

Shading

- ❖ To add realism
- ❖ Shading model
- ❖ Simplification of physical model



Interaction of Light and Surfaces

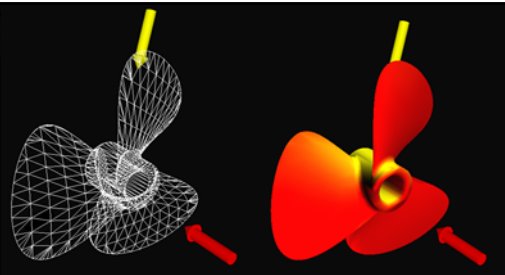

- ❖ The incident light interact with surfaces :
 - Some is absorbed
 - Some is reflected
 - Some is transmitted into interior



UNIVERSITY OF CALGARY

The Goal

- ❖ Input:
 - a 3D object
 - Material and color of the object
 - Position and structure of the light source
 - Value of intensity of the light source
- ❖ Output:
 - The intensity of points of the given object

Graphics 1
Paramaraj Samavathi

UNIVERSITY OF CALGARY

Reflected and Scattered Light

- ❖ Diffuse Scattering
 - Matt surfaces
- ❖ Specular reflection
 - Shiny plastic
 - Highlight
- ❖ Translucent
 - Glass, water
 - Penetrate the surface



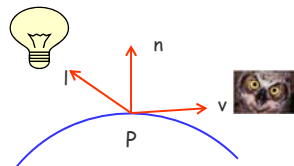
Graphics 1
Paramaraj Samavathi

Color Source

- ❖ Three component intensity (red, green, blue)
- ❖ Luminance of the source
- ❖ The red component of source \implies the red component of image
- ❖ The green component of source \implies the green component of image
- ❖ The blue component of source \implies the blue component of image
- ❖ Three similar but independent calculations
- ❖ We focus on one scalar value only

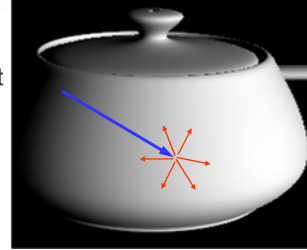
Three Important Vectors

- ❖ To compute the intensity at P , we need
 - The unit normal vector n
 - The unit vector v , from P to the viewer
 - The unit vector l , from P to the light



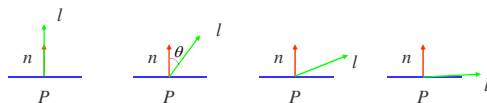
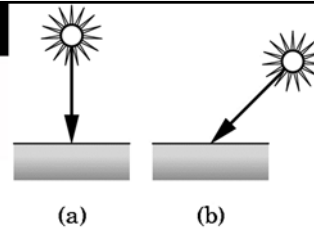
Diffuse Reflection

- ❖ A perfect diffuse reflector scatters the light equally in all directions
- ❖ Same appearance to all viewers
- ❖ Interact with the surface material
- ❖ Source of the color
- ❖ Important factors:
 - Material of the surface
 - The position of the light
- ❖ Rougher surface means higher diffusion
- ❖ Lambertian surface



Lambert's Law

- ❖ L_d : diffuse light from source
- ❖ I_d : diffuse reflection at P
- ❖ $I_d = K_d(l.n)L_d$
- ❖ $0 \leq K_d \leq 1$ is diffuse reflection coefficient
- ❖ K_d relates to the material of the surface



Diffuse Coefficient

❖ K_d is usually determined by a trial and error approach

❖ Examples:

Component	Gold	Black plastic	Silver
Red	0.75	0.01	0.5
Green	0.6	0.01	0.5
Blue	0.22	0.01	0.5

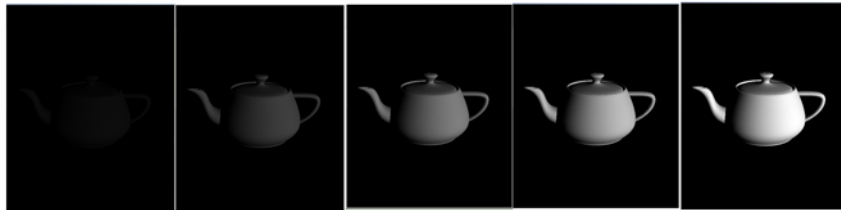
$K_d=0.05$

$K_d=0.25$

$K_d=0.5$

$K_d=0.75$

$K_d=1$



Specular Reflection

- ❖ Diffusion: no highlights, rough surface
- ❖ Specular: highlights, shiny and smooth surfaces
- ❖ View dependent reflection



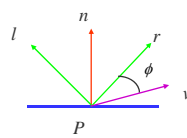
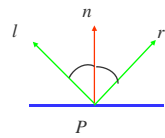
Mid

Be realistic
about yourself

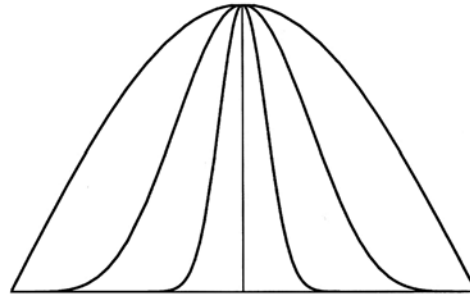


Phong Model for Specular Reflection

- ❖ r is the reflection of l about n
- ❖ $I_s = K_s L_s (r \cdot v)^\alpha$
- ❖ L_s : specular light from source
- ❖ α : shininess coefficient



The Shininess Coefficient



$\alpha = 1$

$\alpha = 2$

$\alpha = 4$

$\alpha = 6$



Summary

given :

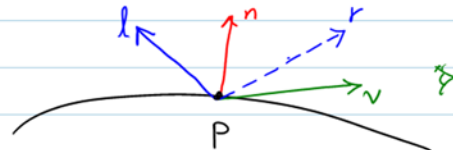
object/viewer information: P, l, n, v

light/material information: $L_d, L_s, k_d, k_s, \alpha, \dots$

$$I_d = k_d (l \cdot n) L_d$$

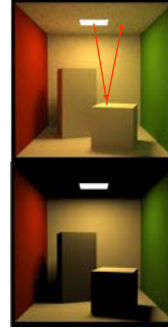
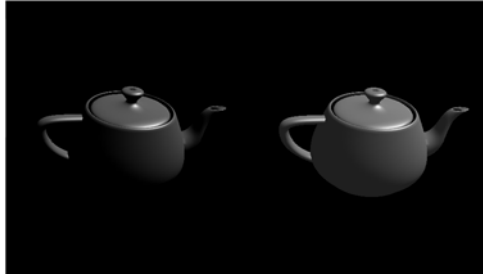
$$I_s = k_s (r \cdot v)^\alpha L_s$$

$$r = 2(n \cdot l)n - l$$



Ambient Light

- ❖ “Physical rules” are too simplified
- ❖ No indirect and global interaction of light



- ❖ To overcome the problem: use a back light called ambient light

Ambient Light Specification

- ❖ Not situated at any particular point
- ❖ Spreads in all directions uniformly
- ❖ $I_a = K_a L_a$
- ❖ L_a : the amount of ambient light in environment
- ❖ I_a : the amount of ambient light at each point
- ❖ $0 \leq K_a \leq 1$: ambient reflection coefficient

$K_a=0$

$K_a=0.5$

$K_a=1$



Combining all Lights Contributions: Phong reflection model

- ❖ The final model = diffuse + specular + ambient



- ❖ $I = K_d L_d (l \cdot n) + K_s L_s (r \cdot v)^\alpha + K_a L_a$

A complete example

I_r, I_g, I_b

- ❖ Brass ambient=(0.33,0.22,0.03)
- ❖ Brass diffuse=(0.78, 0.57,0.11)
- ❖ Brass specular=(0.99, 0.91,0.81)
- ❖ Brass_shininess = 27.8;
- ❖ Right Light=(1.0,0.0,0.0)
- ❖ Left Light=(1.0,1.0,1.0)



Incorporating a Distance Term

- ❖ $I(S, P) = \frac{I(P)}{d^2}$
- ❖ d : distance of S from P
- ❖ Exaggerated result
- ❖ More experimental parameters
- ❖ $I = \frac{1}{a+bd+cd^2}(K_d L_d(l.n) + K_s L_s(r.v)^\alpha) + K_a L_a$
- ❖ a, b, c are control parameters

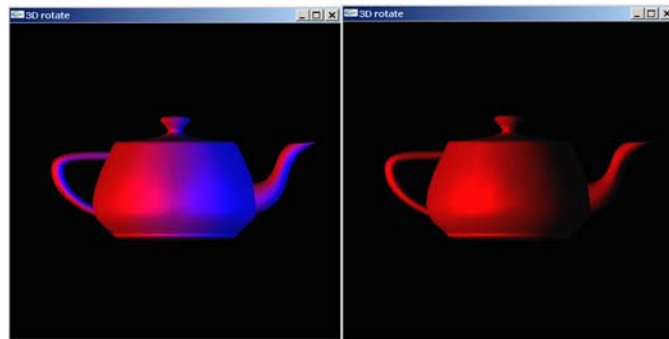
Two light sources

- ❖ Right Light=(1.0,0.0,0.0)
- ❖ Left Light=(1.0,1.0,1.0)
- ❖ OpenGL supports up to 8 light sources



Several Light Sources

- ❖ The total reflection at p is :
“sum of all contributed intensities from all sources”
- ❖ OpenGL allows us to define several light sources

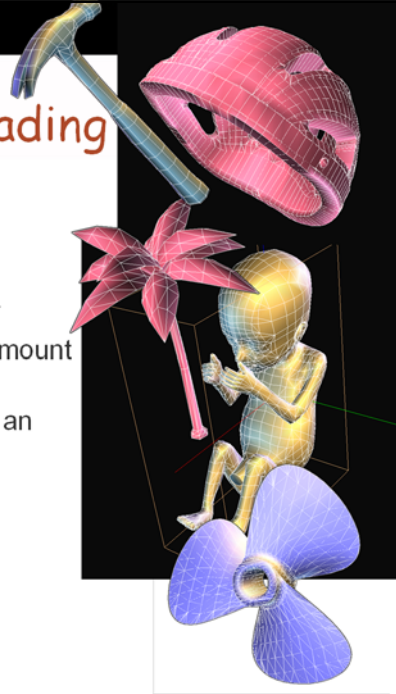


Polygon shading=Fill polygons



Polygonal mesh Shading

- for each face in the mesh
 - for each point on the face
 - *Find normal at this point
 - *Use Phong model to find the color
- ❖ These two steps can require large amount of computations
- ❖ Painting faces (polygon shading) as an alternative approach

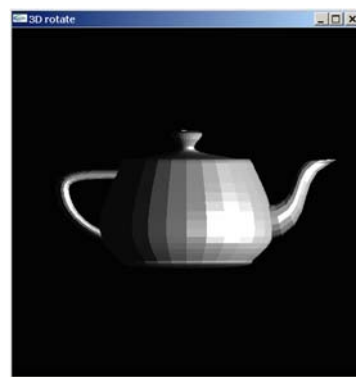
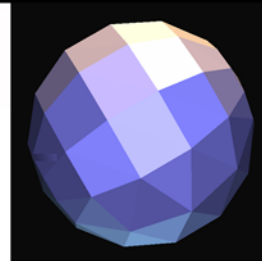


Graphics 1
Farmanuz Samavati



Flat Shading

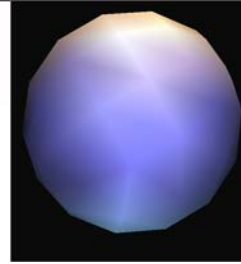
- ❖ Flat shading
 - o Individual faces are visualized
 - o Same color for any point of the face
 - o Suitable for a distant viewer and light sources
 - o OpenGL: `glShadeModel(GL_FLAT)`



Graphics 1
Farmanuz Samavati

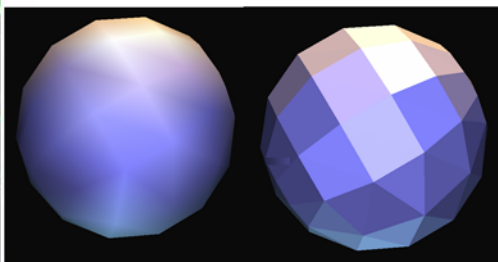
Smooth Shading

- ❖ Smooth shading
 - Visualize underlying surface
 - Each point of the face has its own color
 - Two techniques: Gouraud and Phong shading
 - OpenGL:
`glShadeModel(GL_SMOOTH)`



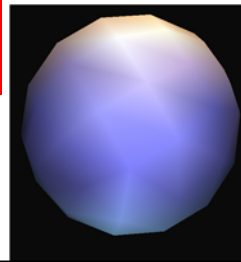
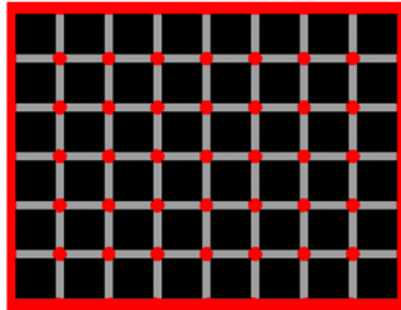
Flat versus Smooth shading

- ❖ Efficient
- ❖ Not suitable for smooth objects
- ❖ Specular highlights are rendered poorly





Does the human vision system always work properly ?

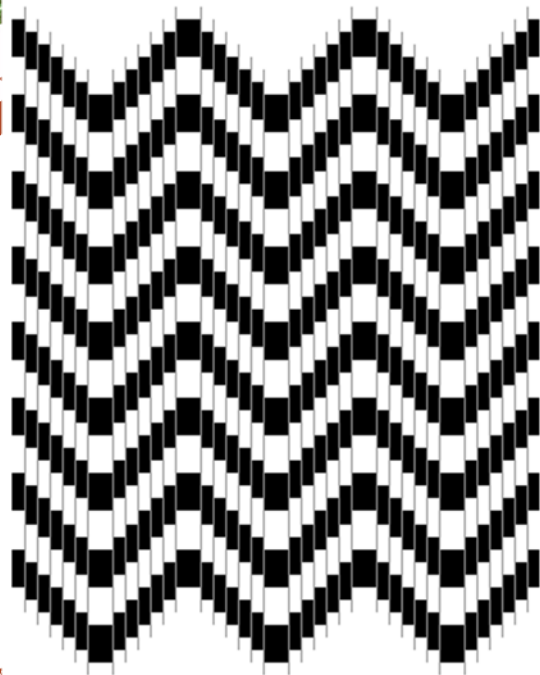


Graphics 1
Paramaraj Samayati



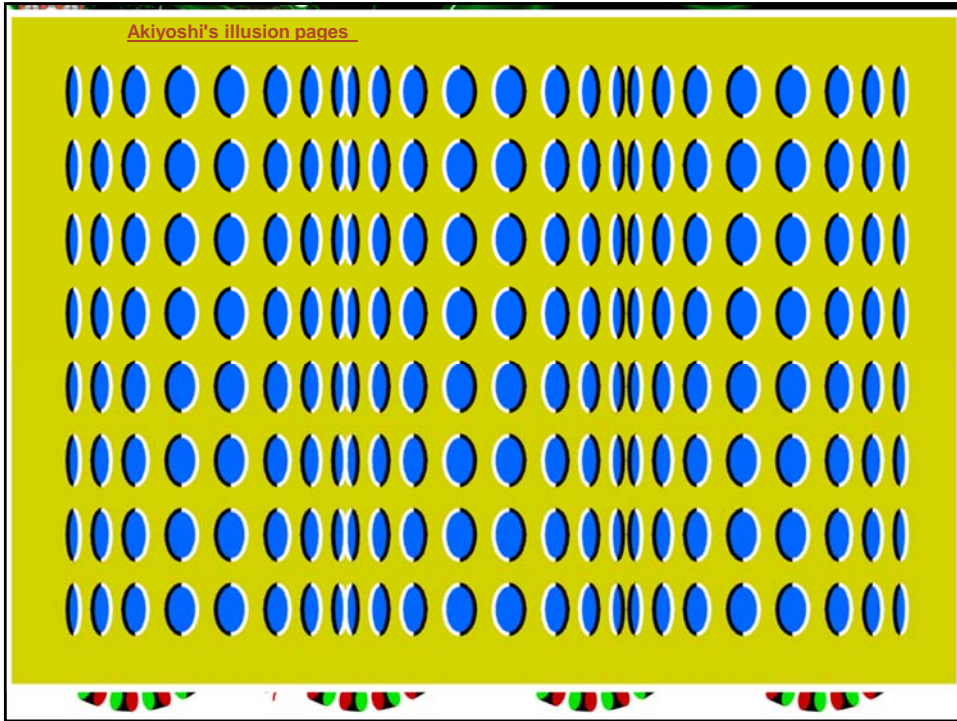
Doe
worl


lways



Graphics 1
Paramaraj Samayati

Ak

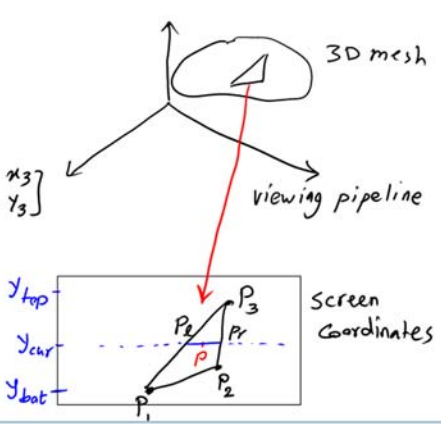


 UNIVERSITY OF CALGARY

Filling Polygons

How to assign color to all pixels inside of a given triangle $\Delta P_1 P_2 P_3$?

- Find screen coordinates
 $P_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$, $P_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$, $P_3 = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix}$
How?
- Fill the triangle using scan-line algorithm



3D mesh

viewing pipeline

Screen Coordinates

y_{top}
 y_{cur}
 y_{bot}

P_1 , P_2 , P_3

Graphics I
Faramarz Samavati

Polygon Fill

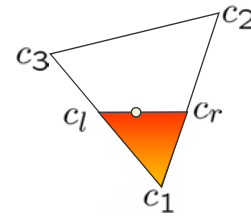
- ❖ For each face in the mesh


```
for (y=ybot; y<=ytop; y++){
  find xleft and xright
  for (x=xleft; x<= xright; x++){
    * find the color c for the pixel at (x,y)
    put c into the pixel at (x,y)
  }
}
```
- ❖ Flat shading:
 - Move (*) to the outside of loops
 - Efficient
 - Sometimes, it is not interesting

Screen space

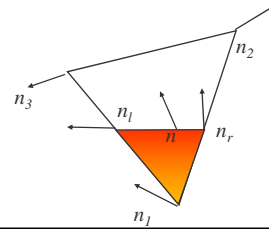
Smooth Shading (Gouraud)

- ❖ Gouraud shading: computes a different value of c
- ❖ Bilinear interpolations
- ❖
- ❖ $c = (1 - \alpha)c_l + \alpha c_r$
- ❖ $c_l = (1 - \beta)c_1 + \beta c_3$
- ❖ $c_r = (1 - \gamma)c_1 + \gamma c_2$
- $\alpha, \beta, \gamma?$
- ❖ More expensive than flat shading



Phong Shading

- ❖ Better realism for highlights
- ❖ Use normal of vertices to interpolate normal of interior points
- ❖ Linear interpolation of n_l and n_r $\Rightarrow \hat{n}_l$
- ❖ Linear interpolation of n_l and n_2 $\Rightarrow \hat{n}_r$
- ❖ Linear interpolation of \hat{n}_l and \hat{n}_r $\Rightarrow n$
- ❖ Normalize n
- ❖ Drawback: relatively slow

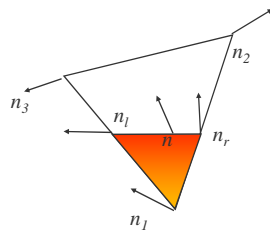


Clarification

- ❖ Phong reflexion model means:

$$I = K_d L_d (l \cdot n) + K_s L_s (r \cdot v)^\alpha + K_a L_a$$

- ❖ While Phong shading means:



Compare

