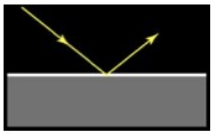
Lighting and Shading (OpenGL-oriented)

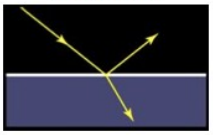
What you know so far (chapter 4)

Simple materials

metal

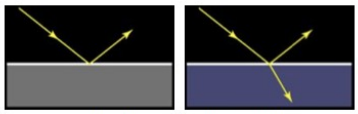


dielectric

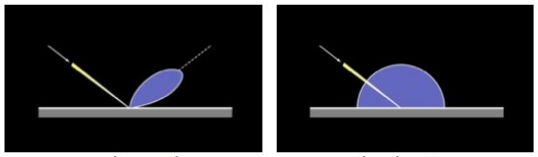


Smooth surfaces of pure materials have ideal specular reflection
Reflectance (fraction of light reflected) depends on angle)

Classic reflection behavior



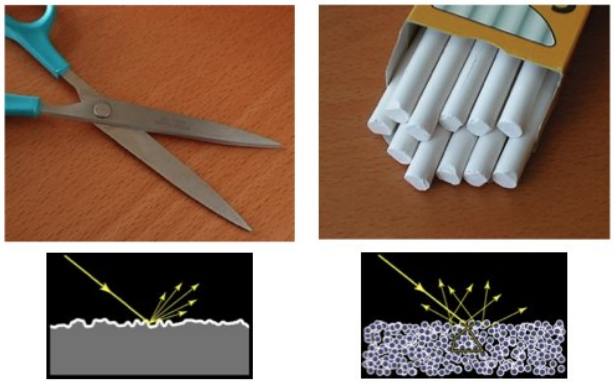
ideal specular (Fresnel)



rough specular

Lambertian

Adding microgeometry



OpengGL Implementaiton

- Example:
  - Siggraph_course_material under the resource page or download the directory on the main page which contains selected siggraph course material; look for the program called lightmaterial.c

Flat shading and perception

- Lateral inhibition: exaggerates perceived intensity

Figure 6.28 Step chart.

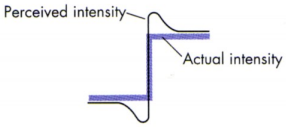
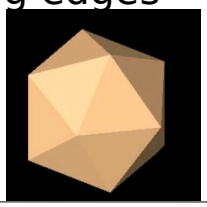


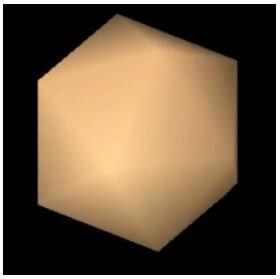
Figure 6.29 Perceived and actual intensities at an edge.

- Mach bands: perceived "stripes" along edges



Interpolative shading

- Enable with
  - glShadeModel(GL_SMOOTH)
- Calculate color at each vertex
- Interpolate color in interior
- Compute during scan conversion (rasterization)
- Much better image, more expensive to calculate



subdivisions

- Each time, multiply number of faces by 4.



Lighting in OpenGL

- Very similar to color
  - But different
- Enable lighting and lights
  - Lighting in general must be enabled
    - glEnable(GL_LIGHTING)
  - Each individual light must be enabled
    - glEnable(GL_LIGHT0);
  - OpenGL supports at least 8 light sources
    - Depending on graphics card

Global ambient light

- Set ambient intensity for entire scene
  - Glfloat al[]={0.2,0,2,0.2,0};

```
glLightModelfv(GL_LIGHT_MODEL_AM
   BIENT, al);
```

  - Properly light backs of polygons

```
glLightModeli(GL_LIGHT_MODEL_TWO
   _SIDED, GL_TRUE);
```


Defining a light source

- Use vectors {r, g, b, a} for light properties
- Beware: light source will be transformed!

```
GLfloat light_ambient[]={0.2, 0.2, 0.2, 1.0};
GLfloat light_diffuse[]={1.0, 1.0, 1.0, 1.0};
GLfloat light_specular[]={1.0, 1.0, 1.0, 1.0};
GLfloat light_position[]={-1.0, 1.0, -1.0, 0};
glLightfv(GL_LIGHT0, GL_AMBIENT,
   light_ambient);
glLightfv(GL_LIGHT0, GL_DIFFUSE,
   light_diffuse);
glLightfv(GL_LIGHT0, GL_SPECULAR,
   light_specular);
glLightfv(GL_LIGHT0, GL_POSITION,
   light_position);
```

Point source vs. directional source

- Directional light given by "position" vector

```
GLfloat light_position[] = {-1.0, 1.0, -1.0, 0.0};
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

- Point source given by "position" point

```
GLfloat light_position[] = {-1.0, 1.0, -1.0, 1.0};
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```


Spotlights

- Create point source as before
- Specify additional properties to create spotlight

```
GLfloat sd[] = {-1.0, -1.0, 0.0};
glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, sd);
glLightf (GL_LIGHT0, GL_SPOT_CUTOFF,   45.0);
glLightf (GL_LIGHT0, GL_SPOT_EXPONENT,  2.0);
```


Defining material properties

- Material properties stay in effect (like color)
- Set both specular coefficients and shiness

```
GLfloat mat_a[]  = {0.1, 0.5, 0.8, 1.0};
GLfloat mat_d[]  = {0.1, 0.5, 0.8, 1.0};
GLfloat mat_s[]  = {1.0, 1.0, 1.0, 1.0};
GLfloat low_sh[] = {5.0};
glMaterialfv(GL_FRONT, GL_AMBIENT,   mat_a);
glMaterialfv(GL_FRONT, GL_DIFFUSE,   mat_d);
glMaterialfv(GL_FRONT, GL_SPECULAR,  mat_s);
glMaterialfv(GL_FRONT, GL_SHININESS, low_sh);
```


Defining and maintaining normals

- Define unit normal before each **vertex**
  - glNormal3f(nx, ny, nz);
  - glVertex3f(x, y, z);
- Length changes under some transforms;
- Ask OpenGL to re-normalize
  - glEnable(GL_NORMALIZE);
  - glEnable(GL_RESCALE_NORMAL)

There are more in graphics...
(chapters 13, 20, 24, 25)

Motivations

Approximate physical reality

Radiosity: Restaurant interior: Guillermo Leal, Evolut

Lighting and Shading

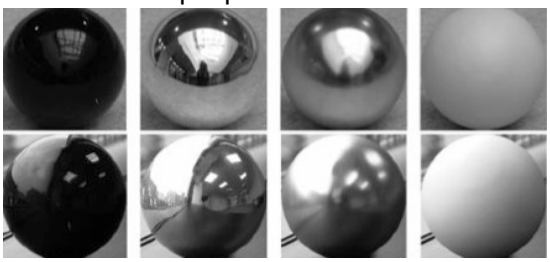
- Phong illumination model (last lecture)
  - Approximate only interaction light, surface, viewer
  - Relatively fast (online), supported in OpenGL
- Ray tracing
  - Follow light rays through a scene
  - Accurate, but expensive (off-line)
- Radiosity (advance CG)
  - Calculate surface inter-reflection approximately
  - Accurate, especially interiors, but expensive (off-line)

Raytracing example

Martin Moeck, Siemens Lighting

Shading

- Variation in observed color across an object
  - Strong affected by lighting
  - Present even for homogeneous material
- Caused by how a material reflects lights
  - Geometry
  - Light source, locations, and properties
  - Material properties



Shading for Computer Graphics

- Need to compute shading
  - Of particular geometry
  - Under particular illumination
  - From a particular viewpoint
- Basic question: how much light reflects from an object toward the viewer?

Inspirations...

- Cornell box images:
  - http://graphics.ucsd.edu/~henrik/images/cbox.html
- Ray-tracing competition
  - http://www.irtc.org/stills/