Rendering:
Lighting and Materials!
Where we are going!

- Rendering Models
- OpenGL HOWTO
  - Materials
- Lighting simulation
- OpenGL HOWTO
  - Texturing
Why does this milk look uncanny?
Materials!!

From Apodaca and Gritz, Advanced RenderMan
Types of Reflection

Mirror
- Ideal reflection
- Reflection Law

Diffuse
- Matte
- Lambert’s Law

Specular
- Highlights and gloss
- Microfacet model
Mirror: Ideal Specular Surface

Law of Reflection

\[ \theta_r = \theta_i \]
Diffuse: Lambert’s Law

\[ E = E_L \cos \theta_L = E_L \max(0, \mathbf{N} \cdot \mathbf{L}) \]

Lights only illuminate one side of the surface
Thus the \( \max(0,N \cdot L) \)
Isotropic vs. Anisotropic

Isotropic

Anisotropic

Sketch by Marc Levoy - cs248, 2008
Specular: Microfacets

Rough Surface

Minnaert p. 28

\[
\begin{align*}
\cos \theta_H &= \hat{N} \cdot \hat{H} \\
\hat{H} &= \frac{\hat{L} + \hat{E}}{|\hat{L} + \hat{E}|} \\
(\cos \theta_H)^s &= (\hat{N} \cdot \hat{H})^s
\end{align*}
\]
Shininess $s$

The larger the value $s$, the narrower the highlight

\[ s = 1 \quad \rightarrow \quad s = 64 \]

\[ \cos^s \theta \]
OpenGL Material Properties

- Ambient: \((Ka)\)
- Diffuse: \((Kd)\)
- Specular: \((Ks, s)\)

Final Image
Simple Materials & Lighting in OpenGL (demo)
Setup the lighting...

```c

//important!

GLfloat light0_position[] = { 1, 1, 1, 0 };
GLfloat light0_diffuse[] = { 1, 1, 1, 1 };
GLfloat light0_specular[] = { 1, 1, 1, 1 };

//if you set these to 0's you won't see diffuse color regardless of your material
//if you set these to 0's you won't see specular highlights regardless of your material

glEnable(GL_LIGHT0);
```

```c

//important!

GLfloat light0_position[] = { 1, 1, 1, 0 };
GLfloat light0_diffuse[] = { 1, 1, 1, 1 };
GLfloat light0_specular[] = { 1, 1, 1, 1 };

//if you set these to 0's you won't see diffuse color regardless of your material
//if you set these to 0's you won't see specular highlights regardless of your material

glEnable(GL_LIGHT0);
```
Setup the materials!

GLfloat mat_ambient[] = {1, .1, .8, 1 }
GLfloat mat_diffuse[] = { .2f, .2f, .6f, 1 }
GLfloat mat_specular[] = { .8f, .8f, .8f, 1 }
GLfloat mat_shininess[] = { 128 }; //range[0,128]

glMaterialfv( GL_FRONT, GL_AMBIENT, mat_ambient );
glMaterialfv( GL_FRONT, GL_DIFFUSE, mat_diffuse );
glMaterialfv( GL_FRONT, GL_SPECULAR, mat_specular );
glMaterialfv( GL_FRONT, GL_SHININESS, mat_shininess );
Changing your material colors

```c
glEnable(GL_COLOR_MATERIAL); //in your setup

//when you want to draw something with a new diffuse AND ambient color…
glColorMaterial(GL_FRONT, GL_AMBIENT_AND_DIFFUSE);
glColor4f(0,0,1,1);
drawSphere();

//now we just change the diffuse to red
.glColorMaterial(GL_FRONT, GL_DIFFUSE);
glColor4f(1,0,0,1);
drawSphere();
```
More Complex Materials!
Surface vs. Subsurface

BRDF

BSSRDF
Subsurface Scattering

\[
S(x_i, \omega_i, x_o, \omega_o) = \frac{dL(x_o, \omega_o)}{d\Phi(x_i, \omega_i)}
\]
Conveys Scale of model! Bonus 😊
Skin: Subsurface

Modeled by Stephen Stahlberg
Skin: Surface Only

Modeled by Stephen Stahlberg
Lighting
Lighting Simulation

How do we compute the illumination at each point on each surface, in a complex scene with different primitives, materials, and lighting properties? (“The Rendering Equation”)
Why this is hard:

- Primitives are complex:
  - lights, materials, shapes
- Infinite number of light paths!
Solution:

Ray tracing!

Albrecht Durer
(16^{th} c)
Point Light: Hard Shadows
Area Light: Now what?
Lighting: Soft Shadows

Soft Shadows
Area Light Source
Dealing with Reflection & Refraction
Indirect Illumination!
Complex Indirect Illumination

Mies Courtyard House with Curved Elements

Modeling: Stephen Duck; Rendering: Henrik Wann Jensen
Cornell Box Test

Measured

Simulated

Program of Computer Graphics
Cornell University
What you need to know about Texturing right now!

Models and texture maps from Poser, a product for creating and rendering human characters.
The Process:

Setup: After opengl is initialized…
1. Load image
2. Generate OpenGL texture

To Draw Textured Object:
1. Bind the texture
2. Draw the object
3. Unbind the texture

Important:
- Must have called glEnable(GL_NORMALIZE);
- You must use texture coords when drawing your object!
Simple Texturing Setup using OpenGL and Libst
Generating a texture with libst

duckPelt = new STTexture(new STImage("duckCM.jpg"));

Gives you this
Drawing a texture on a custom shape...

duckPelt->Bind();
glBegin(GL_POLYGON);

glTexCoord2f(0,1);
gINormal3f(0,0,1);
glVertex3f(0,0,-1);

glTexCoord2f(1,1);
gINormal3f(0,0,1);
glVertex3f(-1,0,1);

glTexCoord2f(1,0);
gINormal3f(0,0,1);
glVertex3f(1,0,1);

duckPelt->UnBind();
Using the OBJ

//in your setup
duck = new STShape("duck.obj");

...

//now to draw…
duckPelt->Bind();
duck->Draw();
duckPelt->UnBind();
Things to play around with 😊

glTexEnvf ( target, parameterToModify , parameter);

Example:
    glTexEnvf ( GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE )

glTexParameterf(target, parameterToModify , parameter);

Example:
    glTexParameterf( GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
    glTexParameterf( GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);

Look in the red book for cool things you can do with these functions!
Why does this milk look uncanny?
Answer: Needs subsurface scattering!
Answer: Needs cookies!
Things to Remember

Three major reflection models

- Mirror reflection
  - Angle of incidence equals angle of reflection
- Diffuse reflection
  - Reflection proportional to the energy falling on the surface
- Specular reflection
  - Microfacet models

Materials

- Combination of diffuse and specular
- Natural materials: e.g. marble, skin

Lighting and Ray Tracing

- Direct vs. indirect
- Point vs. area
- Shadows requires visibility query