CS174A Lecture 12

Announcements & Reminders

- *Team project proposals due (final version): Nov 8*
- *Projects due: Dec 1*
- *Project presentations: Dec 3 and 5, in class*
- *Final exam: Dec 12*

TA Session This Friday

- ***Team project proposals***

Last Lecture Recap

- ***Lighting/Illumination Models***
  - Ambient
  - Diffuse
  - Specular

Next Up

- ***Barycentric Coordinates, Trilinear Interpolations***
- ***Flat and Smooth Shading***
- ***Mappings: Texture, Bump, Environment, Displacement***
- ***Hidden Surface Removal***
  - 2-pass z-buffer algorithm (shadows)
  - Ray casting

Recall: Interpolation Formulas

Special Cases

Linear combination

$\mathbf{w} = a_1\mathbf{v}_1 + \ldots a_m\mathbf{v}_m, \quad a_1,\ldots,a_m$ in R

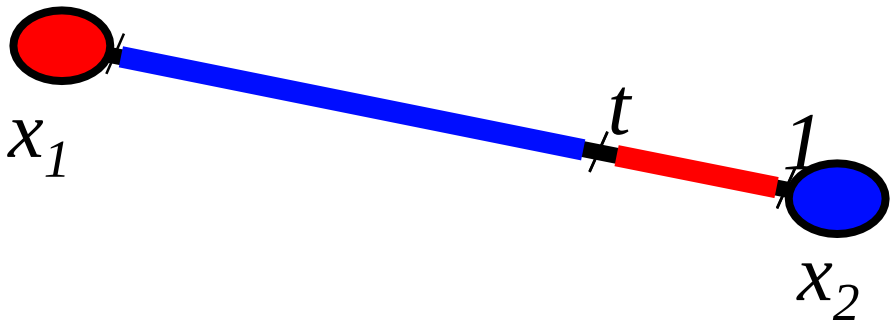
Affine combination:

A linear combination for which $a_1 + \ldots + a_m = 1$

Convex combination

An affine combination for which $a_i \geq 0$ for i = 1,…,m

Barycentric Interpolation



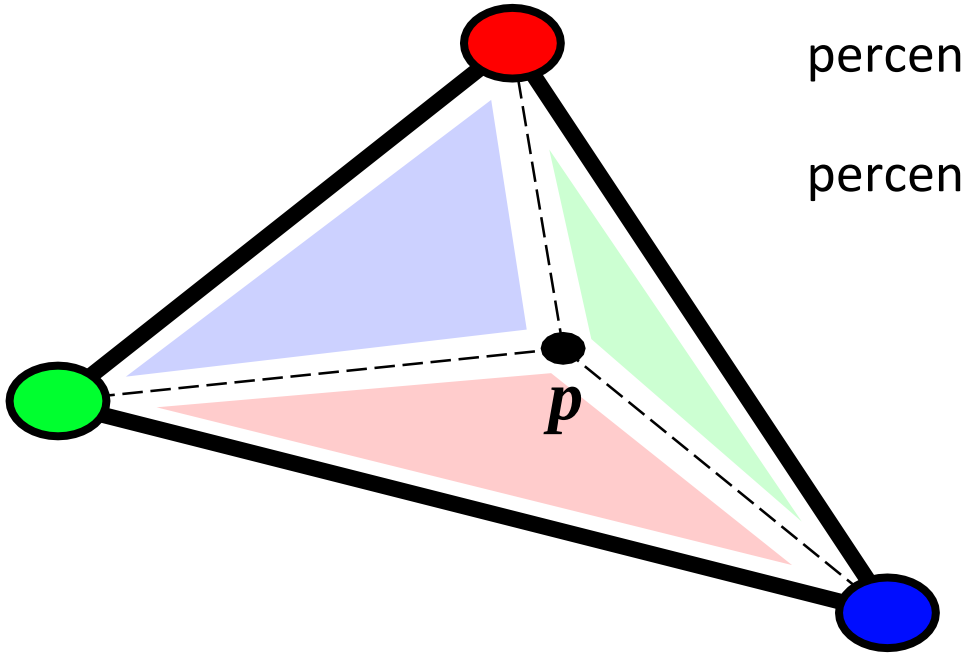
The further t is from the red point, the more blue we want. The further t is from the blue point, the more red we want.

Percent blue = t

Percent red = 1-t

Value at t = $tx_1 + (1-t)x_2$

Barycentric Interpolation



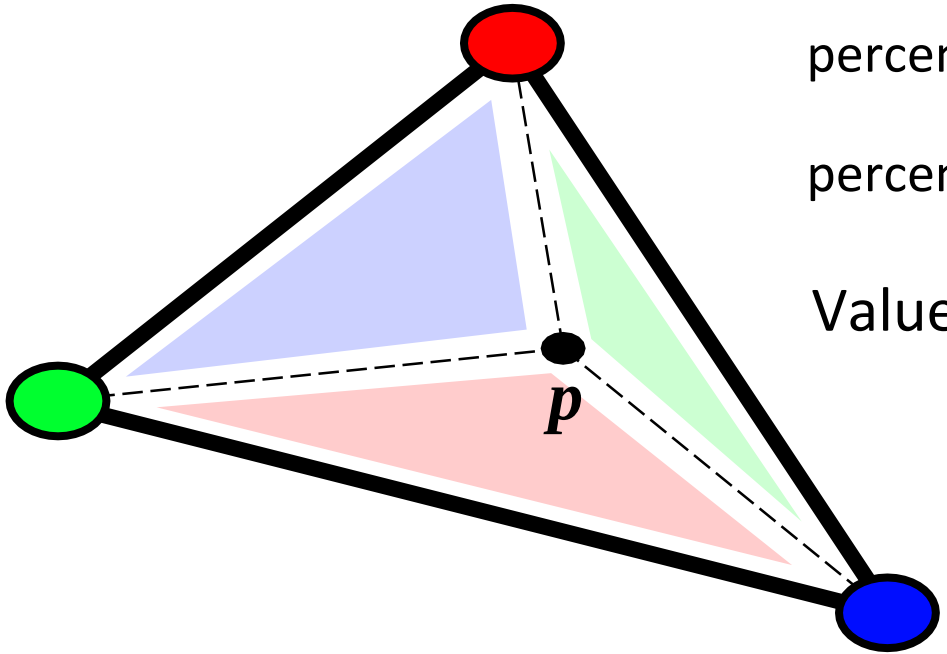
Just like before:

percent red = $\frac{\text{area of red triangle}}{\text{total area}}$

percent green = $\frac{\text{area of green triangle}}{\text{total area}}$

percent blue = $\frac{\text{area of blue triangle}}{\text{total area}}$

Barycentric Interpolation



Just like before:

percent red = $\frac{\text{area of red triangle}}{\text{total area}}$

percent green = $\frac{\text{area of green triangle}}{\text{total area}}$

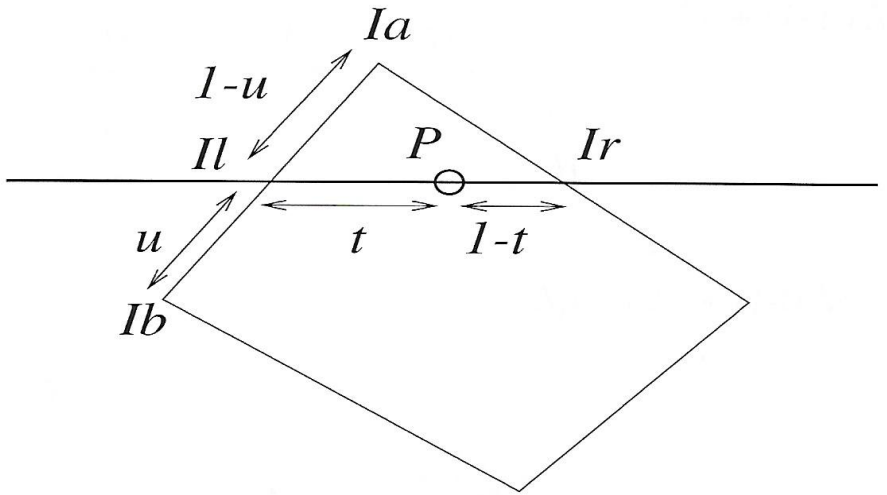
percent blue = $\frac{\text{area of blue triangle}}{\text{total area}}$

Value at $\boldsymbol{p}$:

(% red)(value at red) +
(% green)(value at green) +
(% blue)(value at blue)

Trilinear Interpolation

- Can be used to interpolate z, color, normal, texture, etc.
- Interpolate along 2 edges
- Interpolate along scanline
- Incremental calculations



Polygon vs. Vertex Attributes

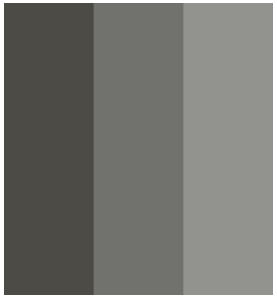
- Polygon Attributes
  - Color
  - Normal
- Vertex Attributes
  - Coordinates (position)
  - Color
  - Normal
  - Texture coords

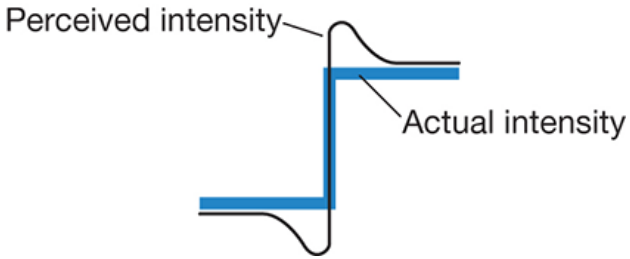
Flat or Constant or Faceted Shading

- Use N of poly
- Find I at center or at any vertex of poly
- Apply that same color to all points inside poly
- In essence, it means:
  - N is constant across poly
  - Light is at $\infty \Rightarrow N \cdot L$ is constant across poly
  - Viewer is at $\infty \Rightarrow N \cdot V$ is constant across poly
- Which space to compute N and illuminate?
  - Either WS or ES, not in PS

Problem with Flat Shading

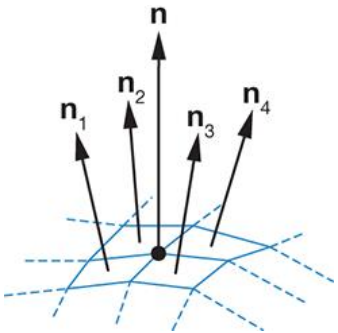
- Mach bands is an optical illusion named after the physicist Ernst Mach. It exaggerates the contrast between edges of the slightly differing shades of gray, as soon as they contact one another, by triggering edge-detection in the human visual system. https://www.youtube.com/watch?v=sItlLNhhiLg

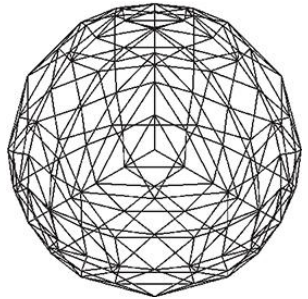




Gouraud Smooth Shading

- Also called “Intensity” interpolation or “Color” interpolation shading
- Find I at each vertex and illuminate vertex
- Interpolate across poly face
- Store normals @ vertices
  - Average normals of polys sharing vertex
  - During tessalation, e.g., sphere
  - What about hard edges?





Gouraud Smooth Shading

- Issues with Gouraud Shading
  - Rotating polygons
  - Specular reflection with large polygons
    - At poly's center
    - At poly's vertex
  - Mach banding not completely eliminated

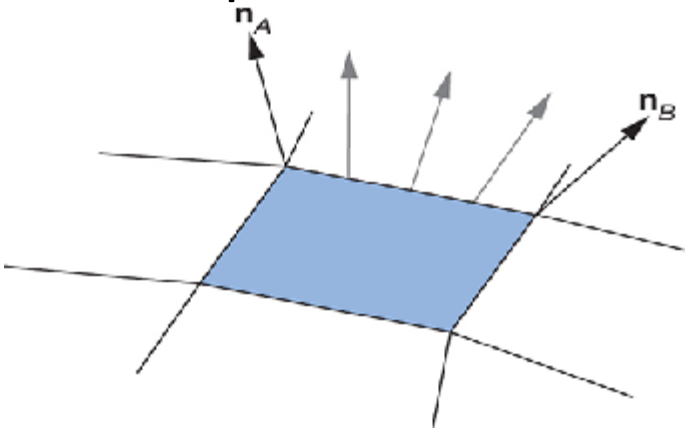
GuerrillaCG Series: Flat vs Gouraud Shading

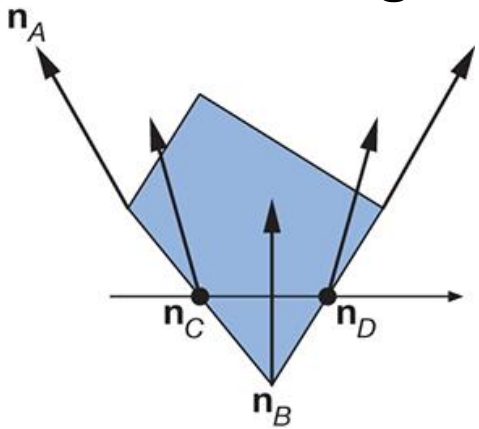
https://youtu.be/PMgjVJoglbc?t=6

- For this to work, shapes must be modelled a certain way (“seams”)
- Must store multiple vertices touching the same position
- Must be able to store conflicting normal data even if position data matches

Phong Smooth Shading

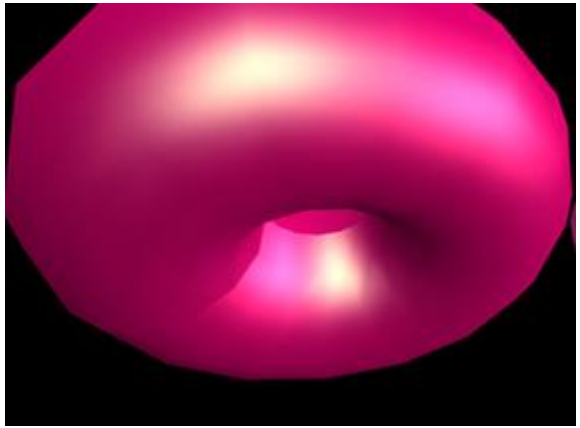
- Also called “Normal Vector” interpolation shading
- Calculate intensity at each pixel
- Interpolate normals similar to others, then normalize
- Much more computation, but much better looking images

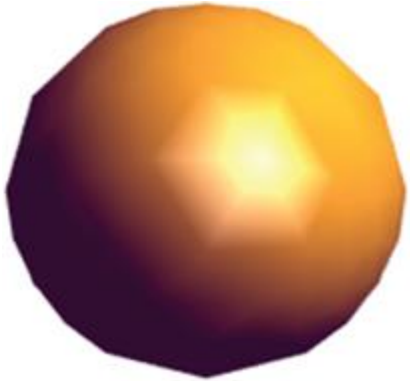




Smooth Shadings

- Issues with interpolated shading
  - Polygon sillouhette
  - Orientation dependence (triangles ok)
  - Problems at shared vertices
  - Unrepresentative vertex normals





Flat vs Smooth Shading

- Problem with this step:
  - Distribute it to all three vertices identically
- **What if some vertices are shared by more than one triangle?**
  - Could happen when we are summarizing our triangles using lists of “indices” into a list of (non-repeating, unique, more compact) vertices

Flat vs Smooth Shading

- **What if some vertices are shared by more than one triangle?**
  - A vertex can't have two normals at once, logistically!
  - Other triangles will fight over assigning normals to a vertex
  - Conclusion: Sometimes we should put repeats in our vertex list. We shouldn't try to re-use indices at seams.
    - Instead, store extra/duplicate vertices at a position, with differing normals.