1

CS-184: Computer Graphics

Lecture #3: Shading

Prof. James O'Brien
University of California, Berkeley

2

Announcements

- **Assignment 1: due September 26**

3

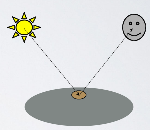
Today

- Local Illumination & Shading
  - The BRDF
  - Simple diffuse and specular approximations
  - Shading interpolation: flat, Gouraud, Phong
  - Some miscellaneous tricks

4

Local Shading

- Local: consider in isolation
  - 1 light
  - 1 surface
  - The viewer
- Recall: lighting is linear
  - Almost always...



Counter example: photochromatic materials

5

Local Shading

- Examples of non-local phenomena
  - Shadows
  - Reflections
  - Refraction
  - Indirect lighting

6

The BRDF

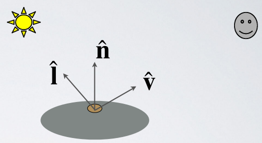
- The **B**i-directional **R**eflectance **D**istribution **F**unction
- Given
  - Surface material
  - Incoming light direction
  - Direction of viewer
  - Orientation of surface
- Return:
  - fraction of light that reaches the viewer
- We'll worry about physical units later...

$$\rho = \rho(\theta_V, \theta_L)$$
$$= \rho(\mathbf{v}, \mathbf{l}, \mathbf{n})$$

7

The BRDF

$\rho(\mathbf{v}, \mathbf{l}, \mathbf{n})$



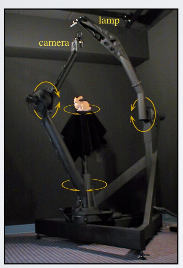
- Spatial variation capture by "the material"
- Frequency dependent
  - Typically use separate RGB functions
  - Does not work perfectly
  - Better:

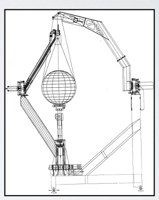
$$\rho = \rho(\theta_V, \theta_L, \lambda_{\text{in}}, \lambda_{\text{out}})$$

8

Obtaining BRDFs

- Measure from real materials





Images from Marc Levoy

9

Obtaining BRDFs

- Measure from real materials
- Computer simulation
  - Simple model + complex geometry
- Derive model by analysis
- Make something up

10

Beyond BRDFs

- The BRDF model does not capture everything
  - e.g. Subsurface scattering (BSSRDF)

Images from Jensen *et. al, SIGGRAPH 2001*

11

Beyond BRDFs

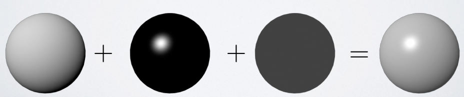
- The BRDF model does not capture everything
  - e.g. Inter-frequency interactions

$\rho = \rho(\theta_V, \theta_L, \lambda_{in}, \lambda_{out})$ This version would work....

12

A Simple Model

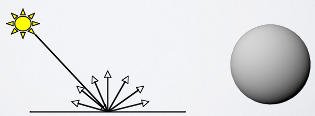
- Approximate BRDF as sum of
  - A diffuse component
  - A specular component
  - A "ambient" term



13

Diffuse Component

- Lambert's Law
  - Intensity of reflected light proportional to cosine of angle between surface and incoming light direction
  - Applies to "diffuse," "Lambertian," or "matte" surfaces
  - Independent of viewing angle
- Use as a component of non-Lambertian surfaces



14

Diffuse Component

Comment about two-side lighting in text is wrong...

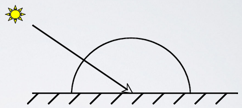
$$k_d I(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})$$

$$\max(k_d I(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}), 0)$$

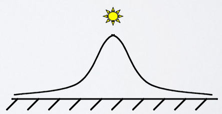
15

Diffuse Component

- Plot light leaving in a given direction:



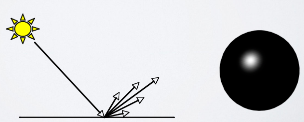
- Plot light leaving from each point on surface



16

Specular Component

- Specular component is a mirror-like reflection
- Phong Illumination Model
  - A reasonable approximation for some surfaces
  - Fairly cheap to compute
- Depends on view direction

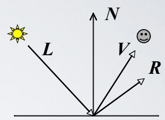


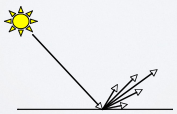
17

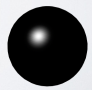
Specular Component

$k_s I(\hat{\mathbf{r}} \cdot \hat{\mathbf{v}})^p$

$k_s I \max(\hat{\mathbf{r}} \cdot \hat{\mathbf{v}}, 0)^p$





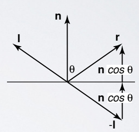


18

Specular Component

- Computing the reflected direction

$$\hat{\mathbf{r}} = -\hat{\mathbf{l}} + 2(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}})\hat{\mathbf{n}}$$

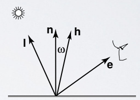


19

Specular Component

- "Half-angle" approximation for specular

$$\hat{\mathbf{h}} = \frac{\hat{\mathbf{l}} + \hat{\mathbf{v}}}{||\hat{\mathbf{l}} + \hat{\mathbf{v}}||}$$



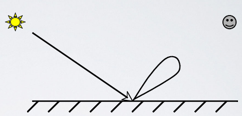
different specular term $k_s I (\hat{\mathbf{h}} \cdot \hat{\mathbf{n}})^p$

**Don't use half-angle approximation in your assignments!*

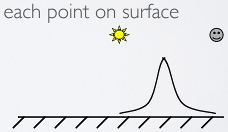
20

Specular Component

- Plot light leaving in a given direction:



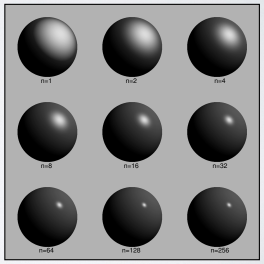
- Plot light leaving from each point on surface



21

Specular Component

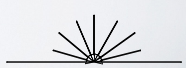
- Specular exponent sometimes called "roughness"



22

Ambient Term

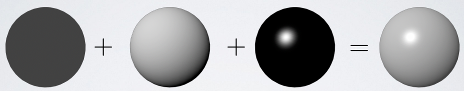
- Really, its a cheap hack
- Accounts for "ambient, omnidirectional light"
- Without it everything looks like it's in space



23

Summing the Parts

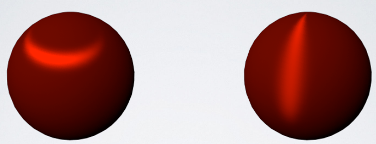
$$R = k_a I + k_d I \max(\hat{\mathbf{l}} \cdot \hat{\mathbf{n}}, 0) + k_s I \max(\hat{\mathbf{r}} \cdot \hat{\mathbf{v}}, 0)^p$$

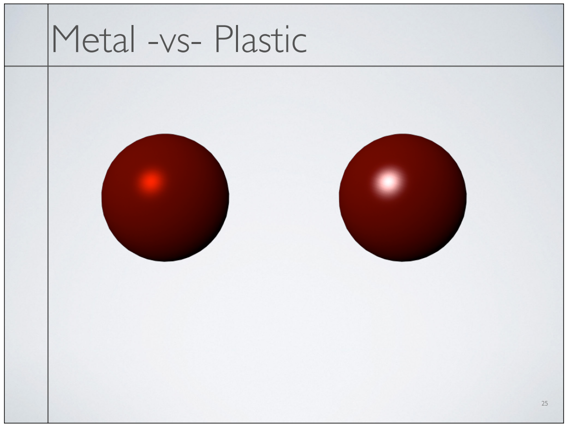


- Recall that the $k_?$ are by wavelength
  - RGB in practice
- Sum over all lights

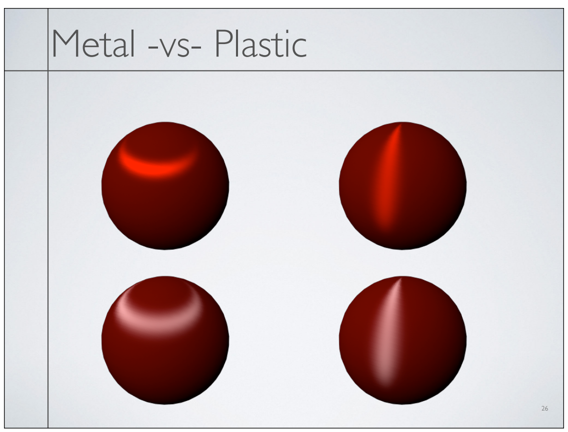
24

Anisotropy



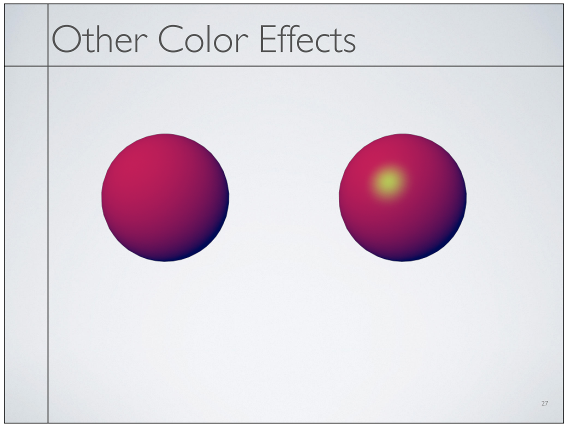


25

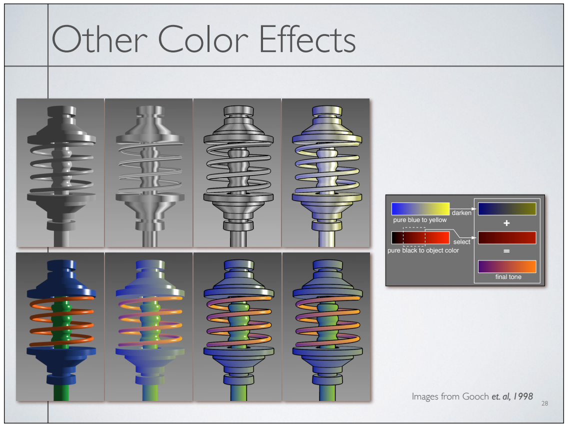


26

27

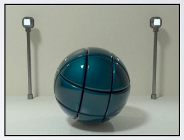

Other Color Effects
27

28


Other Color Effects
pure blue to yellow
darken
pure black to object color
select
final tone
Images from Gooch et. al, 1998
28

29

Measured BRDFs



BRDFs for automotive paint

Images from Cornell University Program of Computer Graphics

30

Measured BRDFs

BRDFs for aerosol spray paint

Images from Cornell University Program of Computer Graphics

31

Measured BRDFs

BRDFs for house paint

Images from Cornell University Program of Computer Graphics

31

32

Measured BRDFs

BRDFs for lucite sheet

Images from Cornell University Program of Computer Graphics

32

33

Details Beget Realism

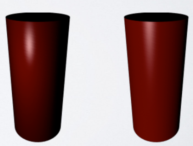
- The “computer generated” look is often due to a lack of fine/subtle details... a lack of richness.

From bustledress.com

34

Direction -vs- Point Lights

- For a point light, the light direction changes over the surface
- For “distant” light, the direction is constant
- Similar for orthographic/perspective viewer



35

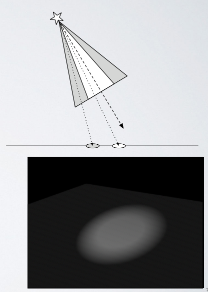
Falloff

- Physically correct: $1/r^2$ light intensify falloff
  - Tends to look bad (why?)
  - Not used in practice
- Sometimes compromise of $1/r$ used

36

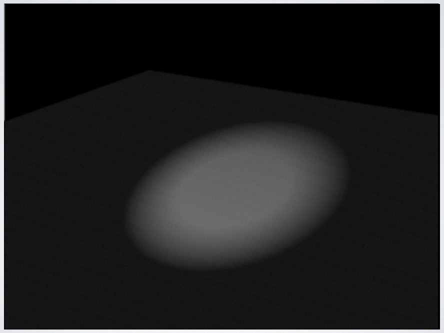
Spot and Other Lights

- Other calculations for useful effects
  - Spot light
  - Only light certain objects
  - Negative lights
  - *etc.*



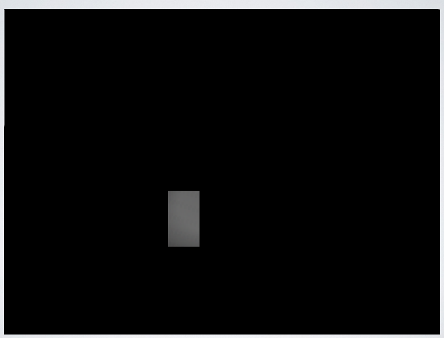
37

Ugly....



38

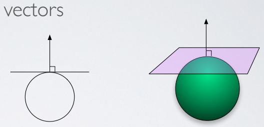
Ugly....



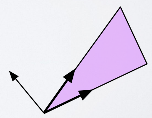
39

Surface Normals

- The normal vector at a point on a surface is perpendicular to all surface tangent vectors



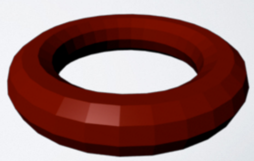
- For triangles normal given by right-handed cross product



40

Flat Shading

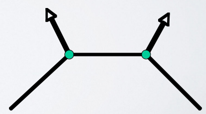
- Use constant normal for each triangle (polygon)
  - Polygon objects don't look smooth
  - Faceted appearance very noticeable, especially at specular highlights
  - Recall mach bands...



41

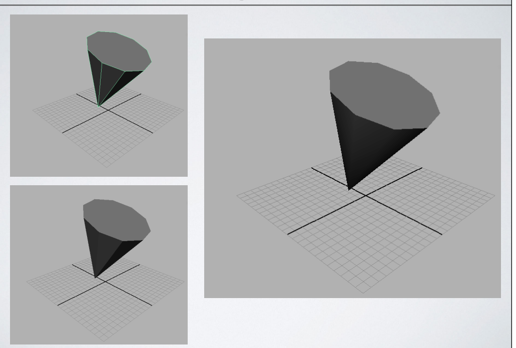
Smooth Shading

- Compute "average" normal at vertices
- Interpolate across polygons
- Use threshold for "sharp" edges
  - Vertex may have different normals for each face



42

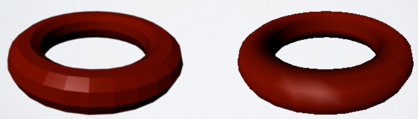
Smooth Shading



43

Gouraud Shading

- Compute shading at each vertex
  - Interpolate colors from vertices
  - Pros: fast and easy, looks smooth
  - Cons: terrible for specular reflections



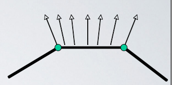
Flat Gouraud

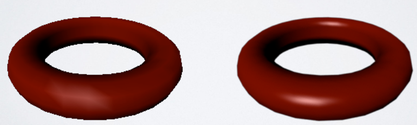
Note: Gouraud was hardware rendered...

44

Phong Shading

- Compute shading at each pixel
  - Interpolate ***normals*** from vertices
  - Pros: looks smooth, better speculars
  - Cons: expensive





Gouraud Phong

Note: Gouraud was hardware rendered...