CSE-170 Computer Graphics

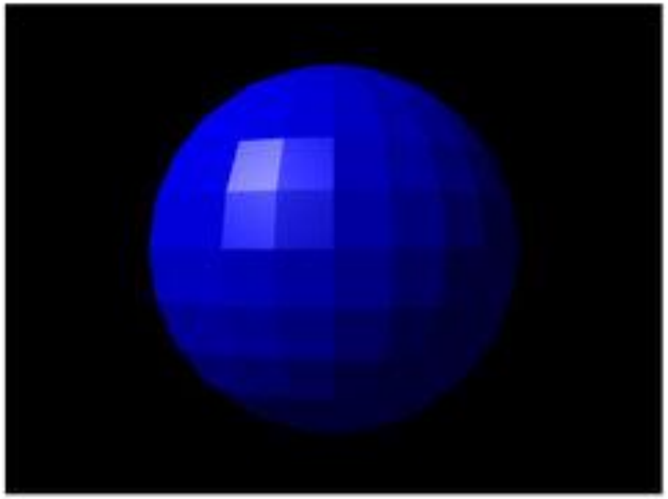
Lecture 10

Shading

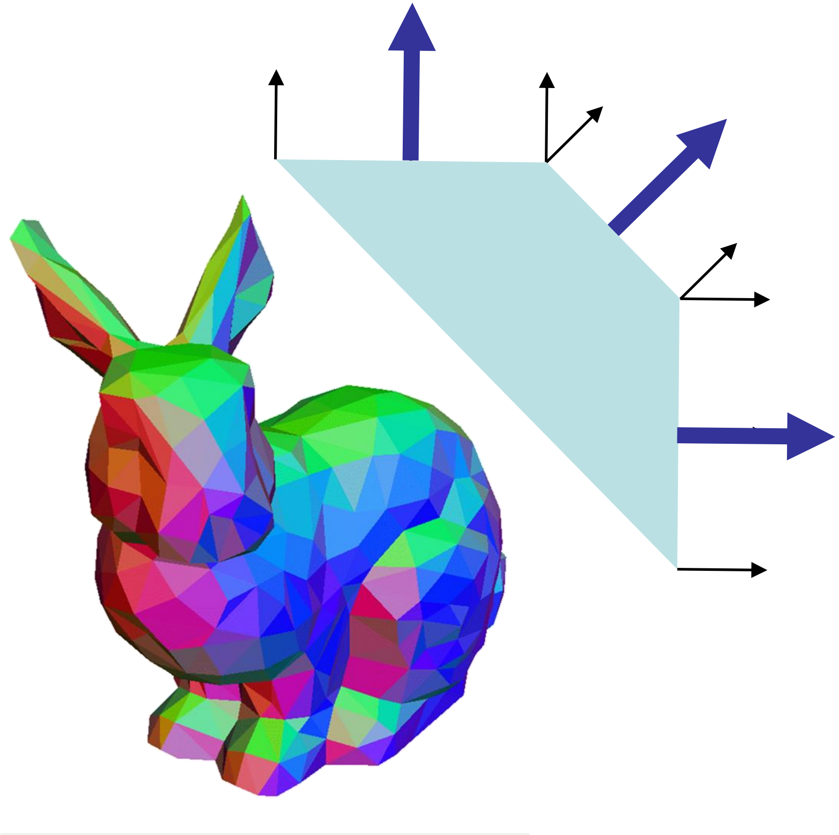
Dr. Renato Farias
rfarias2@ucmerced.edu

Flat Shading

- For each polygonal face, its normal is used to illuminate the entire face
  - Discontinuous shading occurs at the edges between the flat faces
  - Ok if the goal is to render a polyhedron with flat faces, but not for smooth surfaces such as a sphere:

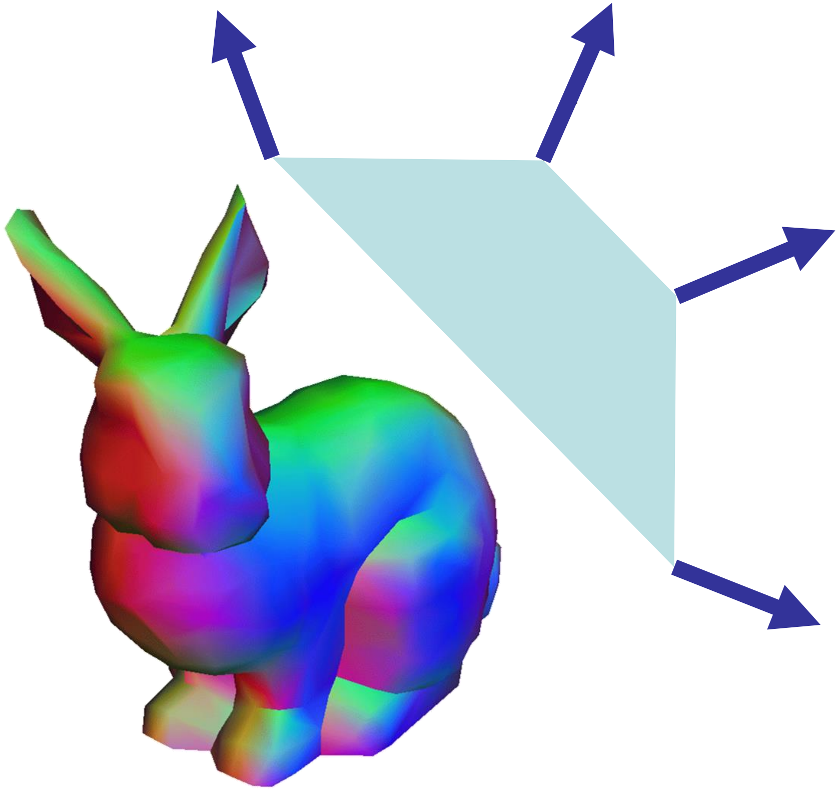


Flat and Smooth Shading



Flat Shading:

Vertices per triangle use “face normals”



Smooth Shading:

Vertices use the normal to the “ideal surface”

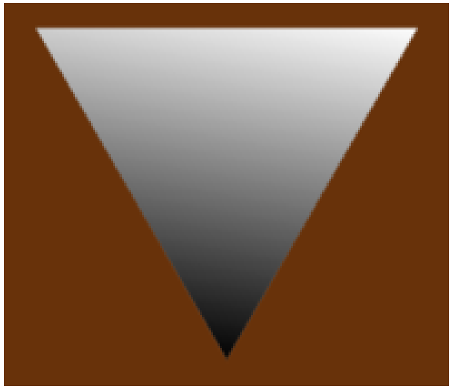
Smooth Shading

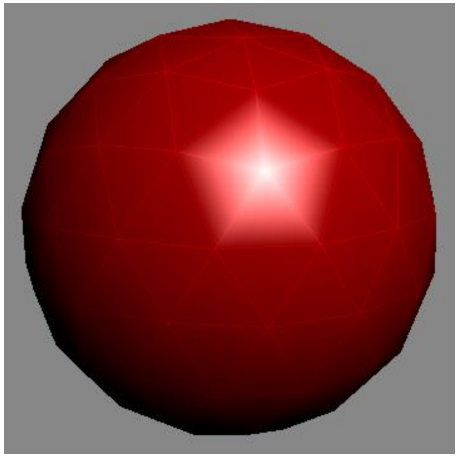
- First, to achieve correct smooth shading results, correct normal vectors are needed

- Then, two popular smooth shading techniques can be applied:
  - Gouraud shading
  - Phong shading

Gouraud Shading

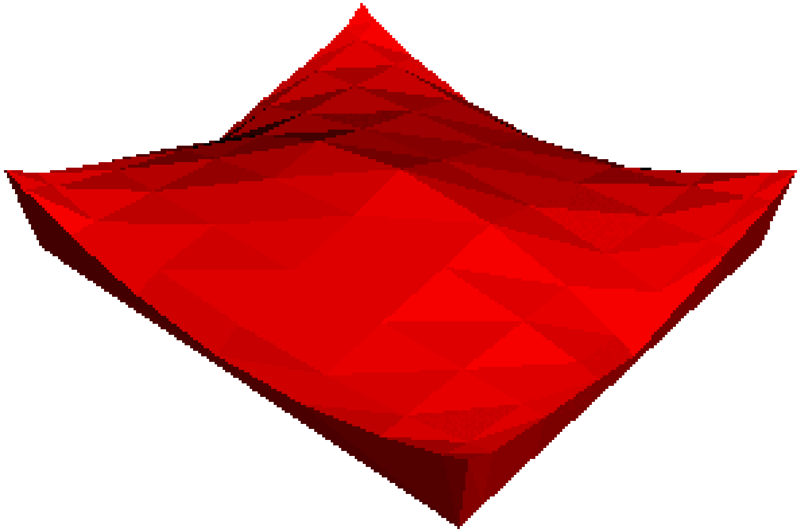
- Achieves smooth shading without computing illumination on every point inside a triangle
  - Illuminate triangle vertices, and obtain 3 colors
    - Normals are always given per-vertex
  - Then interpolate the 3 colors inside the triangle
    - Using interpolation based on barycentric coordinates
  - Problem: specular reflections in the middle of a triangle are missed



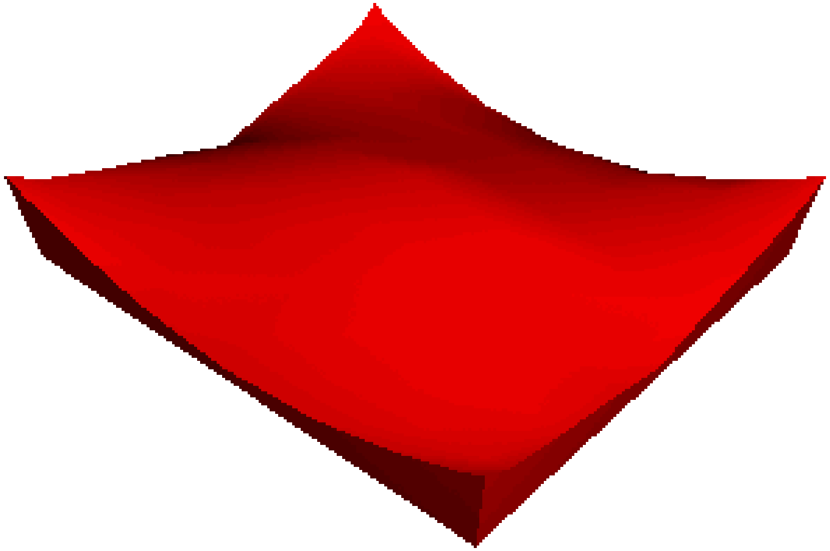


Gouraud Shading

Flat Shading



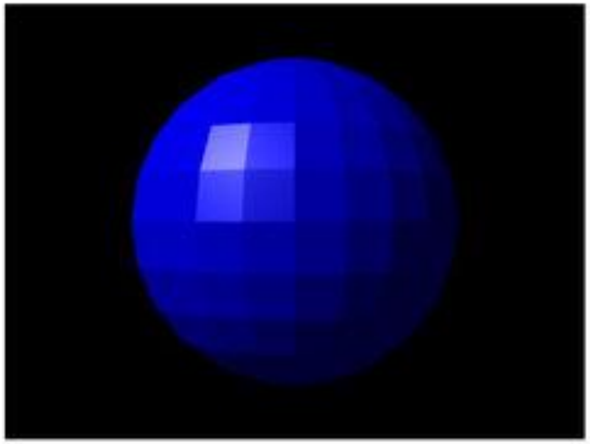
“Gouraud-Smoothed”



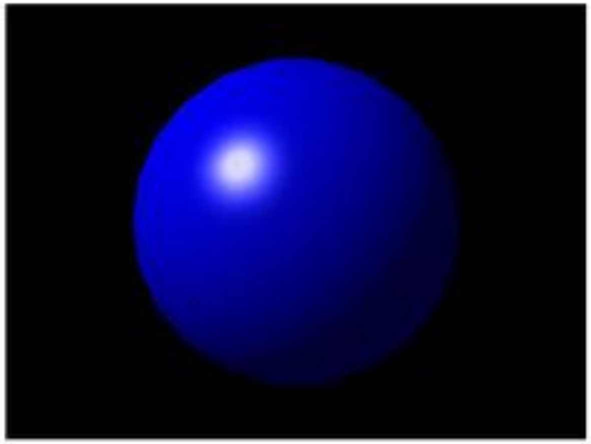
Phong Shading

- Interpolate normals from the given per-vertex normals for each interior point:
  - Each interior point will have a different normal
  - Phong illumination is then applied to every interior point using the interpolated normal
  - This fixes specular reflections!

    (it still cannot fix the “outer border polygonal appearance” of models…)



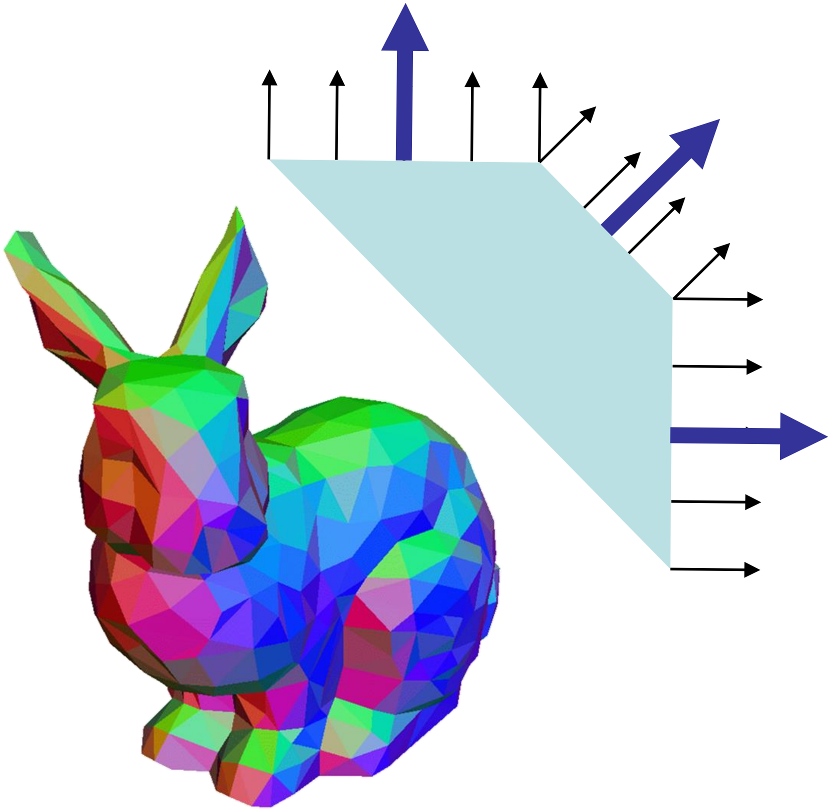
FLAT SHADING



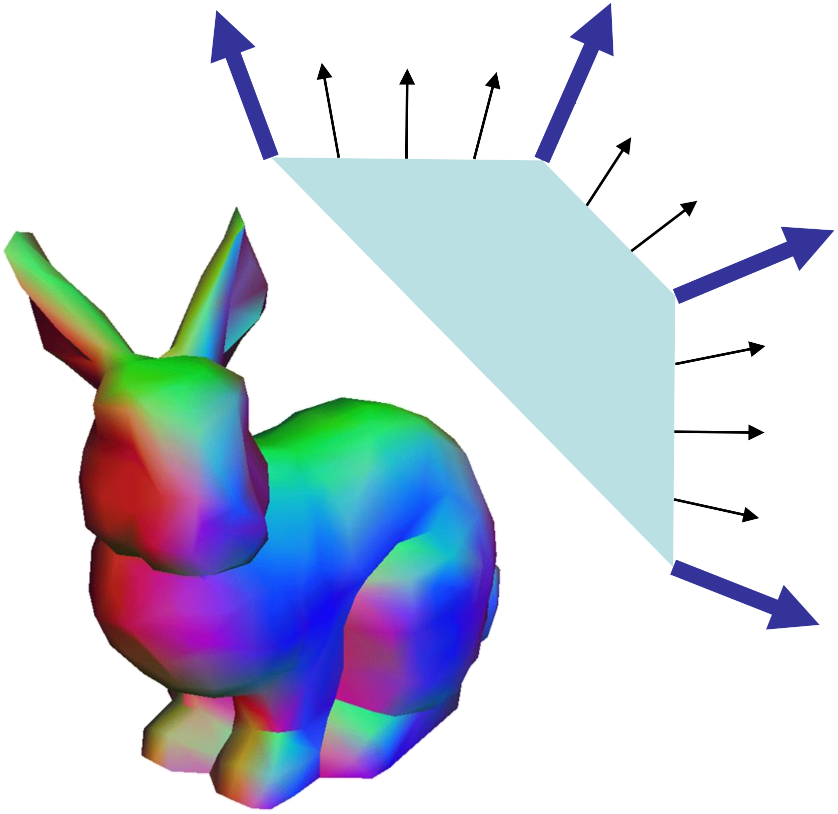
PHONG SHADING

Phong Shading

- Normals “reconstruct the ideal surface”

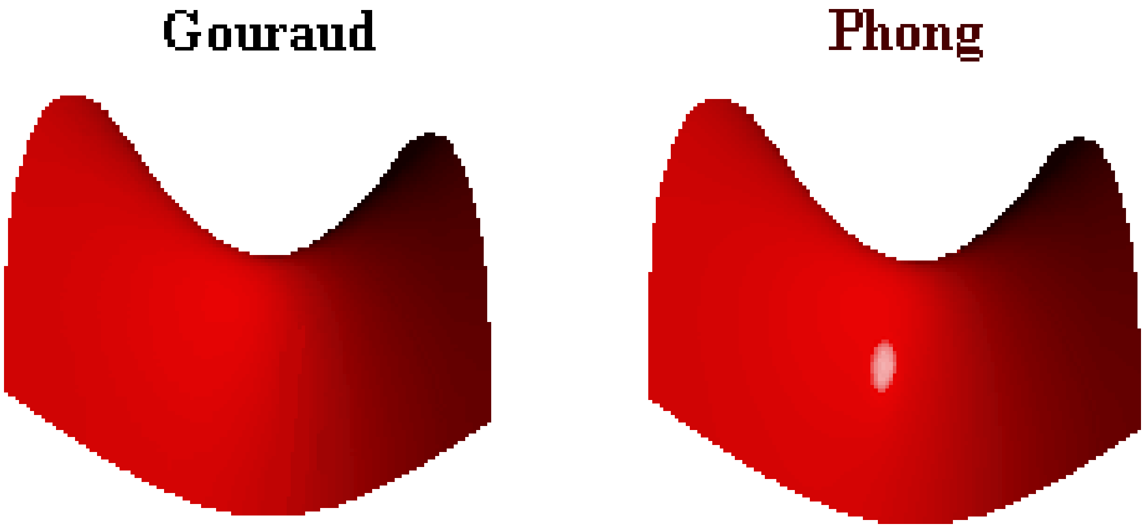


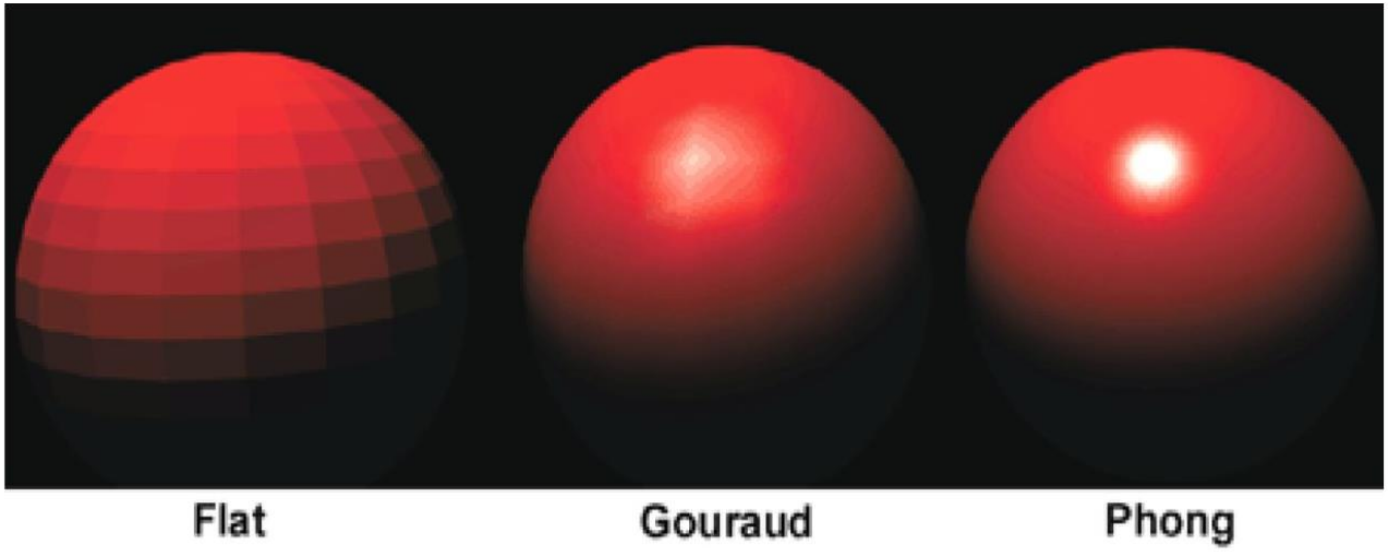
Face Normals



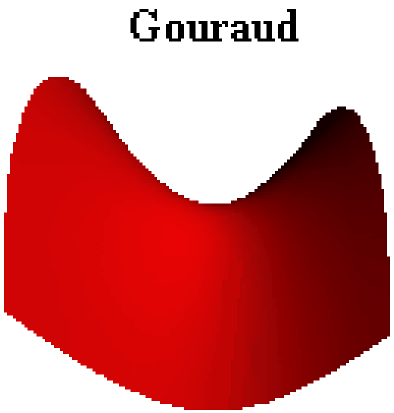
Normals for Phong
(must be renormalized during interpolation)

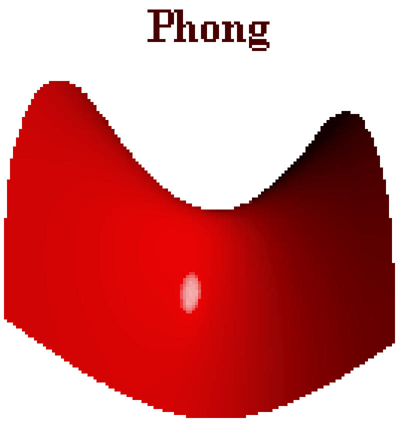
Smooth Shading





Smooth Shading





- Illumination equation only evaluated per-vertex
- Illumination equation implemented in the vertex shader

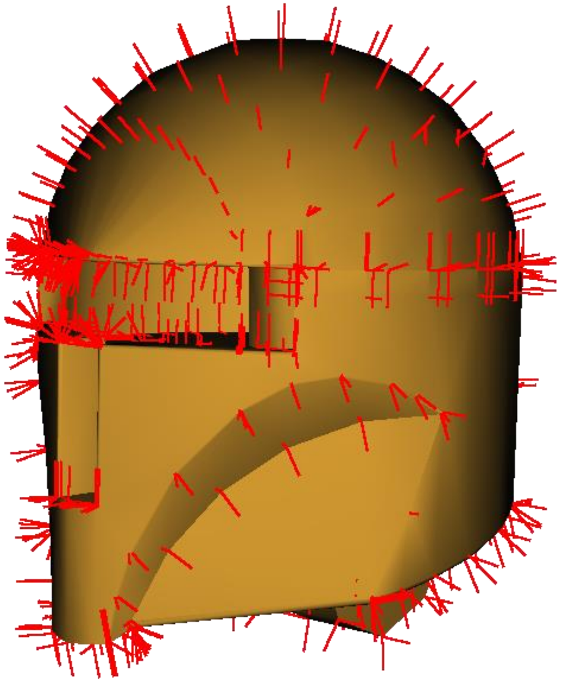
- Illumination equation evaluated per-pixel
- Illumination equation implemented in the fragment shader

Transformations and Illumination

- Important notes:
  - Remember: non-rigid transformations may not preserve angles!
    - So a transformed normal vector may not be normal to its corresponding transformed surface anymore (use the transposed inverse)
  - Do not mix...
    - Phong illumination model (an equation), with
    - Phong shading model (when all interior points are Phong-illuminated, not interpolated)

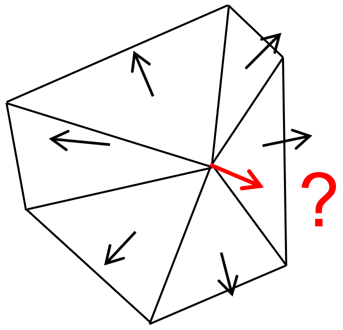
Defining Normals for Phong

- Normals are defined per vertex
  - Computed normals will define if vertices are...
    - in segments supposed to be edges
    - or in the middle of a smooth surface
  - Automatic generation of normals is possible and important
    - designers may also manually define normal vectors
    - file formats may give lists of normals per vertex, per face, etc.
  - Back-face culling optimization
    - Triangles with “normals pointing away from viewer” are not rendered



Computing Smooth Normals

- Different methods can be used to determine the normal of a vertex from the normals of the triangles sharing the vertex:
  - – Weight uniformly: just take the average
  - – Weight by area
  - – Weight by inverse area
  - – Plane fitting to shared vertices
  - – Weight by angle



Simplest approach:
use the average of the normal
vectors of all adjacent faces