_0);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse_0);
glLightfv(GL_LIGHT0, GL_SPECULAR, specular_0);
```

• **Distant source (W=0)**

GLfloat light_0_pos[]={1.0, 2.0, 3.0, 0.0};

• **Spotlight source**

glLightf(source, parameter, value)

parameter:

GL_SPOT_DIRECTION (*value:* spotlight direction, default: 0.0, 0.0, -1.0)

GL_SPOT_EXPONENT (*value:* exponent, default: 0.0)

GL_SPOT_CUTOFF (*value:* angle, default: 180.0)

• Changing the position of the light source

- Light sources are objects and are affected by model-view transformations

- The following three cases are possible:

  * Light sources remain stationary as objects move

  * Light sources that move while objects remain stationary

  * Light sources that move with the objects

• Distance term (accounts for light attenuation)

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

glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, a, default: 1.0)
glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, b, default: 0.0)
glLightf(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, c, default: 0.0)

• Global ambient term (independent of any light source)

GLfloat global_ambient[]={0.1, 0.1, 0.1, 1.0};

glLightModelv(GL_LIGHT_MODEL_AMBIENT, global_ambient);

Specification of materials in OpenGL

- The function that defines the effect of lighting on an object is:

glMaterialf(face, parameter, value)

where face:

GL_FRONT_AND_BACK, *GL_FRONT*, *GL_BACK*

and parameter:

GL_AMBIENT (*value:* ambient components, default: 0.2, 0.2, 0.2, 1.0)
GL_DIFFUSE (*value:* diffuse components, default: 0.8, 0.8, 0.8, 1.0)
GL_SPECULAR (*value:* specular components, default: 0.0, 0.0, 0.0, 1.0)
GL_SHINESS (*value:* specular component, default: 0.0)

(reflectivity coefficients: k_a, k_d, k_s)

- Example

```
GLfloat ambient[]={0.2, 0.2, 0.2, 1.0};
GLfloat diffuse[]={1.0, 0.8, 0.0, 1.0};
GLfloat specular[]={1.0, 1.0, 1.0, 1.0};

glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, ambient);
glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, diffuse);
glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, specular);
glMaterialfv(GL_FRONT_AND_BACK, GL_SHINESS, 100.0);
```

<u>Warning:</u> material properties remain the same until changed !!!

- Example

```
GLfloat off[] = {0.0, 0.0, 0.0, 0.0};
GLfloat white[] = {1.0, 1.0, 1.0, 1.0};
GLfloat red[] = {1.0, 0.0, 0.0, 1.0};
GLfloat deep_blue[] = {0.1, 0.5, 0.8, 1.0};
GLfloat shiny[] = {50.0};
GLfloat dull[] = {0.0};

/* Draw a small, dark blue sphere with shiny highlights */
glMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, deep_blue);
glMaterial(GL_FRONT, GL_SPECULAR, white);
glMaterial(GL_FRONT, GL_SHININESS, shiny);
glutSolidSphere(0.2, 10, 10);

/* Draw a large, red cube made of non-reflective material */
glMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, red);
glMaterial(GL_FRONT, GL_SPECULAR, off);
glMaterial(GL_FRONT, GL_SHININESS, dull);
glutSolidCube(10.0);

/* Draw a white, glowing sphere */
glMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, off);
glMaterial(GL_FRONT, GL_SPECULAR, off);
glMaterial(GL_FRONT, GL_SHININESS, dull);
glMaterial(GL_FRONT, GL_EMISSION, white);
glutSolidCube(10.0, 20, 20);
```

• Self-luminous surfaces

- For surfaces which look like emitting light (making a surface emmitive does not make it a light source !!)

GLfloat emission[]={0.0, 0.3, 0.3};
glMaterialfv(GL_FRONT_AND_BACK, GL_EMISSION, emission);