Computer Graphics 5 - Illumination and Shading

Tom Thorne

Slides courtesy of Taku Komura
www.inf.ed.ac.uk/teaching/courses/cg
Overview

Lighting

- Phong illumination model

Shading

- Flat shading
- Gouraud shading
- Phong shading
The eye works like a camera

- Sensors at the back of eye
- Sense the amount of light coming from different directions
- Similar to CMOS and CCDs
Light coming into the eye

- Position of the point the eye is looking at
- Position of the light source
- Colour and intensity of light
- Vector from eye to point
- Normal vector of surface at point
- Physical properties of the object
Phong illumination model

A simple 3 parameter model comprised of 3 illumination terms

- **Diffuse**: non-shiny illumination and shadows
- **Specular**: shiny reflections
- **Ambient**: background illumination

\[ \text{Diffuse} + \text{Specular} + \text{Ambient} = \]
Diffuse (Lambertian) reflection

When light hits an object with a rough surface, it is reflected in all directions

- Amount of light hitting the surface depends on the angle between the normal vector and the incident vector of the incoming light.
- The larger the angle (up to 90 degrees), the larger the area the incident light is spread over.
Diffuse reflection

\[ I = I_p k_d \cos \theta \]

- \( I_p \): Light intensity
- \( \theta \): Angle between normal vector and direction to light source
- \( k_d \): Diffuse reflectivity

Note that there is no dependence on the angle between the direction to the camera and the surface normal.
Specular reflection

- Direct reflections of the light source off of a shiny surface
- Smooth surfaces
Specular reflection

\[ l = l_p k_s \cos^n \alpha \]

- \( l_p \) Light intensity
- \( \alpha \) Angle between reflection vector and direction to camera
- \( k_s \) Specular reflectivity
- \( n \) Specular intensity
Specular reflection

\[ R = 2\mathbf{N}(\mathbf{N} \cdot \mathbf{L}) - \mathbf{L} \]
Specular reflection

- Specular light with different $n$ values

$n=4$  $n=100$
Combining diffuse and specular reflection
What is missing?

- Only points on the surface that are directly lit by the light are illuminated
Ambient lighting

➤ Light reflected or scattered from other objects in the scene
➤ Environmental light
➤ Precise simulation of this is very hard!
Ambient lighting

Very simple approximation:
\[ I = k_a l_a \]
Combined lighting models

Combining ambient, diffuse and specular highlights gives the Phong Illumination model

\[ I = I_a k_a + I_p \left( k_d \cos \theta + k_s \cos^n \alpha \right) \]

Ambient + Diffuse + Specular = I
Multiple light sources

- For multiple light sources we simply compute the illumination from each source and sum them.

\[ I = I_a k_a + \sum_{i=1}^{m} I_i (k_d \cos \theta + k_s \cos^n \alpha) \]
Using dot products

\[ I = I_a k_a + \sum_{l=1}^{m} I_l (k_d (N \cdot L_l) + k_s (V \cdot R_l)^n) \]

- \( V \) Vector from the surface to the viewer
- \( N \) Normal vector at point on surface
- \( R \) Reflection vector
- \( L \) Vector from surface to light source

\( \overrightarrow{L} \) means the vector \( L \) normalized to be of unit length.
\[
I^R = I^R_a k_a^R + \sum_{p=1}^{P} I^R_p (k^R_d (\overrightarrow{N} \cdot \overrightarrow{L}) + k^R_s (\overrightarrow{V} \cdot \overrightarrow{R}))
\]

\[
I^G = I^G_a k_a^G + \sum_{p=1}^{P} I^G_p (k^G_d (\overrightarrow{N} \cdot \overrightarrow{L}) + k^G_s (\overrightarrow{V} \cdot \overrightarrow{R}))
\]

\[
I^B = I^B_a k_a^B + \sum_{p=1}^{P} I^B_p (k^B_d (\overrightarrow{N} \cdot \overrightarrow{L}) + k^B_s (\overrightarrow{V} \cdot \overrightarrow{R}))
\]
Local illumination model

This model considers only light sources and the properties of surfaces. We don’t consider light reflected from other surfaces.

- Real time rendering
- Cost depends on number of light sources
Problems

Certain things cannot easily be rendered with this model:

- Brushed metal
- Marble (subsurface scattering)
- Colour bleeding
Overview

Lighting

- Phong illumination model

Shading

- Flat shading
- Gouraud shading
- Phong shading
How do we colour the surface?

We know how to colour single points on the surface, but how do we colour the whole object?

- Shading
- Performed during rasterisation
Shading models

- Flat shading (one lighting calculation per polygon)
- Gouraud shading (one lighting calculation per vertex)
- Phong shading (one calculation per pixel)
Flat shading

- Colour is computed once for each polygon
- All pixels in a polygon are set to the same colour
- Works for objects made of flat faces
Flat shading

Suffers from an effect called Mach banding

- Eyes are sensitive to sudden changes in brightness
- Artificial changes in brightness are introduced on either side of the boundary
Mach band

An optical illusion, discovered by Ernst Mach
Gouraud shading
Gouraud shading

- Colour is computed once per vertex using the local illumination model
- Polygons interpolate colours over their surface
Computing vertex normals

Vertex normals are found by averaging the face normals:

\[
\overline{N}_V = \frac{\sum_{i=1}^{n} \overline{N}_i}{\|\sum_{i=1}^{n} \overline{N}_i\|}
\]
Computing vertex normals

Transforming vertex normals, we need to take care to ensure they remain perpendicular to the surface.

Tangent vectors remain tangent to the surface after transformation. For any tangent vector \( t \):

\[
\begin{align*}
    n \cdot t &= 0 \\
    n^T (M^{-1} M) t &= 0 \\
    (n^T M^{-1})(M t) &= 0
\end{align*}
\]

Therefore \( n^T M^{-1} \) is perpendicular to the transformed \( Mt \) for any \( t \). So the desired normal vector is:

\[
    n' = (M^{-1})^T n
\]
Problems with Gouraud shading

In specular reflection the highlight can be sharp, depending on the shape of \( \cos^n \alpha \)

- Gouraud shading interpolates linearly and so can make the highlight much bigger
- Gouraud shading can miss highlights that occur in the middle of a polygon
Problems with Gouraud shading
Phong shading
Phong shading

- Lighting computation is performed at each pixel
- Normal vectors are interpolated over the polygon
Phong shading

Able to produce highlights that occur in the middle of a polygon
Phong example
Problems with interpolation shading

Vertex normals can be incorrect when calculated as average of face normals

Solutions:

- Add more polygons
- Test for angles and use different vertex normals for adjacent polygons
Problems with interpolation shading

Vertices not shared by all polygons

B is not a vertex of the large polygon. Shading calculated at vertex B won’t necessarily be the same as the interpolated calculations made when shading the large polygon.
Summary

Illumination

- Phong Illumination model combining ambient, diffuse and specular lighting

Shading

- Gouraud shading
- Phong shading
References

- Foley, Chapter 16 (Illumination and shading), up to 16.3
- Shirley, Chapter 10 (Surface shading)