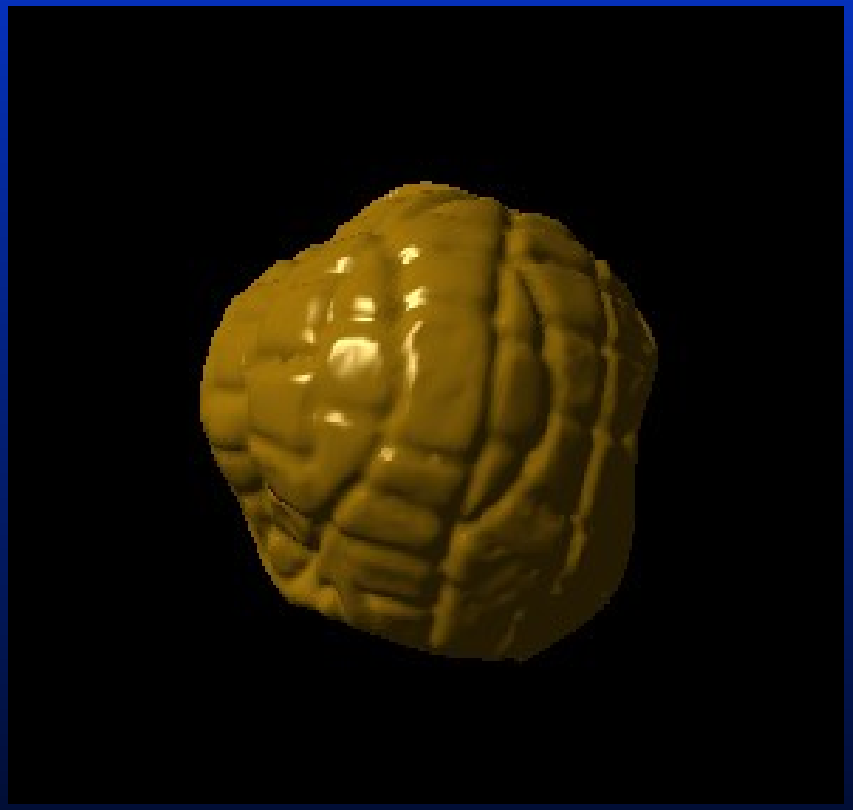
Appearance-Preserving Simplification

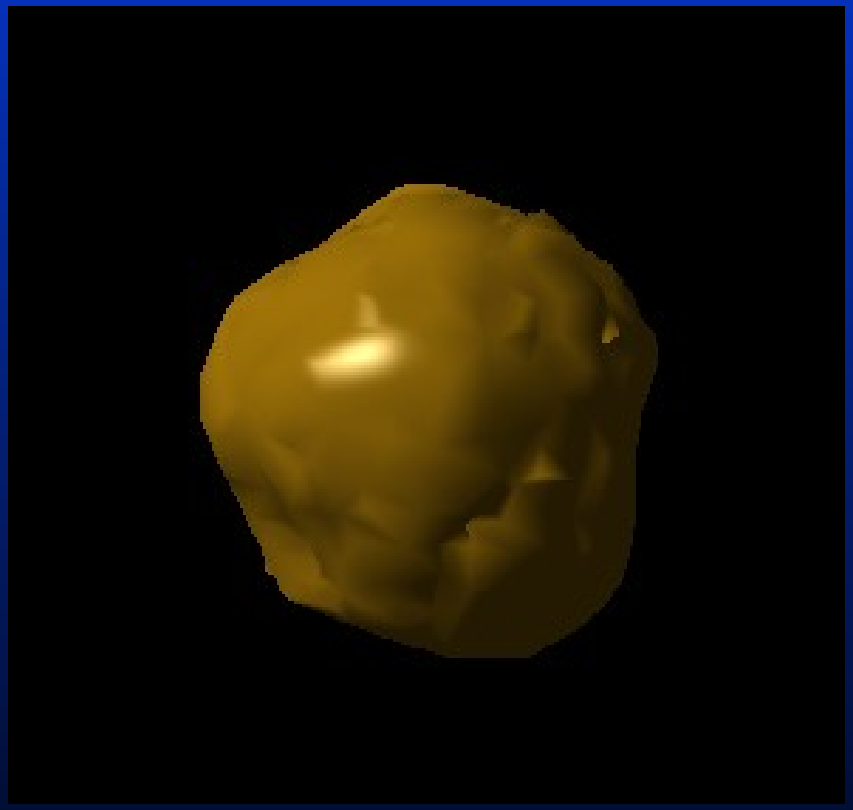
Jonathan Cohen
Computer Science Department
Johns Hopkins University
<cohen@cs.jhu.edu>
http://www.cs.jhu.edu/~cohen

Appearance Preservation

- Preserve three appearance attributes:
  - Surface Position
  - Surface Curvature
  - Material Color
- Each may require different sampling

Normals Undersampled

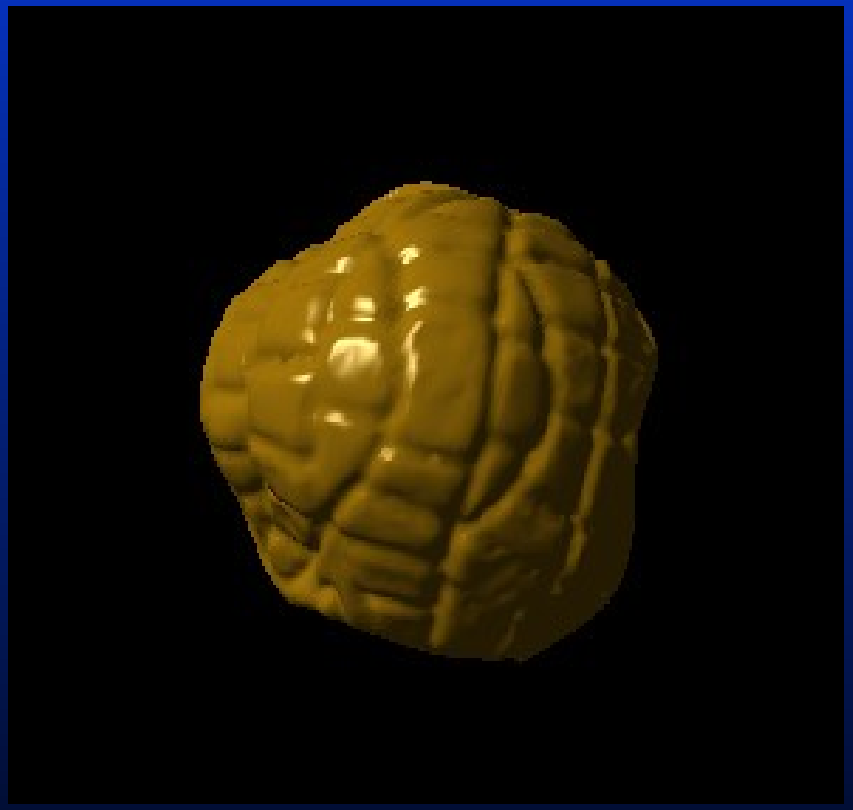




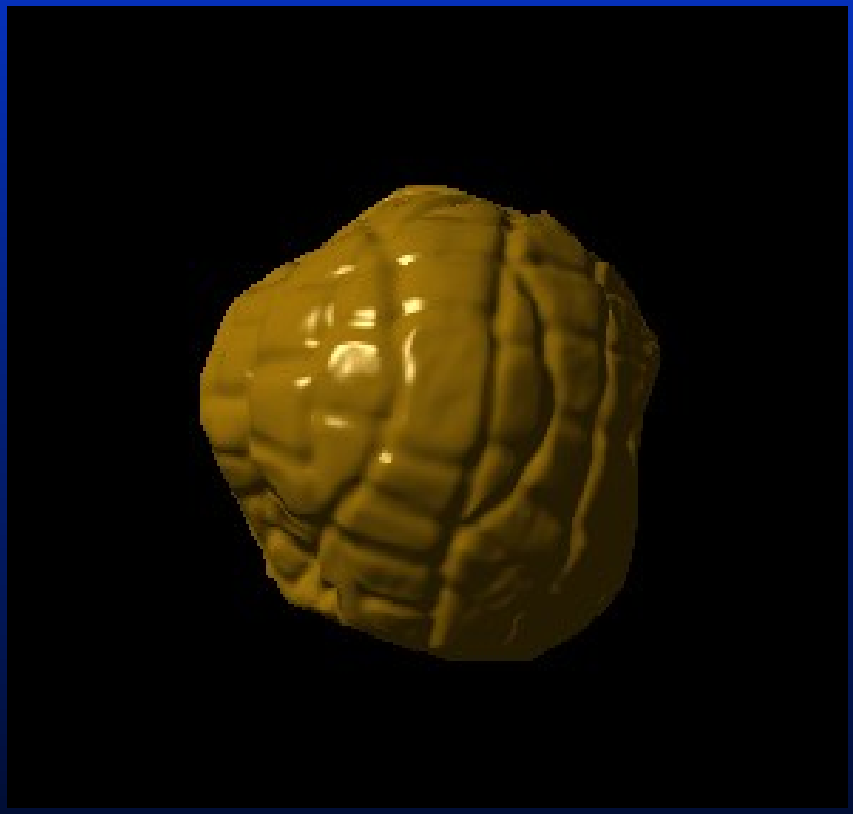
13,433 triangles

1,749 triangles
10 pixels of surface deviation

Normals Properly Sampled

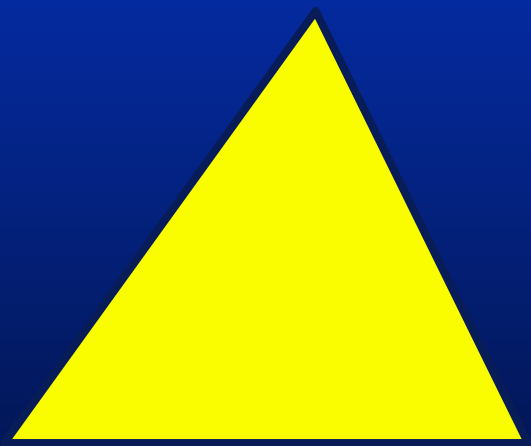


13,433 triangles



1,749 triangles,
10 pixels of deviation

Traditional Polygonal Representation



v = vertex coordinate = (x,y,z)

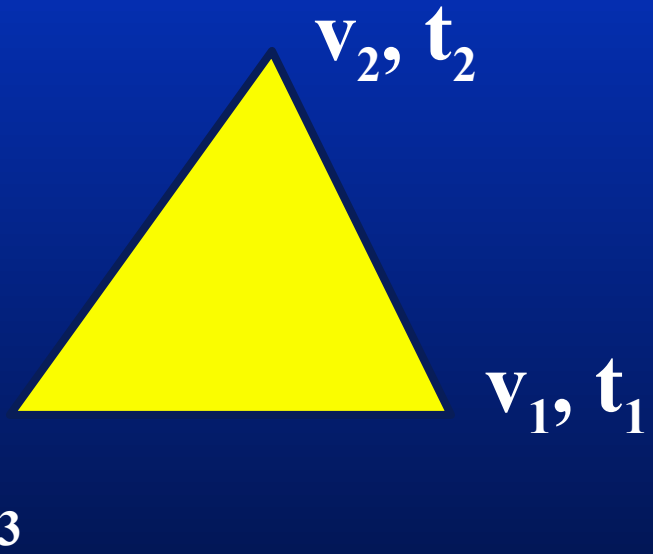
c = color = (r,g,b)

n = normal = (n_x,n_y,n_z)

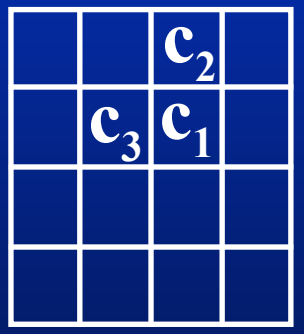
Traditional Simplification

- Filters surface position, colors, and normals
- Must filter all three *equally*

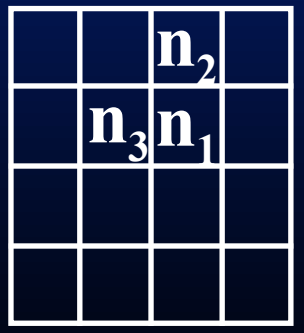
Decoupled Representation



texture map



normal map



v = vertex coordinate = (x,y,z)

t = texture coordinate = (u,v)

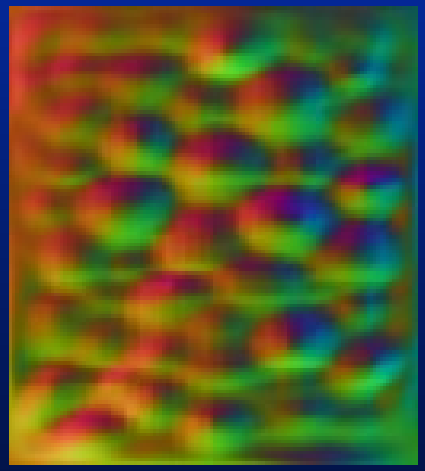
c = color = (r,g,b)

n = normal vector = (n_x,n_y,n_z)

Decoupled Approach

- Simplification filters surface position and texture coordinates
- Color and normal attributes filtered per-pixel (mip-mapping, etc.)

Sample Normal Map



polygonal surface patch **normal map**

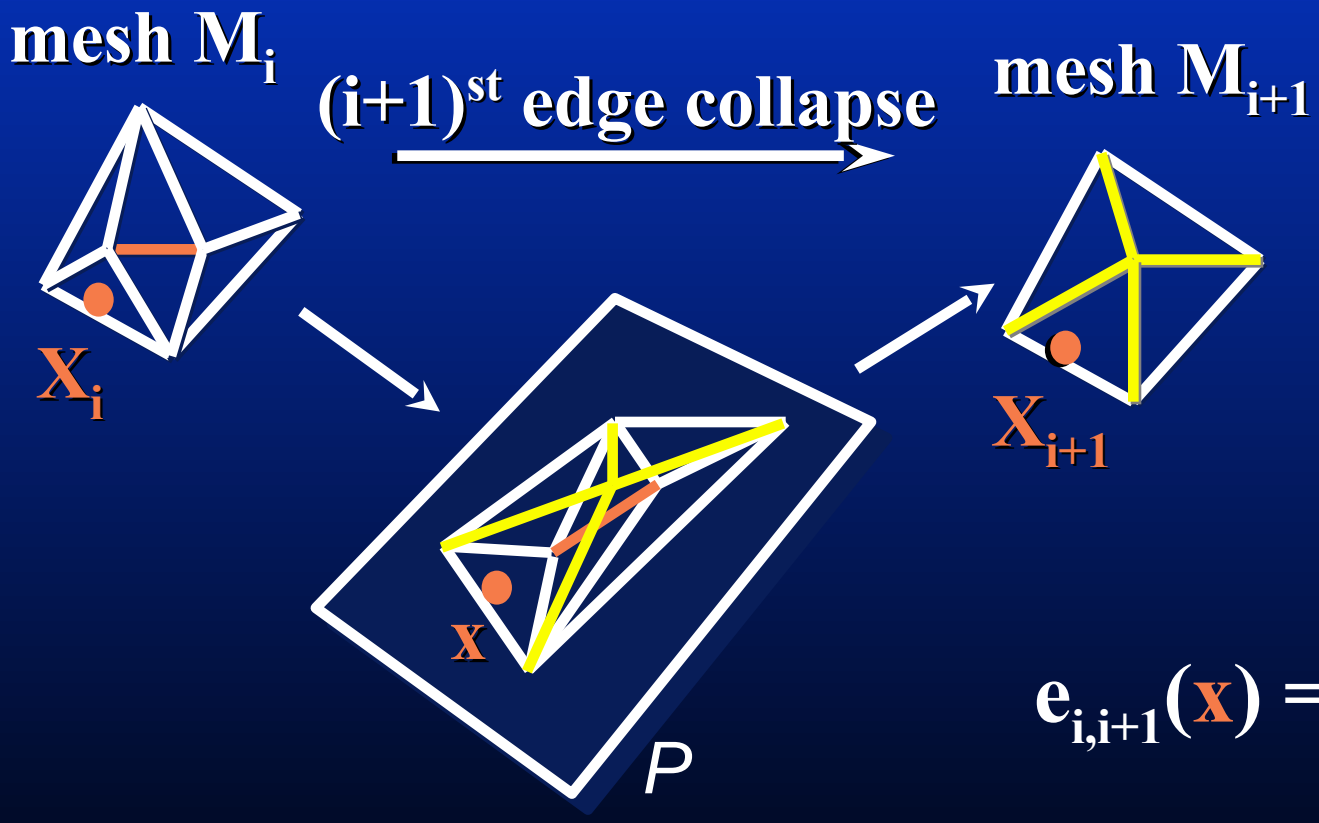
Normal Map vs. Bump Map

- Normal map
  - Absolute normals in object space
  - Constant as object is simplified
    - Same normal map okay for all LODs
- Bump map
  - Perturbations of triangle normal
  - Changes as object is simplified
    - Need different bump map for each LOD

Texture Deviation Metric

- Distance between *corresponding* 3D points
  - Same 2D texture coordinates
  - Projects at run time to 2D pixel deviation
- Intuitive error tolerance specification
  - Pixels of deviation for both surface position and texture error!

Point Correspondence



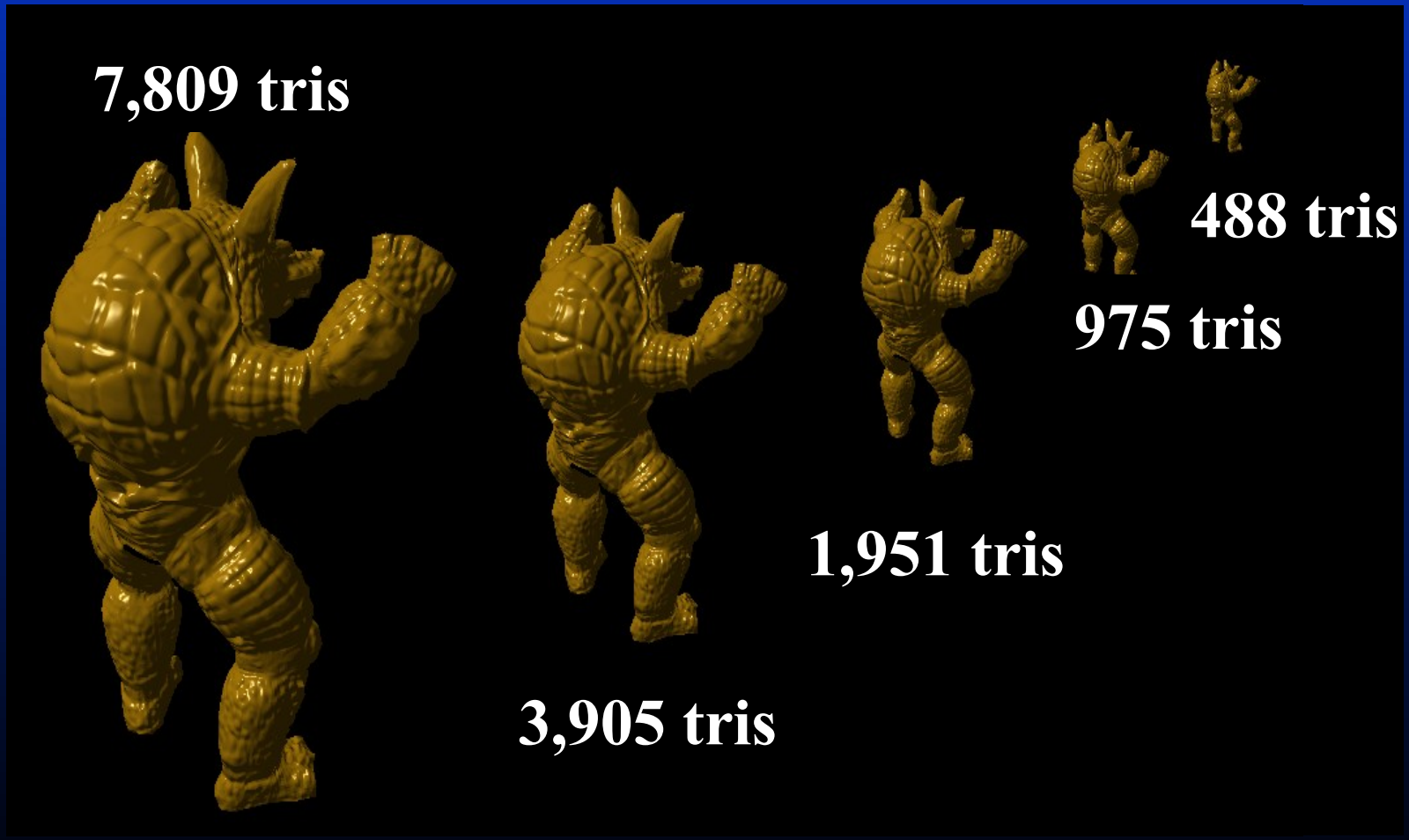
$$e_{i,i+1}(x) = \| X_{i+1} - X_i \|$$

$$E_{i,i+1} = \max_{x \in P} e_{i,i+1}(x)$$

Hardware Requirements

- Texture and normal (or bump) map capability
  - Bandwidth for attribute map lookups
  - Per-pixel lighting computation
- Demonstrated on PixelFlow
- Possible on commercial hardware
  - Normal map shading by Heidrich 99
- Used more recently in displaced subdivision surfaces

APS Level-of-detail Hierarchy



model courtesy of Stanford and Caltech

Original

250,000 Tris

Phong Shading

62,000 Tris

3 pixel error

Original

250,000 Tris

Normal Map

62,000 Tris

3 pixel error

Original

250,000 Tris

Phong Shading

8,000 Tris

15 pixel error

Original

250,000 Tris

Normal Map

8,000 Tris

15 pixel error

Original

250,000 Tris

Phong Shading

1,000 Tris

78 pixel error

Original

250,000 Tris

Normal Map

1,000 Tris

78 pixel error

APS Executive Summary

- Colors and normals stored in texture and normal maps
- *Texture deviation* computed using parametric correspondence
- Preserves colors and normals, bounding texture motion in object and screen space