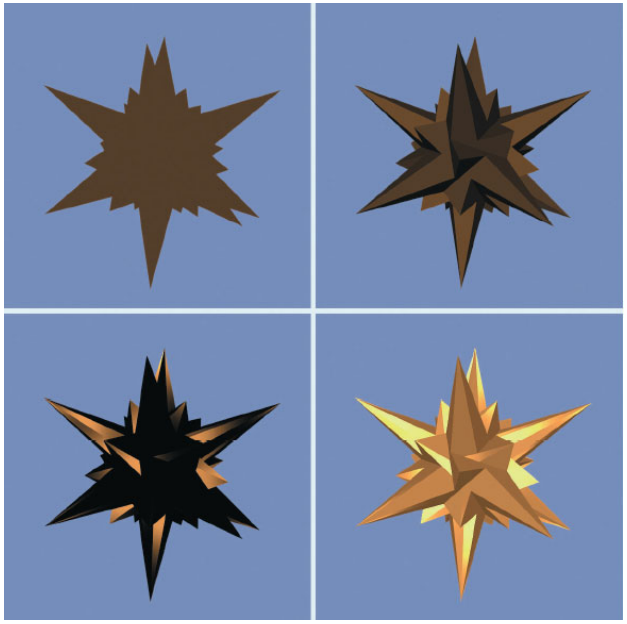
Lecture 10

Shading principles - continued

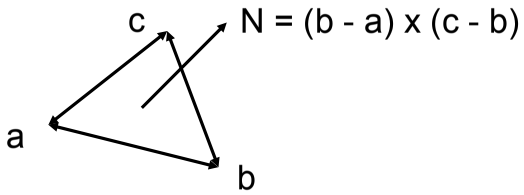
Ambient + diffuse + specular



Vertex normals vs. face normals

What are the normals to the surface?

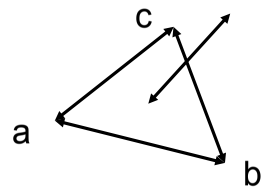
Each polygonal face has a normal.



We call these face normals.

Flat shading

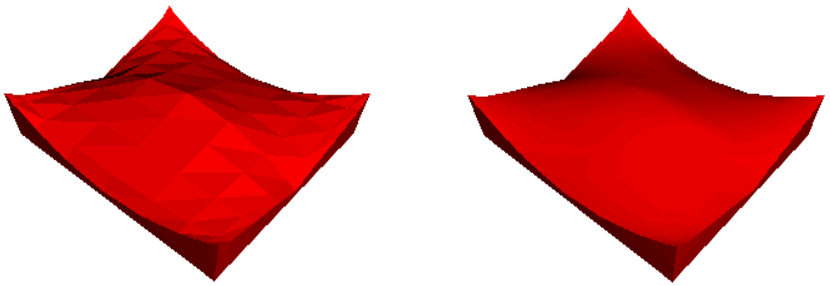
Assume a constant color across the polygon



Uses face normals
Equivalent to single point sampling…
Polygon mesh is only an approximation.
Can we do better?

Flat vs. smooth shading

The following surfaces contain IDENTICAL geometry!

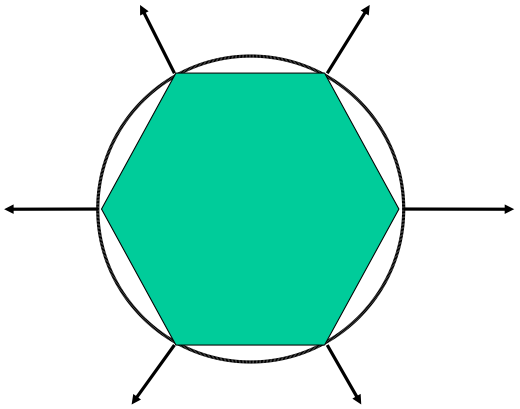


Images source: Naval Postgraduate School, Monterey, CA

Where does the smoothness on the right come from?

Vertex normals vs. face normals

Should use the actual surface's normals



Usually stored at the vertices of the object
Can calculate as averages of face normals

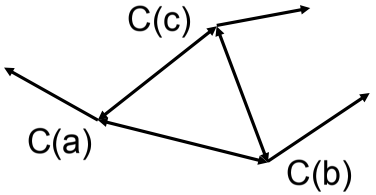
Interpolation

Given vertex normals, how to color the interior:

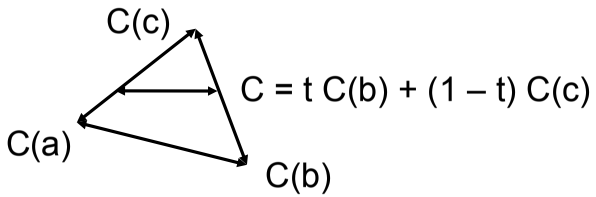
- Gouraud interpolation
- Phong interpolation

Gouraud [color] interpolation

1.calculate the color at each vertex

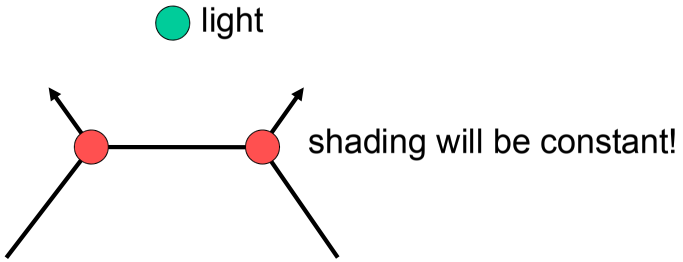


2. interpolate vertex colors at desired location (pixel)



Gouraud interpolation problems

Misses some highlights

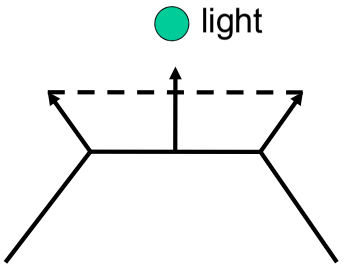


Shading is not linear

Phong [normal] interpolation

1. interpolate vertex normals at desired location

2. compute the color using interpolated normal



Interpolation is usually done component-wise (x,y,z)

Interpolation: flat vs. Gouraud vs. Phong

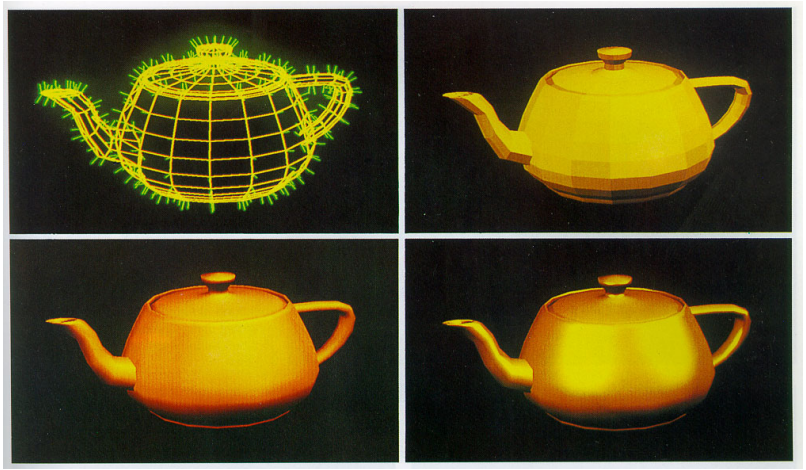
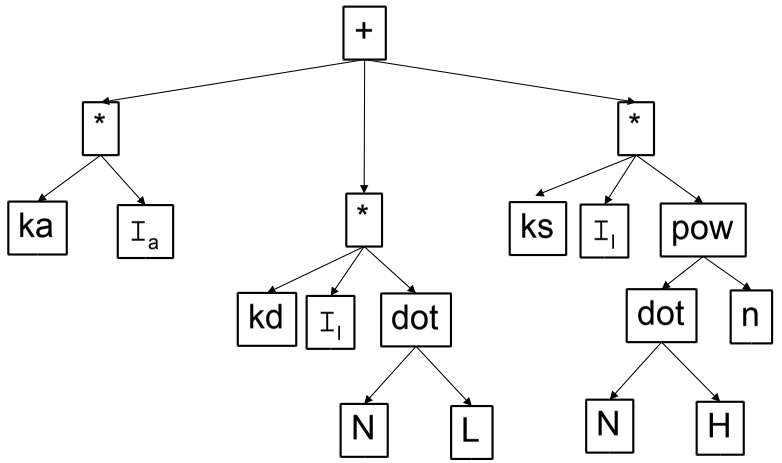


Image courtesy of Watt & Watt, Advanced Animation and Rendering Techniques

Shade trees

Phong shade tree:

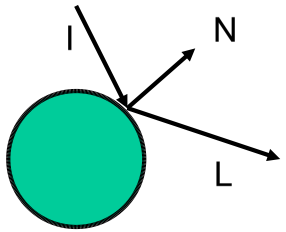


Shading language (RenderMan)

A language for implementing shading models

State is passed to/from the shader by global variables
Ci – Outgoing ray color
Oi – Outgoing ray opacity

Cs – Surface color
Os – Surface opacity
P – Surface point
N - Surface normal
I - Direction of viewing (eye ray)
L - Direction to the light source



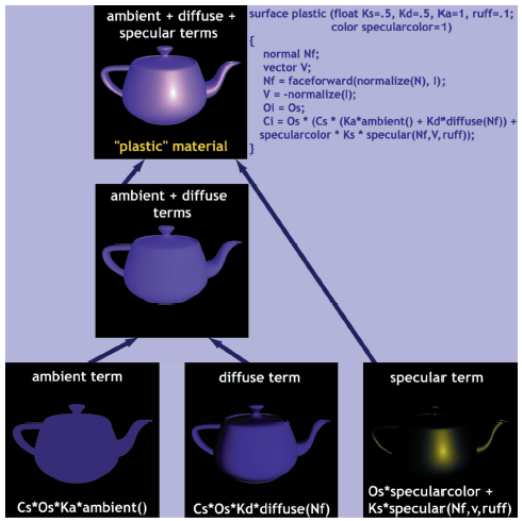
Shading language, sample shader

Code (RSL) for Phong or 'plastic' surface shader:

```
surface
plastic(float Ks = .5, float Kd = .5, float Ka = 1,
        float roughness = .1, color specularColor = 1)
{
  point Nf = faceforward(N,I);

  Oi = Os;
  Ci = Os*(Cs*(Ka*ambient()+Kd*diffuse(Nf)
       + specularColor*Ks*specular(Nf,-I,roughness);
}
```


Shade 'tree' for plastic shader



Constant, matte, plastic shaders