CSE 167: Introduction to Computer Graphics Lecture #6: Illumination Model

Jürgen P. Schulze, Ph.D. University of California, San Diego Spring Quarter 2015

Announcements

- Project 3 due this Friday at Ipm
- Grading starts at 12:15 in CSE labs 260+270
- Next Thursday: Midterm
 - Midterm discussion on Monday at 4pm



Lecture Overview

- Depth Testing
- Illumination Model







• At each pixel, we need to determine which triangle is visible

			+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+			+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	4-	÷	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
. . .	t.	_±_	- 1	+.	.±.	. . .	t.	- ±-	. . .	t_	_ ±_	. ÷.	+_	_ ±_	. ÷.	t.	- ±-	+
+	+	+	+	+	+	+	+	+	+-	Ē		T	7	+	١C	+	÷	+
	マリン)				

Painter's Algorithm

- Paint from back to front
- Every new pixel always paints over previous pixel in frame buffer
- Need to sort geometry according to depth
- May need to split triangles if they intersect



 Outdated algorithm, created when memory was expensive



Z-Buffering

Store z-value for each pixel

Depth test

- During rasterization, compare stored value to new value
- Update pixel only if new value is smaller

```
setpixel(int x, int y, color c, float z)
if(z<zbuffer(x,y)) then
  zbuffer(x,y) = z
  color(x,y) = c</pre>
```

- z-buffer is dedicated memory reserved for GPU (graphics memory)
- Depth test is performed by GPU



Z-Buffering in OpenGL

In your application:

- > Ask for a depth buffer when you create your window.
- Place a call to glEnable (GL_DEPTH_TEST) in your program's initialization routine.
- Ensure that your zNear and zFar clipping planes are set correctly (in glOrtho, glFrustum or gluPerspective) and in a way that provides adequate depth buffer precision.
- Pass GL_DEPTH_BUFFER_BIT as a parameter to glClear.



Z-Buffering

Problem: translucent geometry

- Storage of multiple depth and color values per pixel (not practical in real-time graphics)
- Or back to front rendering of translucent geometry, after rendering opaque geometry
 - Does not always work correctly: programmer has to weight rendering correctness against computational effort





Lecture Overview

- Depth Testing
- Illumination Model



Shading

- Compute interaction of light with surfaces
- Requires simulation of physics
- "Global illumination"
 - Multiple bounces of light
 - Computationally expensive, minutes per image
 - Used in movies, architectural design, etc.



Global Illumination









Interactive Applications

- No physics-based simulation
- Simplified models
- Reproduce perceptually most important effects
- Local illumination
 - Only one bounce of light between light source and viewer





- Position object in 3D
- Determine colors of vertices
 - Per vertex shading
- Map triangles to 2D
- Draw triangles
 - Per pixel shading —



Lecture Overview

OpenGL's local shading model



- What gives a material its color?
- How is light reflected by a
 - Mirror
 - White sheet of paper
 - Blue sheet of paper
 - Glossy metal









Model reflection of light at surfaces

Assumption: no subsurface scattering

Bidirectional reflectance distribution function (BRDF)

- Given light direction, viewing direction, how much light is reflected towards the viewer
- For any pair of light/viewing directions!



Simplified model

- Sum of 3 components
- Covers a large class of real surfaces



Simplified model

- Sum of 3 components
- Covers a large class of real surfaces



- Ideal diffuse material reflects light equally in all directions
- View-independent
- Matte, not shiny materials
 - Paper
 - Unfinished wood
 - Unpolished stone









- Beam of parallel rays shining on a surface
 - Area covered by beam varies with the angle between the beam and the normal
 - The larger the area, the less incident light per area
 - Incident light per unit area is proportional to the cosine of the angle between the normal and the light rays
- Object darkens as normal turns away from light
- Lambert's cosine law (Johann Heinrich Lambert, 1760)
- Diffuse surfaces are also called Lambertian surfaces







Given

- Unit surface normal n
- Unit light direction L
- Material diffuse reflectance (material color) k_d
- Light color (intensity) c₁





Notes

- Parameters k_d , c_l are r,g,b vectors
- Need to compute r,g,b values of diffuse color c_d separately
- Parameters in this model have no precise physical meaning
 - c_i: strength, color of light source
 - k_d : fraction of reflected light, material color



- Provides visual cues
 - Surface curvature
 - Depth variation



Lambertian (diffuse) sphere under different lighting directions



OpenGL

- Lights (glLight*)
 - Values for light: $(0,0,0) \leq c_l \leq (1,1,1)$
 - Definition: (0,0,0) is black, (1,1,1) is white
- OpenGL
 - Values for diffuse reflection
 - Fraction of reflected light: $(0,0,0) \le k_d \le (1,1,1)$
- Consult OpenGL Programming Guide (Red Book)
 - See course web site



Simplified model

- Sum of 3 components
- Covers a large class of real surfaces





Specular Reflection

Shiny surfaces

- Polished metal
- Glossy car finish
- Plastics

Specular highlight

- Blurred reflection of the light source
- Position of highlight depends on viewing direction



Specular highlight



Specular Reflection

Ideal specular reflection is mirror reflection

- Perfectly smooth surface
- Incoming light ray is bounced in single direction
- Angle of incidence equals angle of reflection



Law of Reflection

Angle of incidence equals angle of reflection

$$\vec{\mathbf{R}} + \vec{\mathbf{L}} = 2\cos\theta \ \vec{\mathbf{n}} = 2(\vec{\mathbf{L}} \cdot \vec{\mathbf{n}})\vec{\mathbf{n}}$$
$$\vec{\mathbf{R}} = 2(\vec{\mathbf{L}} \cdot \vec{\mathbf{n}})\vec{\mathbf{n}} - \vec{\mathbf{L}}$$





Specular Reflection

Many materials are not perfect mirrors

Glossy materials





Glossy teapot



Glossy Materials

- Assume surface composed of small mirrors with random orientation (micro-facets)
- Smooth surfaces
 - Micro-facet normals close to surface normal
 - Sharp highlights
- Rough surfaces
 - Micro-facet normals vary strongly
 - Blurry highlight



Glossy Surfaces

- Expect most light to be reflected in mirror direction
- Because of micro-facets, some light is reflected slightly off ideal reflection direction
- Reflection
 - Brightest when view vector is aligned with reflection
 - Decreases as angle between view vector and reflection direction increases



Phong Shading Model

- Developed by Bui Tuong Phong in 1973
- Specular reflectance coefficient k_s
- Phong exponent p
 - Greater *p* means smaller (sharper) highlight





Phong Shading Model



₹UCSD

33

Blinn Shading Model (Jim Blinn, 1977)

Modification of Phong Shading Model

- > Defines unit halfway vector $\mathbf{h} = rac{\mathbf{L} + \mathbf{e}}{\|\mathbf{L} + \mathbf{e}\|}$
- Halfway vector represents normal of micro-facet that would lead to mirror reflection to the eye



Blinn Shading Model

- The larger the angle between micro-facet orientation and normal, the less likely
- Use cosine of angle between them
- Shininess parameter s
- Very similar to Phong Model



Simplified model

- Sum of 3 components
- Covers a large class of real surfaces



Ambient Light

- In real world, light is bounced all around scene
- Could use global illumination techniques to simulate
- Simple approximation
 - Add constant ambient light at each point: $k_a c_a$
 - Ambient light color: c_a
 - Ambient reflection coefficient: k_a
- Areas with no direct illumination are not completely dark



Complete Blinn-Phong Shading Model

- Blinn-Phong model with several light sources I
- All colors and reflection coefficients are vectors with 3 components for red, green, blue



