Ray Casting

• Simplest shading approach is to perform independent lighting calculation for every pixel

\[ I = I_E + K_A I_{AL} + \sum_i (K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i) \]
Polygon Rendering Methods

• Given a freeform surface, one usually approximates the surface as a polyhedra.
• How do we calculate in practice the illumination at each point on the surface?
• Applying the illumination model at each surface point is computationally expensive.
Polygon Shading

- Can take advantage of spatial coherence
  - Illumination calculations for pixels covered by same primitive are related to each other

\[ I = I_E + K_A I_AL + \sum_i (K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i) \]
Piecewise linear approximation
Polygonal Approximation
Smooth Shading
Polygon Shading Algorithms

Wireframe

Flat

Gouraud

Phong
Flat Shading

What if a faceted object is illuminated only by directional light sources and is either diffuse or viewed from infinitely far away?

One illumination calculation per polygon
Assign all pixels inside each polygon the same color
Flat Shading

- A fast and simple method.
- Gives reasonable result only if all of the following assumptions are valid:
  - The object is really a polyhedron.
  - Light source is far away from the surface so that $N \cdot L$ is constant over each polygon.
  - Viewing position is far away from the surface so that $V \cdot R$ is constant over each polygon.
Flat Shading

Objects look like they are composed of polygons
OK for polyhedral objects
Not so good for ones with smooth surfaces
Gouraud Shading

- Produces smoothly shaded polygonal mesh
  - Piecewise linear approximation
  - Need fine mesh to capture subtle lighting effects
Polygon Smooth Shading
Gouraud Shading

- What if smooth surface is represented by polygonal mesh with a normal at each vertex?

\[
I = I_E + K_A I_{AL} + \sum_i \left( K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i \right)
\]
Gouraud Shading

• Smooth shading over adjacent polygons
  – Curved surfaces
• Renders the polygon surface by linearly interpolating intensity values across the surface.

Mesh with shared normals at vertices
Gouraud Shading

- One lighting calculation per vertex
  - Assign pixels inside polygon by interpolating colors computed at vertices
Gouraud Shading

1. Determine the average unit normal at each polygon vertex.
2. Apply an illumination model to each vertex to calculate the vertex intensity.
3. Linearly interpolate the vertex intensities over the surface polygon.
The normal vector at a vertex

The normal $N_v$ of a vertex is an average of all neighboring normals:

$$N_v = \frac{\sum_k N_k}{|\sum_k N_k|}$$

Which is simply the following normalized vector:

$$N_v = \sum_k N_k$$
Bilinear by three linear interpolations

Two linear interpolations along the y-axis, and one along the x-axis.
Linear Interpolation

\[
I = w_2 I_a + w_1 I_b
\]

\[
I = w_2 I_a + (1 - w_2) I_b
\]

\[
I = w_2 (I_a - I_b) + I_b
\]
Bilinear Interpolation

- Bilinearly interpolate colors at vertices down and across scan lines

\[ A = \alpha l_1 + (1-\alpha) l_3 \]
\[ B = \beta l_2 + (1-\beta) l_3 \]
\[ I = \phi A + (1-\phi) B \]
Bilinear Interpolation

- \( I_a = \frac{(Y_s - Y_2)}{(Y_1 - Y_2)} \times I_1 + \frac{(Y_1 - Y_s)}{(Y_1 - Y_2)} \times I_2 \)
- \( I_b = \frac{(Y_s - Y_3)}{(Y_1 - Y_3)} \times I_1 + \frac{(Y_1 - Y_s)}{(Y_1 - Y_3)} \times I_3 \)
- \( I_p = \frac{(X_b - X_p)}{(X_b - X_a)} \times I_a + \frac{(X_p - X_a)}{(X_b - X_a)} \times I_b \)
Gouraud Shading of a sphere
Phong Shading

A more accurate method for rendering a polygon surface is to interpolate normal vectors, and then apply the illumination model to each surface point.
Flat

Gouraud

Phong
Phong Shading

1. Determine the average unit normal at each polygon vertex.
2. Linearly interpolate the vertex normals over the surface polygon.
3. Apply the illumination model along each scan line to calculate pixel intensities for each surface point.
Phong Shading

- What if polygonal mesh is too coarse to capture illumination effects in polygon interiors?

\[ I = I_E + K_A I_{AL} + \sum_i (K_D (N \cdot L_i) I_i + K_S (V \cdot R_i)^n I_i) \]
Phong Shading

One lighting calculation per pixel;
Approximate surface normals for points inside polygons
by bilinear interpolation of normals from vertices
Phong Shading

- Bilinearly interpolate surface normals at vertices down and across scan lines

\[ A = \alpha N_1 + (1-\alpha)N_3 \]

\[ B = \beta N_2 + (1-\beta)N_3 \]

\[ I = \phi A + (1-\phi)B \]
Flat Shading

Gouraud Shading

Phong Shading
Diffuse surface

With additional specular component
Polygon Shading Algorithms

Wireframe

Flat

Gouraud

Phong
Shading Issues

- Problems with interpolated shading:
  - Polygonal silhouettes
  - Perspective distortion
  - Orientation dependence (due to bilinear interpolation)
  - Problems at T-vertices
  - Problems computing shared vertex normals
One shade or color for the entire object, e.g., there really is no shading being done.

A Pixar Shutterbug example image with faceted shading.
A Pixar Shutterbug example image with faceted shading.

A Pixar Shutterbug example image with Gouraud shading and no specular highlights.
A Pixar Shutterbug example image with Gouraud shading and no specular highlights.

A Pixar Shutterbug example image with Gouraud shading and specular highlights.
Summary

• 2D polygon scan conversion with a sweep-line algorithm
  – Flat
  – Gouraud
  – Phong

Less expensive
More accurate
Global Illumination

In the real world light is everywhere.

• Reflects in every direction from every surface onto every surface.

• Anywhere in the world, light comes from infinite directions around.

In the lighting equation we used the Ambient intensity to approximate this.
Ambient Occlusion

- Full GI still too expensive for full feature film.
- Ambient Occlusion is used in most modern films to simulate indirect lighting in an overcast day.
- Usually, rendered separately and ‘baked’ as texture or 3D data that modifies values of direct lighting.
AO - advantages

- Much cheaper than GI.
- Usually does not depend on lighting, looks ok with most light settings.
- Can be computed once for each scene and reused for every frame.
Three Point Lighting

• Basic and commonly used lighting technique
  1. Key light
  2. Fill light
  3. Back light
Key light

- Creates the subject's main illumination, and defines the most visible lighting and shadows.
- Simulates main source of illumination
Fill light

- Softens and extends the illumination, simulates secondary light sources
- At most, half as bright as your key light,
- usually, casts no shadow
Back light

- creates a "defining edge" to help visually separate the subject from the background