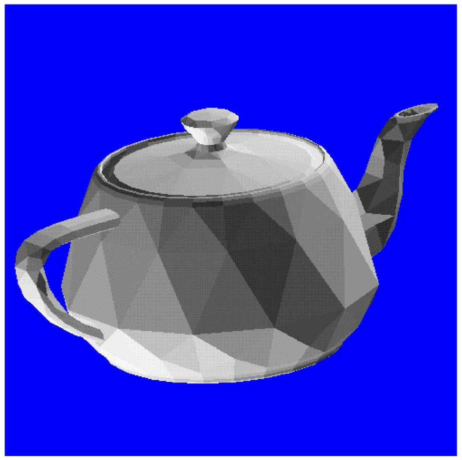
Illumination Models III: Polygon Shading

Shading algorithms apply lighting models to polygons, through interpolation from the vertices. OpenGL uses the present state to compute vertex colors, using the Phong illumination (lighting) model.

If the shading model, set by

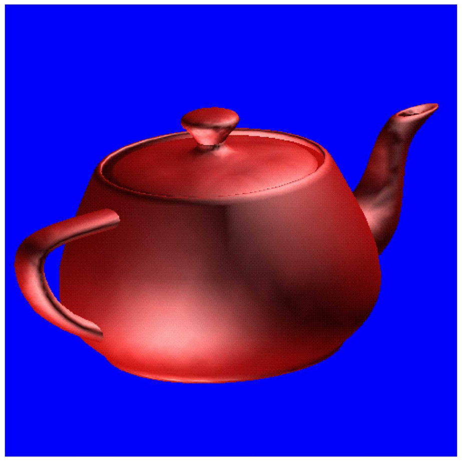
glShadeModel(GL-FLAT)



then color is computed (via the Phong lighting model) for only the first vertex in a polygon, and each polygon will appear in a solid color.

However if the polygons are used to approximate a curved object (such as the quadrics, and/or NURBs) then

glShadeModel(GL-SMOOTH)



will result in OpenGL performing the Lighting calculation at each vertex, and interpolate colors across each polygon.

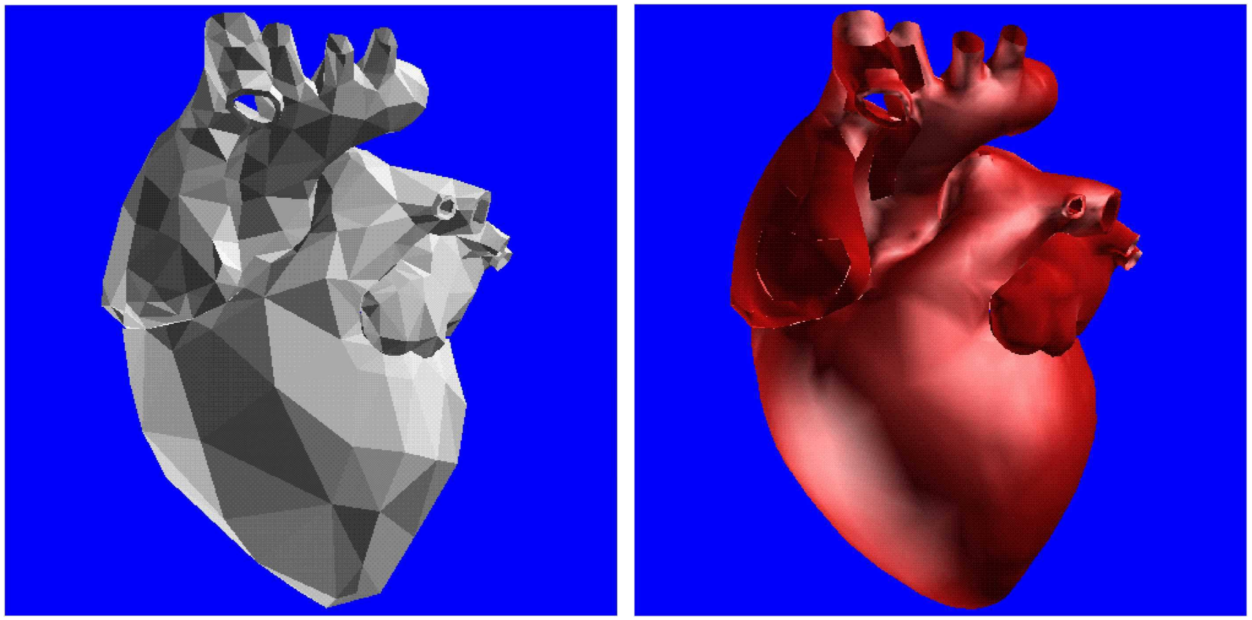
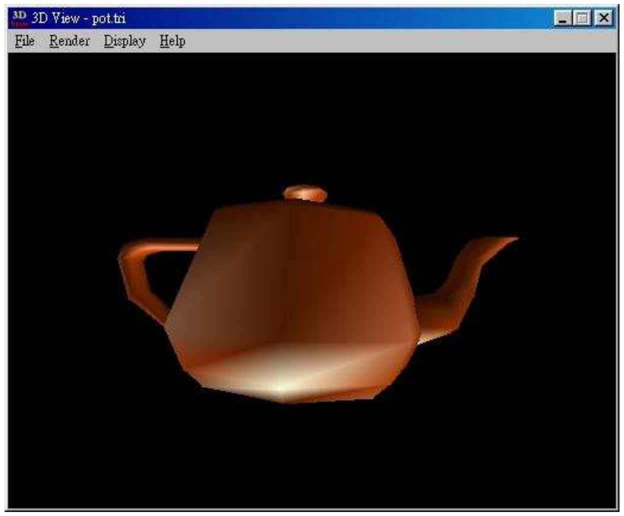


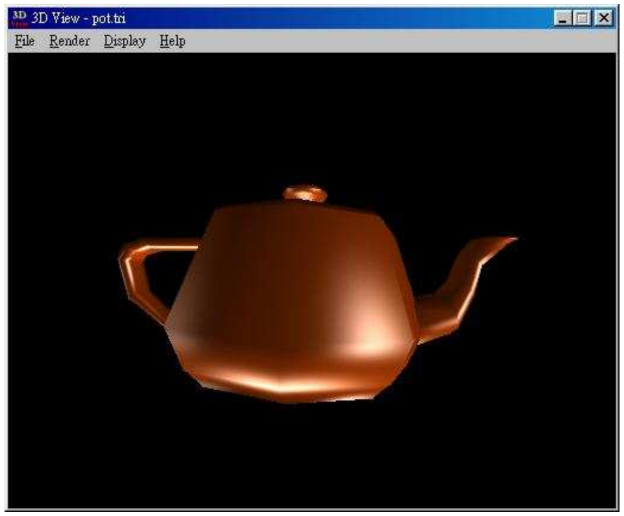
Figure 1: glShadeModel(GL-FLAT), glShadeModel(GL-SMOOTH)

Reflection Based Shading

To a large extent the quality of shading depends on normals, which we considered in the last Phong illumination lecture. This illumination calculation requires normals to have unit length. Its always more efficient to enforce this in the application program by normalizing and setting all normals to be of unit length. However, we can also enable automatic normalization by

glEnable(GL-NORMALIZE)





Shading Models: Gouraud and Phong

Gouraud Shading: Lighting is only computed at the vertices, via reflection illumination model, and the RGB reflection intensities are interpolated across the (convex) polygon

Phong Shading: A normal is specified at each vertex, and this normal is interpolated across the polygon. At each pixel, a lighting calculation is performed based on reflection illumination model.

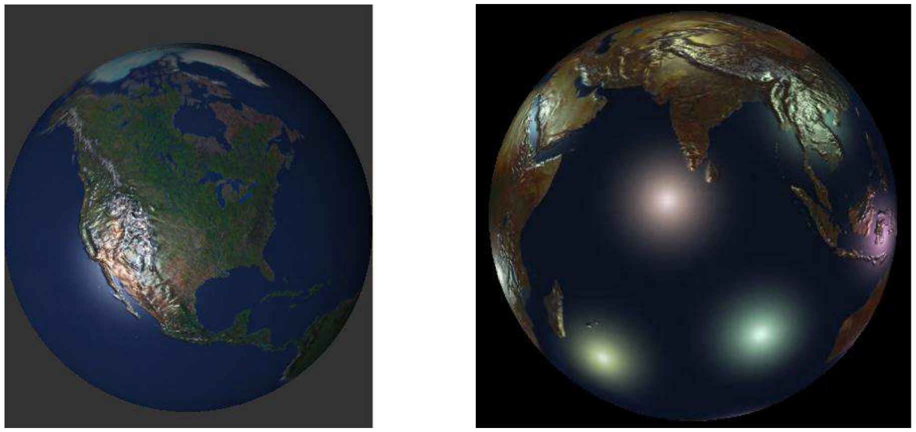
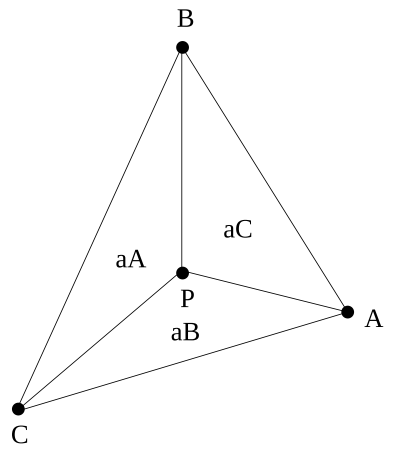


Figure 2: Gouraud vs Phong Shading

Gouraud Shading

- *Gouraud shading* linearly interpolates colors across a polygon from the color at vertices
- Lighting calculations are only performed at the vertices to obtain the color vectors
- Highlights can be missed or blurred
- Common in hardware renderers; model that OpenGL supports
- Linear Interpolation is easily defined for triangles, and quads, using *barycentric coordinates*
- Triangular Gouraud shading using *barycentric coordinates* is *invariant* under affine transformations

Linear Interpolation on Triangles for Gouraud Shading



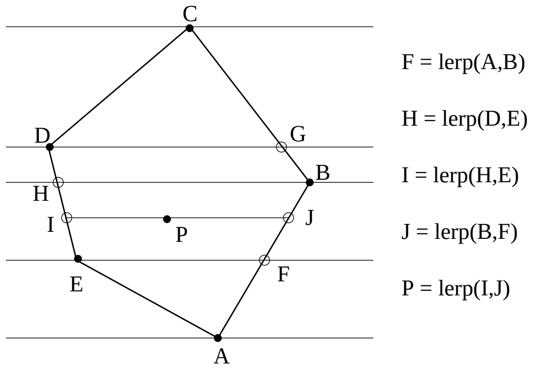
$$\alpha_A = D(P,B,C)/D(A,B,C)$$
$$\alpha_B = D(A,P,C)/D(A,B,C)$$
$$\alpha_C = D(A,B,P)/D(A,B,C)$$
$$\alpha_A + \alpha_B + \alpha_C = 1$$
$$P = \alpha_A\,A + \alpha_B\,B + \alpha_C\,C$$

$$D(A,B,C) = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & x_A & y_A & z_A \\ 1 & x_B & y_B & z_B \\ 1 & x_C & y_C & z_C \end{vmatrix}$$

- Every simple polygon P can always be decomposed into disjoint triangles , so Gouraud shading using OpenGL,via above technique is possible for all (polyhedral) shapes with polygonal faces.
- Every convex polygon Q can be easily decomposed into quads using mid-points of polygon's edges and centroid of P (as mentioned earlier in the Subdivision lecture for necessary preprocessing of Catmull-Clark schemes) . However, this causes polyhedral quad decomposition propagation or proliferation. Neverthess, it allows for Gouraud Shading using OpenGL.

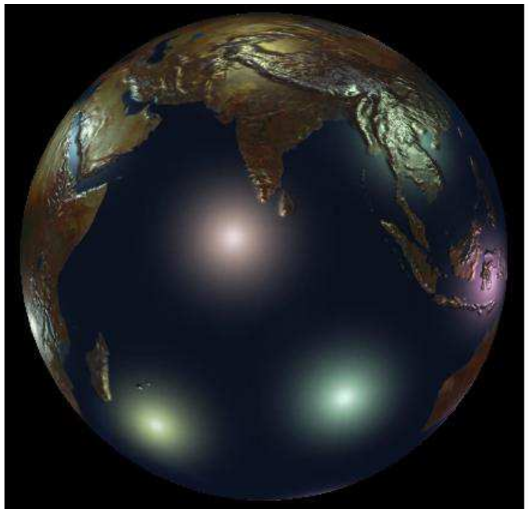
Gouraud Shading via Quad decompositions

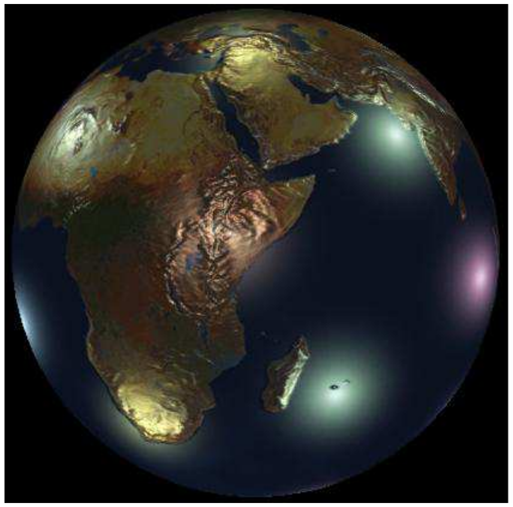
- Gouraud shading for polygons with more than three vertices:
  - Sort the vertices by y coordinate
  - Slice the polygon into trapezoids with parallel top and bottom
  - Interpolate colors along each edge of the trapezoid...
  - Interpolate colors along each scanline



- Gouraud shading gives *bilinear* interpolation within each trapezoid
- Since rotating the polygon can result in a different trapezoidal decomposition, n-sided Gouraud interpolation is *not affine invariant*
- Gouraud shading on a per pixel basis can be implemented in a similar fashion, by interpolating along each of the raster scans

Phong Shading





- *Phong Shading* interpolates lighting model parameters, *not* colors
- Much better rendition of highlights
- A *normal* is specified at each vertex of a polygon
- Vertex normals are independent of the polygon normal
- Vertex normals should relate to the surface being approximated by the polygon
- The normal is interpolated across the polygon (using Gouraud techniques).

- At each pixel,
  - Interpolate the normal...
  - Interpolate other shading parameters...
  - Compute the view and light vectors...
  - Evaluate the lighting model
- The lighting model does not have to be the Phong lighting model...
- Normal interpolation is nominally done by vector addition and renormalization
- Several “fast” approximations are possible
- The view and light vectors may also be interpolated or approximated

Next Time: Global Illumination

Reading Assignment and News

Please review the appropriate sections related to this weeks' lectures in chapter 5, and associated exercises, of the recommended text.

(Recommended Text: Interactive Computer Graphics, by Edward Angel, Dave Shreiner, 6th edition, Addison-Wesley)

Please track Blackboard for the most recent Announcements and Project postings related to this course.

(http://www.cs.utexas.edu/users/bajaj/graphics2012/cs354/)