Illumination Models for Graphics

CS 211A
Can be very complex

- The incoming light can come from a source, or bouncing off another object, or after multiple bounces
- Sources can be extended
- Multiple interactions between light and surface
Very simple models

- Assumes point light source
- Models only the direct illumination from the source
  - Does not consider light reaching after bouncing off other objects
- Illumination models evaluated only at the vertices
For every vertex

- Ambient component
- Diffused component
- Specular component
Ambient Component

- Equal amount of light from all directions
- Approximates the indirect illumination
- $I_a k_a$
  - $I_a$ = intensity of ambient light
  - $k_a$ = percentage of the light reflected by the object
Lighting at a point on surface

- $I = I_d k_d \cos \theta$
  - $I_d = \text{intensity of light}$
  - $k_d = \text{coefficient of diffuse reflection}$
- $I = I_d k_d (N \cdot L)$
Diffused Component

- \( I = I_d k_d (N \cdot L) \)
- \( I_d (N \cdot L) \) is like the irradiance
- \( k_d \) is like the reflectivity
- No dependency on viewer
  - View independent
Ambient and Diffused Lighting

\[ I = I_p k_d (N.L) \]

\[ I = (I_a k_a + I_p k_d (N.L)) \]

Figure 10. Diffuse reflection for \( k_d = 0.4, 0.55, 0.7, 0.85, 1.0 \).
(© [AW94] Figure 14.03)

Figure 11. Ambient and diffuse reflection with \( k_d = 0.4 \) and \( k_a = 0.0, 0.15, 0.3, 0.45, 0.6 \).
(© [AW94] Figure 14.04)
Specular Component

- $I_s k_s \cos^n(\alpha)$
- $\cos(\alpha)$: fall off as $V$ moves away from $R$
- $n$ gives the sharpness
Specular Component

- $R$ depends on $L$
- Depends on both $V$ and $L$
- Like the BRDF
- $n$ controls the view-dependency also
Providing Control

• Providing enough control so that one can simulate effect via trial and error of many different parameters
• May be not be close to the physical phenomenon
• For e.g. Different brightness of the same light can be used for different component computation
Attenuation Control

- Diffused component
- \( I = I_d f_{att} k_d(N.L) \)
  \[-f_{att} = 1/(a+bd+cd^2)\]
  - \( d \) = distance of light from the surface
  - \( a, b \) and \( c \) are user defined constants
Attenuation of Light

Increasing distance from the light source

- a=0, b=0, c=1
- a=0.25, b=0.25, c=0.5
- a=0, b=1, c=0
Other issues

- \((I_a k_a + I_d k_d (N.L) + I_s k_s \cos^n(\alpha))\)
- For different channels
  - Do the same operation for all channels
- Multiple lights
  - Only one ambient light source
  - Multiple point light sources
    - Addition of light from different light sources
Ambient
Ambient + Diffuse
Ambient + Diffuse + Specular
What is Shading?

- Illumination model
- How do we use these models to **shade** the triangles in the graphics pipeline?
- How did we generate the picture on the right?
Method

• Evaluate illumination model at the vertices of the triangles
  – After model-view transformation

• Use interpolation to color the interior of the triangles during rasterization
  – Different shading methods use different interpolation

• Assume that the polygonal models approximate smooth surfaces
Normal Computation

- Normal of a triangle
  - $N = (B-A) \times (C-A)$
- Vertices are in anticlockwise direction with respect to normal

- Normal of a vertex
  - Average of all the triangle incident on the vertex
  - $N_v = (N_1 + N_2 + N_3 + N_4)/4$
Constant/Flat/Faceted Shading

• Illumination model applied once per triangle
• Using normal of the triangle
• Shade the whole triangle uniformly
  – Color associated with triangles and not vertices
Gouraud Shading

- Interpolating illumination between vertices
  - Calculate the illumination using vertex normals at vertices
  - Bilinear interpolation across the triangle
Gouraud Shading

- Edges get same color, irrespective of which triangle they are rendered from
  - Shading is continuous at edges
- Tends to spread sharp illumination spots over the triangle
Phong Shading

- Interpolate the normal across the triangle
- Calculate the illumination at every pixel during rasterization
  - Using the interpolated normal
- Slower than Gouraud
- Does not miss specular highlights
  - Good for shiny specular objects
Gouraud vs. Phong Shading

Gouraud

Phong

Spreads highlights across the triangle

Gouraud

Phong

Misses a highlight completely
Flat Shading
Gouraud Shading
Phong Shading
Shading

- Independent of the Illumination model used
- Phong Shading and Phong Illumination
- Artifacts
  - Piecewise planar approximation
  - Screen Space Interpolation
- Simple and hence widely used
Artifacts: Mach Bands

At discontinuities

Actual Intensity

Perceived Intensity
Artifacts: Mach Bands

- Common in flat shading since shading is discontinuous at edges
- Also present in Gouraud shading
  - Gradient of the shading may change suddenly
- Phong shading reduces it significantly
  - But cannot be eliminated
  - At sharp changes in surface gradient
Artifacts: Screen Space Interpolation

- Shading is interpolated while rasterization
- \( S_p = (S_1 + S_2)/2 \)
- \( z_s \neq (z_1 + z_2)/2 \)
Artifacts: T-junctions

- The shading at the T-junction are different when calculated from different triangles
- Shading discontinuity
Artifacts: Vertex Normals

- Vertex normal does not reflect the curvature of the surface adequately
  - Appear less flat than it actually is