Polygon Rendering Methods

Application of an illumination model to the rendering of the standard graphics objects those formed with polygon surfaces

The objects are usually polygon mesh approximation of curved surface objects but they may also be polyhedra that are not curved surface approximations

Scan line algorithms typically apply a lighting model to obtain polygon surface rendering in one or two ways each polygon can be rendered with a single intensity or the intensity can be obtained at each point of the surface using an interpolation scheme

A. <u>Constant Intensity Shading</u>
Fast and simple method for rendering of an object with polygon surfaces in CIS also called flat shading

Single intensity calculated for each polygon and useful for quickly displaying the general appearance of curved surface

This method is accurate if

a. **the object is a polyhedron and is not an approximation of an object with a curved surface**
b. **all light sources illuminating the object are sufficiently far from the surface**
c. **the viewing position is sufficiently far from the surface so that V.R is constant over the surface**

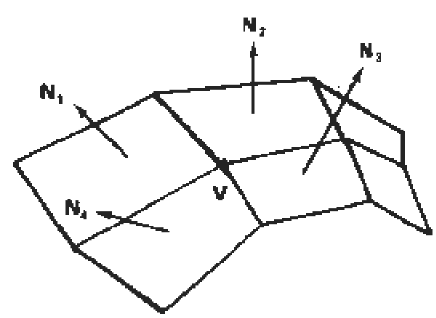
Even if all conditions are not true , we can still reasonable approximate surface lighting effects using small polygon facets with flat shading and calculate the intensity for each facet at the center of the polygon.

B. <u>Gouraud Shading</u>
This intensity interpolation scheme developed by Gouraud renders a polygon surface by linearly interpolating intensity values across the surface

Intensity values for each polygon are matched with the values of the adjacent polygon along the common edge thus eliminating the intensity discontinuities occur in “flat shading”

Calculation for each polygon surfaces

a. **Determine the average unit normal vector at each polygon vertex.**
b. **Apply an illumination model to each vertex to calculate the vertex intensity**
c. **Linearly interpolate the vertex intensities over the surface of the polygon**



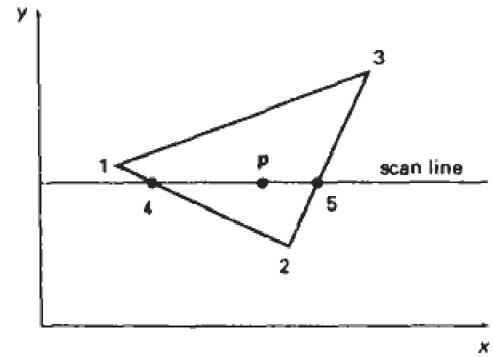
N_1 normal to ABCD plane , N_2 normal to CDEF plane and so on .

For any vertex position V normal unit vector

$$\mathbf{N}_v = \frac{\sum_{k=1}^{n} \mathbf{N}_k}{\left|\sum_{k=1}^{n} \mathbf{N}_k\right|}$$

once N_v is known intensity at vertices can be obtained from lighting model

Next step: Interpolating intensities along polygon edges
fast method to find intensity at 4 using 1 and 2 using only vertical displacement



$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} \cdot I_1 + \frac{y_1 - y_4}{y_1 - y_2} \cdot I_2$$

Similar process for I_5, using 3 and 2

For interior point p interpolated from the bounding intensities at point 4 & 5

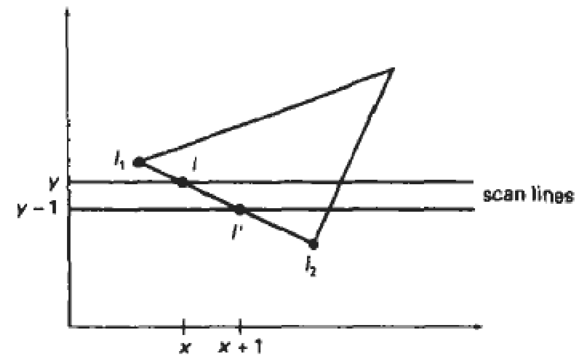
$$I_p = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x5 - x_4} I_5$$

Easier than this is incremental calculations for successive edge intensity values

$$I = \frac{y - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y}{y_1 - y_2} I_2$$

for next scan line y - 1

$$I' = I + \frac{I_2 - I_1}{y_1 - y_2}$$



Similar calculation to obtain intensities at successive horizontal pixel positions along each scan line

For color, intensity of each color component is calculated

Gouraud shading can be combined with a hidden surface algorithm to fill in the visible polygon

Advantages:

removes discontinuities of intensity at the edge compared to constant shading model

Disadvantages:

highlights on the surface are sometimes displayed with anomalous shapes and linear intensity interpolation can cause bright or dark intensity streak s called Mach Bands to appear on the surfaces.

Mach bands can be reduced by dividing the surface into a greater number of polygon faces or Phong shading(requires more calculation)

<u>Phong Shading</u>

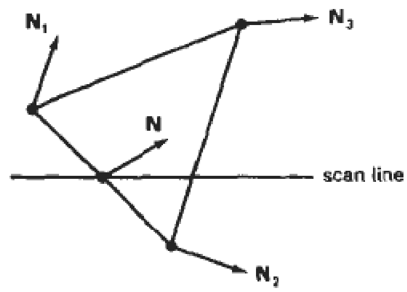
More accurate method for rendering a polygon surface is to interpolate normal vector and then apply the illumination model to each surface point called "Phong Shading" or "Normal Vector Interpolation Shading".

It displays more realistic highlights on a surface and greatly reduces Mach band effect.
Steps

a. **Determine the average unit vector normal at each polygon vertex**
b. **Linearly interpolate the vertex normals over polygon surface**
c. **Apply an illumination model along each scan line to calculate projected pixel intensities for the surface points**

N can be obtained by vertically interpolating between edge end point normals (N1 and N2)

$$N = \frac{y - y_2}{y_1 - y_2} \cdot N_1 + \frac{y_1 - y}{y_1 - y_2} \cdot N_2$$



Incremental methods are used to evaluate normals between scan lines and along each individual scan line (as in Gouraud) at each pixel position along a scan line the illumination model is applied to determine the surface intensity at that point

Produces accurate results than the direct interpolation but it requires considerable more calculations

Fast Phong Shading (FPS)

FPS approximates the intensity calculations using a Taylor series expansion and triangular surface patches

Surface normal at any point (x,y) over a triangle as $N = Ax + By + C$

A,B,C are determined from three vertex equations $N_k = Ax_k + By_k + C$ **….** $k = 1,2,3$ **(x_k, y_k vertex position)**
Omitting reflexivity and attenuation parameters

$$I_{diff}(x,y) = \frac{L \cdot N}{|L| \cdot |N|} \qquad \frac{L \cdot (Ax + By + C)}{|L| \cdot |Ax + By + C|} \qquad \frac{(L \cdot A)x + (L \cdot B)y + L \cdot C}{|L| \; |Ax + By + C|}$$

We can write

$$I_{diff}(x,y) = \frac{ax + by + c}{(dx^2 + exy + fy^2 + gx + hy + i)^{1/2}} \qquad \text{……………………..(i)}$$

Where a,b,c,d are used to represent the various dot products eg $a = \frac{L \cdot A}{|L|}$

Finally denominator in eq(i) can be expressed as Taylor series expansion and retain terms up to second degree in x and y. This yields

$$I_{diff}(x,y) = T_5x^2 + T_4xy + T_3y^2 + T_2x + T_1y + T_0 \qquad \text{……………………..(ii)}$$

Where each T_k is a function of parameters a ,b ,c and so forth

Using forward difference we can evaluate (ii) with only two additions for each pixel position (x,y) once the initial forward different parameter have been evaluated

FPS is two times slower than Gouraoud shading, Normal Phong shading is 7 times slower than Gouraud

FPS can be extended to include specular reflections, FPS algorithms can be generalized to include polygons other than triangles and infinite viewing positions.