$$I = I_a k_a O_d + \Sigma\ f_{att} I_p [k_d O_d (\overline{N} \bullet \overline{L}) + k_s O_s (\overline{R} \bullet \overline{V})^n]$$

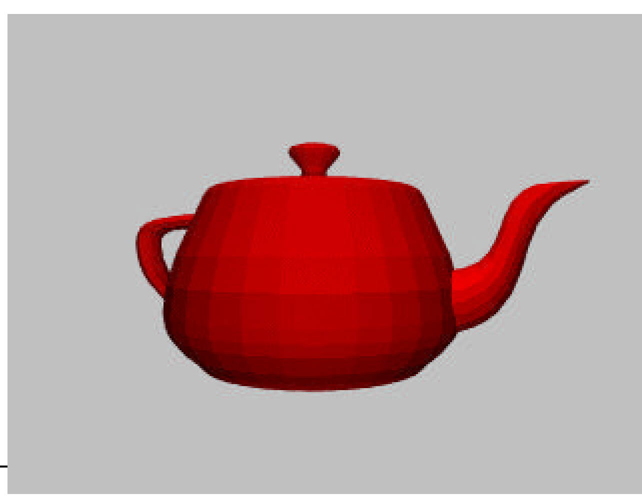
Shading Models

- the process of applying the illumination model to various surface points is called shading.
- shading can be a costly process involving the computation of and normalizing of vectors to light sources and the viewer.
- the brute-force method of doing this for each point on the surface is VERY expensive!

Constant shading:

- the simplest shading method applies only one illumination calculation for each primitive. This technique is called constant or flat shading. It is most often used on polygonal primitives.
- e.g. the illumination model is applied once to determine a single intensity value which is then used to shade the entire polygon
- This approach has several assumptions:
  1. the light source is at infinity so $N \bullet L$ is constant across the polygon face
  2. the viewer is at infinity so $N \bullet V$ is constant across the polygon face
  3. the polygon represents the actual surface being modeled and is not an approximation to a curved surface

If either of the first two assumptions are wrong, then if we are to use constant shading, we need some method to determine a single value for each of L and V (e.g. center of the polygon or the first vertex).

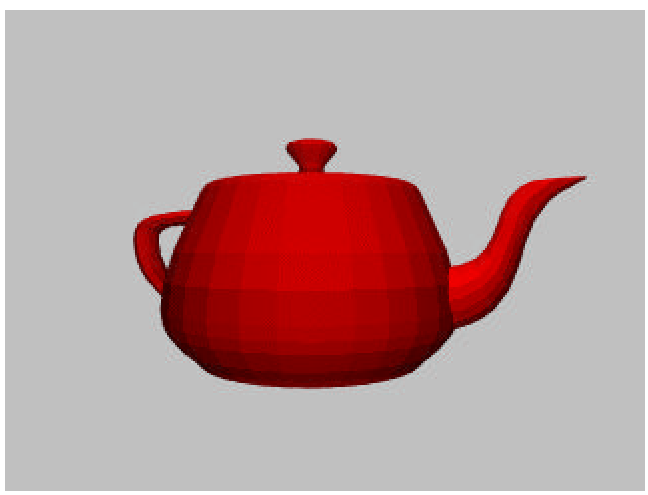


Interpolated Shading:

- shading information is linearly interpolated across a polygon from values determined for its vertices
- approximates evaluating the illumination model at each point on the polygon
- problem, if the polygon DOES NOT represent the actual surface being modeled (is an approximation to a curved surface like a polygon-mesh)

Polygon-mesh

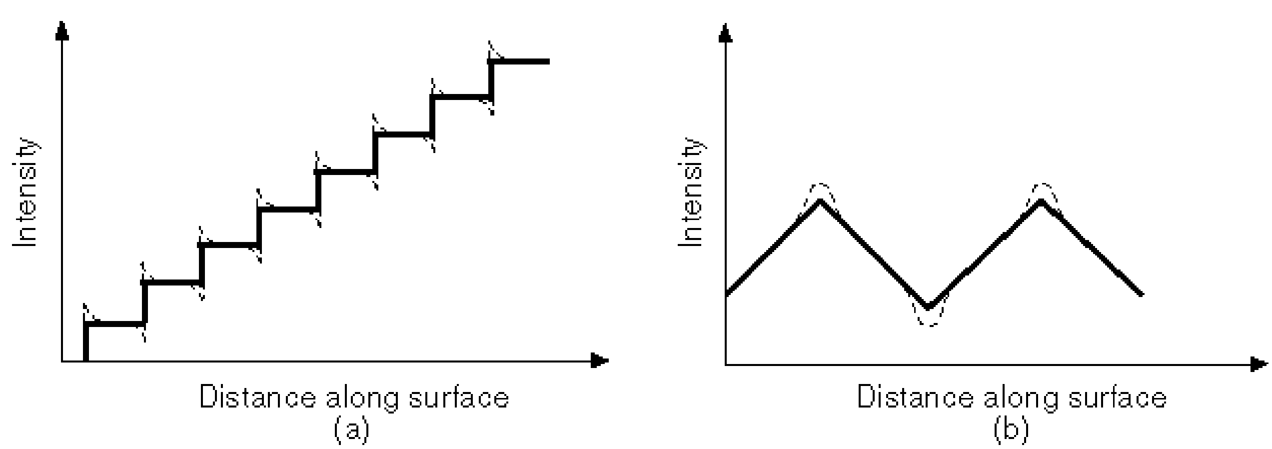
- if each polygonal facet in the mesh is shaded individually, it is easily distinguished from neighbors who orientation is difference, producing a “faceted” appearance.



- This is true for constant shading, interpolated shading or even per-pixel illumination calculations because the two adjacent polygons of different orientation have different intensities along their borders

- can we use a finer mesh???

- Mach band effect
  - perceived differences in shading between adjacent facets is exaggerated
  - at the border between two facets, the dark facet looks darker and the light facet looks lighter
  - caused by lateral inhibition of the receptors of the eye
  - the more light a receptor receives, the more that receptor inhibits the response of the receptors adjacent to it



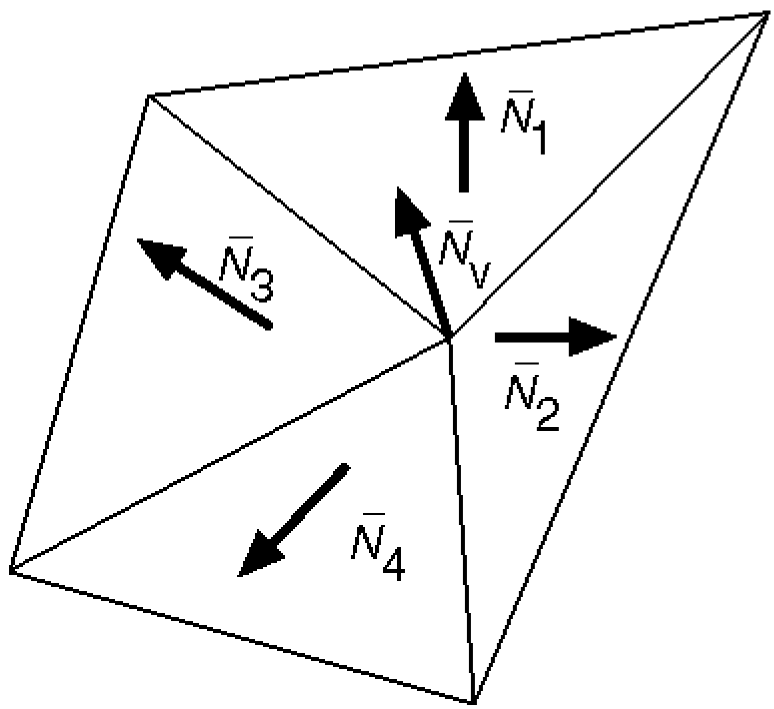
- to help counteract this, two basic shading models take advantage of the imformation provided by adjacent polygons

Gouraud Shading

- intensity interpolation shading

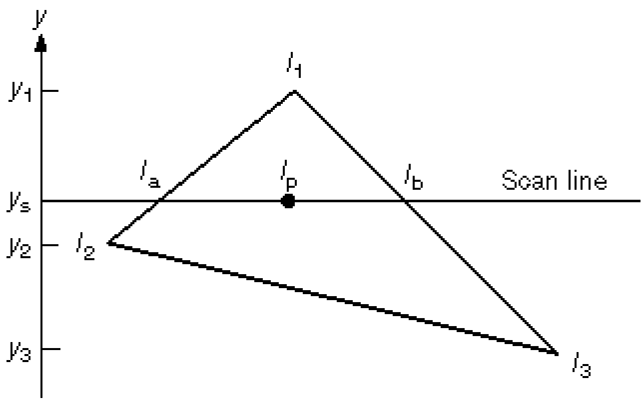
- extends the concept of interpolated shading by interpolating polygon vertex illumination values
- If you are fortunate you will be given a surface normal for each vertex but often vertex normals are unavailable, in this case we approximate a vertex normal by averaging the normals of the adjacent faces.

$$\bar{n}_v = \sum_{i=1}^{k} \frac{\bar{n}_i}{|\bar{n}_i|}$$



- the next step is to find vertex intensities by using the vertex normals with any desired illumination model

- finally, each polygon is shaded by linear interpolation of vertex intensities along each edge and then between edges along each scan line



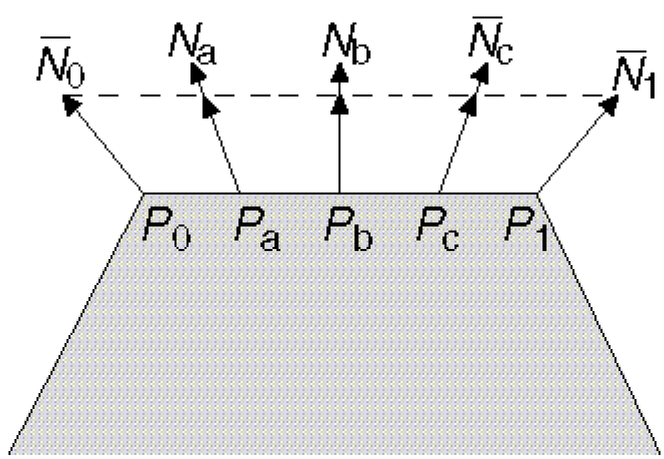
$$I_a = I_1 - (I_1 - I_2)\frac{y_1 - y_s}{y_1 - y_2}$$

$$I_b = I_1 - (I_1 - I_3)\frac{y_1 - y_s}{y_1 - y_3}$$

$$I_p = I_b - (I_b - I_a)\frac{x_b - x_p}{x_b - x_a}$$

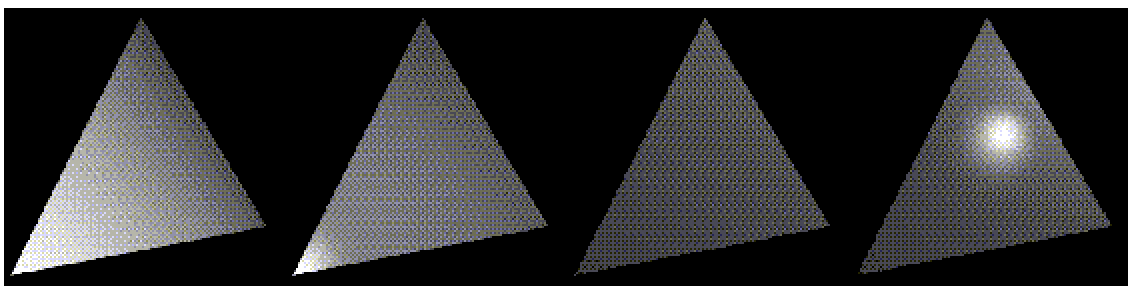
Pong Shading

- normal vector interpolation shading
- interpolates the surface normal vector $\overline{N}$, rather than the intensity.
- interpolation occurs across a polygon span on a scan line, between starting and ending normals for the span
- these normals are themselves interpolated along polygon edges from vertex normals
- at each pixel along a scan line, the interpolated normal is normalized and a new intensity calculation is performed using any illumination model



- Pong shading gives substantial improvements over Gouraud shading when specular-reflectance is used because highlights are more accurately represented.

- If Gouraud shading is used, then the intensity across the polygon is linearly interpolated between the highlight intensity and the lower intensities of the adjacent vertices, spreading the highlight over the polygon
- Also, if the highlight isn't at a vertex, then Gouraud shading may miss the highlight all together since no interior point can be brighter than the brightest vertes from which it is interpolated



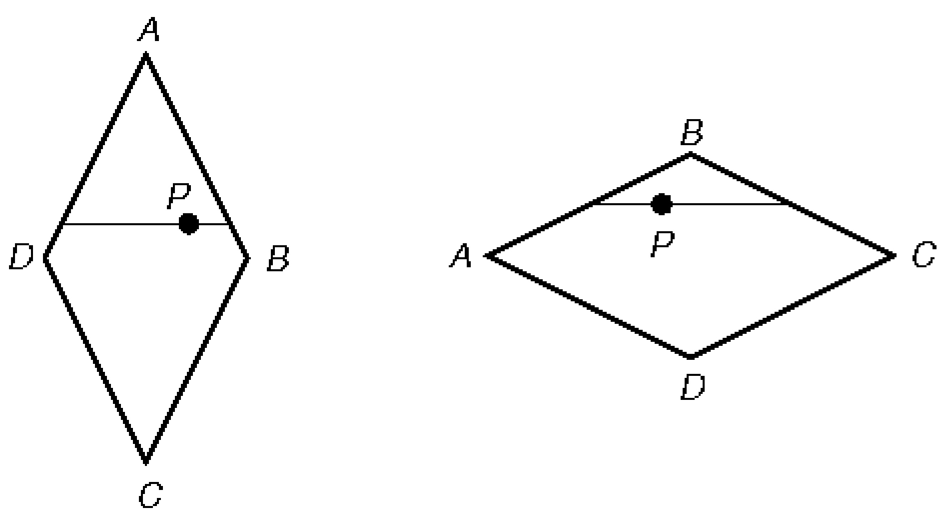
- In general, Pong shading gives superior results but greatly increases the cost of shading since the interpolated normal must be normalized every time it is used in an illumination model

Problems with interpolated shading:

- Polygonal silhouette
  - the silhouette edge of a mesh is still clearly polygonal. We can improve this by breaking the surface into smaller polygons but at an increase in expense

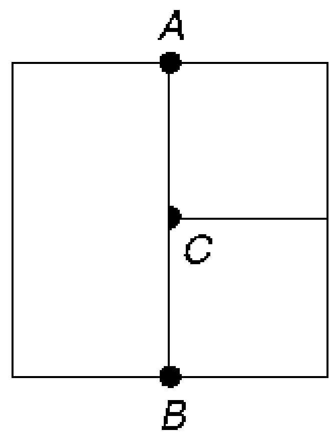
- Perspective distortion
  - Anomalies are introduced if interpolation is performed after perspective transformation in the 3D screen-coordinates system rather than in the WC system

- Orientation dependence
  - The interpolation is orientation dependent because values are interpolated between vertices and across horizontal scan lines



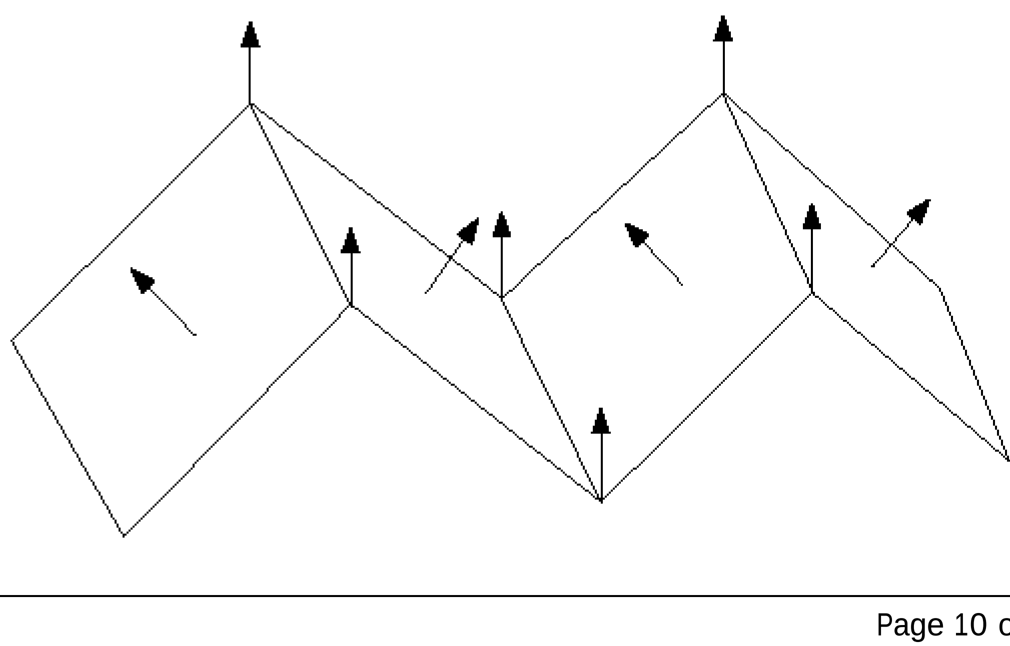
(a) (b)

- Problems at shared vertices
  - When two adjacent polygons fail to share a vertex that lies along their common edge



  - The eliminate this discontinuity we can insert an extra vertex

- Underrepresented vertex normals
  - Computed vertex normals may not represent the surface geometry adequately

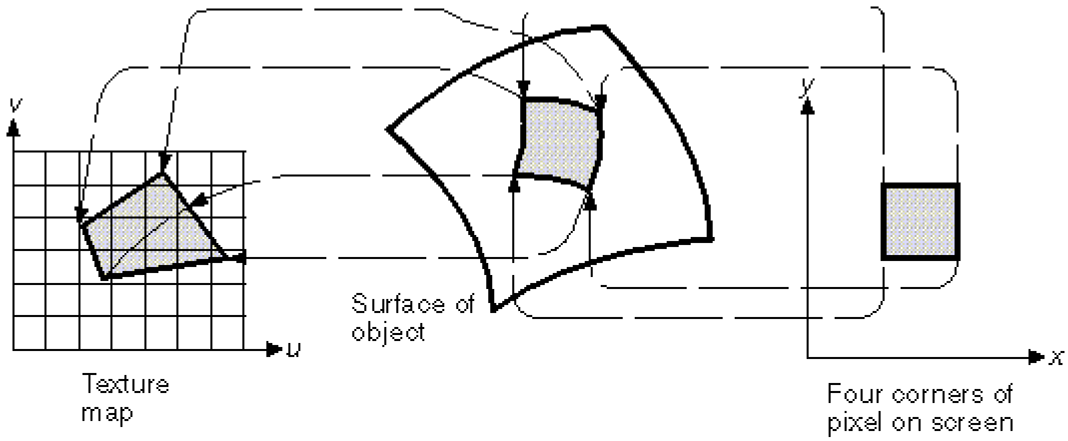


Surface detail:

- Surface detail polygons
  - like simple stickers
  - use surface-detail polygons to show features (such as doors, windows, and lettering) on a base polygon (such as the side of a building)
  - each surface-detail is coplanar with its based polygon so it does not need to be compared with other polygons during visible surface determination
  - when the base polygon is shaded, its surface-detail polygons and their material properties take precedence for those parts of the base polygon that they cover

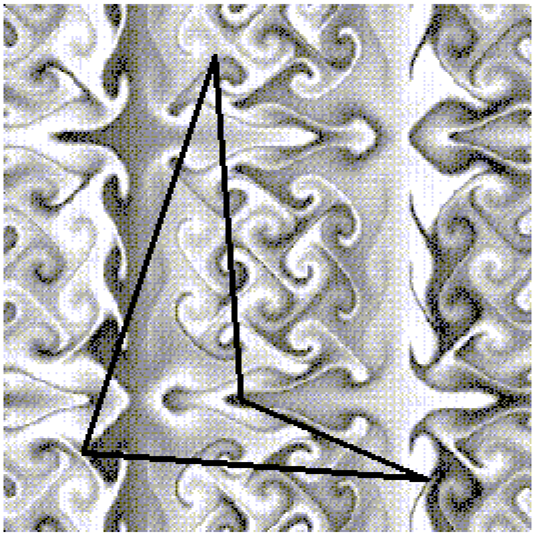
- Texture mapping
  - like more intricate deformable stickers
  - map an image onto a surface
  - each surface-detail is coplanar with its based polygon so it does not need to be compared with other polygons during visible surface determination
  - the image is called a "texture" and its individual elements are called "texels"
  - at each rendered pixel, selected texels are used either to substitute for or to scale one of more of the surface's material properties

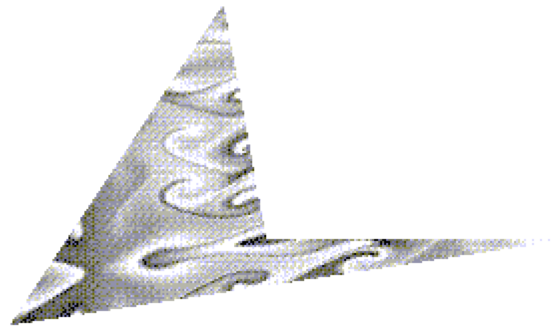
- choices exist for how the texture should be mapped to the object and how it should be applied at each pixel.



Example:

1. map the four corners of the pixel onto the surface
2. map the pixels corner points on the surface are mapped into the texture's coordinate space
3. we compute a value for a pixel by summing all of the texel that that lie within the quadrilateral

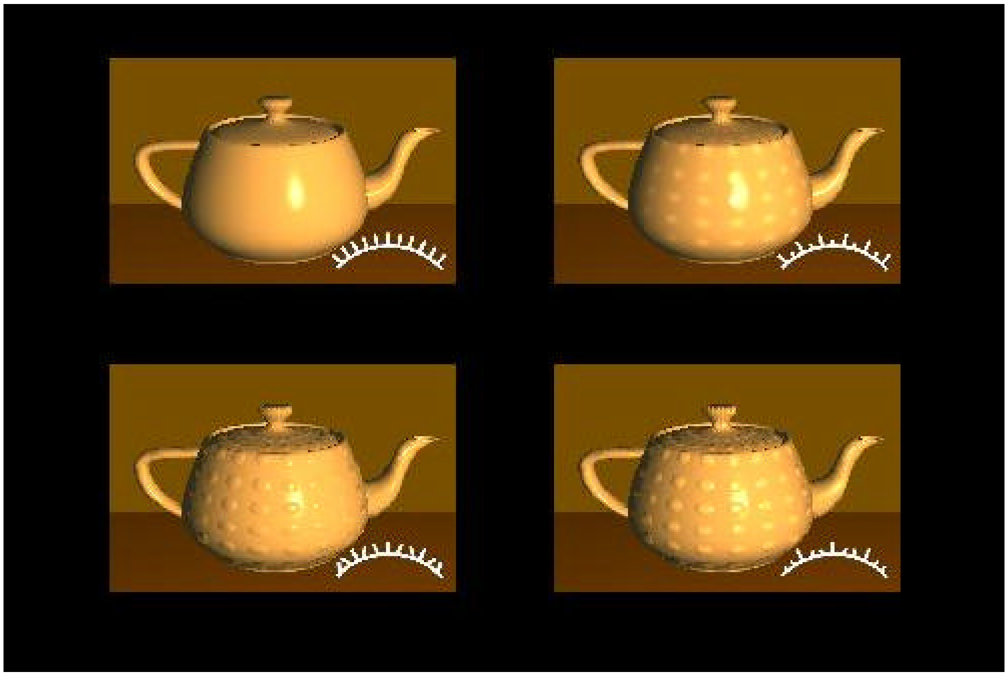




- Notice how the texture is distorted to match the distortion of the quadrilateral. In this case, it's stretched in the x direction and compressed in the y direction; there's a bit of rotation and shearing going on as well.
- Depending on the texture size, the quadrilateral's distortion, and the size of the screen image, some of the texels might be mapped to more than one fragment, and some fragments might be covered by multiple texels.

Bump mapping:

- Texture mapping affects a surface's shading, but the surface continues to appear geometrically smooth
- Bump mapping pertubates the surface normals before it is used in the illumination model just as a roughness in the surface might
- A bump map is an array of displacements, each of which can be used to simulate displacing a point on a surface a little above or below that point's actual position



Displacement Mapping (Cook)

- The actual surface is displaced instead of only the surface normals