Lecture 20: LODs

Fall 2004
Kavita Bala
Computer Science
Cornell University

Research on many lights

- Ward '91

- Shirley, Wang, Zimmerman '94

- Fernandez, Bala, Greenberg '02

- Wald and Slusallek '03

- Environment Map Sampling…

Rendering w/ Environment Maps
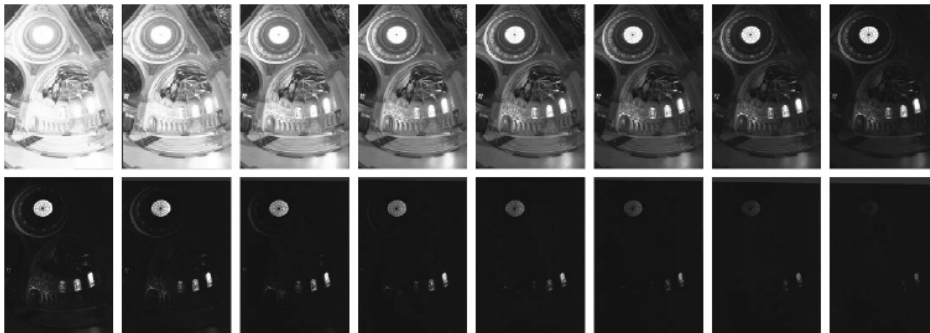
- High lighting complexity

- Rich: captures real world

High-Dynamic Range Imagery
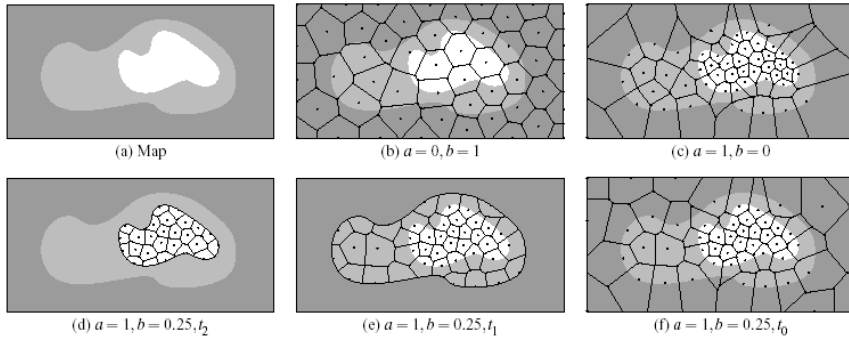
- Multiple Exposures

Sampling Environment Maps

- Say we have a fixed budget of samples
  - Say 300 or 3000

- Goal: sample environment maps $L^a \Delta \omega^b$
- Basic idea: sample according to …
  - Illumination importance sampling: over-samples small bright lights
  - Area-based stratified sampling: distribute samples over area (under-samples bright lights)

New Metric

- Instead $\quad L\Delta\omega^{\frac{1}{4}}$

(a) Map

(b) $a=0, b=1$

(c) $a=1, b=0$

(d) $a=1, b=0.25, t_2$

(e) $a=1, b=0.25, t_1$
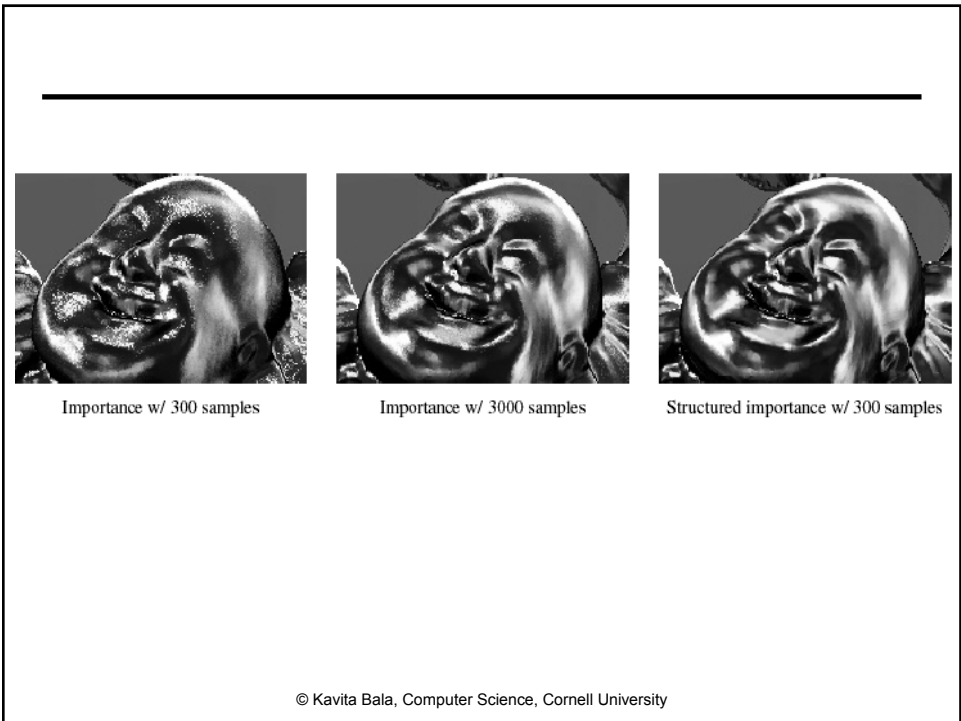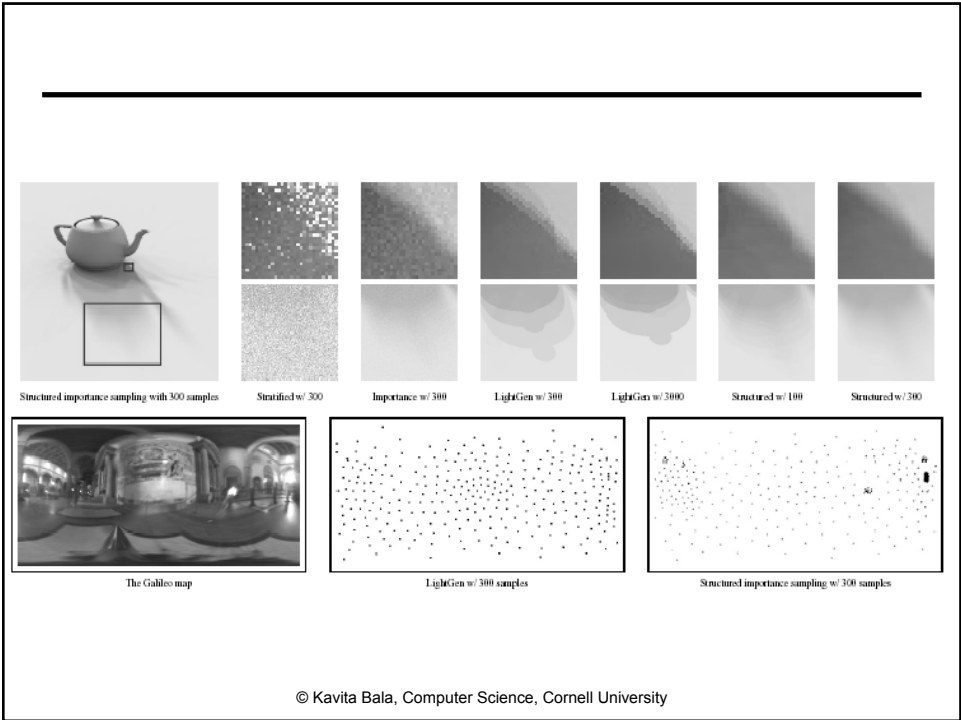
(f) $a=1, b=0.25, t_0$

Hierarchical Threshold

- Create regions from $i\sigma$ to $(i+1)\sigma$

- Distribute samples in region according to metric
$$N_j = N \frac{Metric_j}{Metric_{map}}$$

- Distribute samples in each region according to Hochbaum-Shmoys
  – Minimize maximum distance

Structured importance sampling with 300 samples | Stratified w/ 300 | Importance w/ 300 | LightGen w/ 300 | LightGen w/ 3000 | Structured w/ 100 | Structured w/ 300

The Galileo map

LightGen w/ 300 samples

Structured importance sampling w/ 300 samples

Importance w/ 300 samples

Importance w/ 3000 samples

Structured importance w/ 300 samples

Summary

- Rendering with many lights remains an open problem
  - Techniques are linear in number of lights
  - Hard to handle visibility

- Some interesting new approaches needed for interactive rendering with many lights

Complexity

- Lighting: many lights, environment maps
  - Global illumination, shadows

- Materials: BRDFs, textures

- Geometry: Level-of-detail, point-based representations

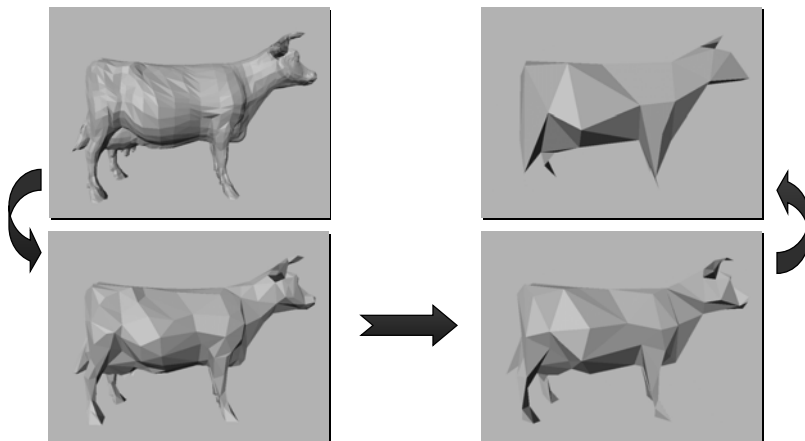- All: impostors, image-based rendering

LOD

- What is Level of detail used for?

- When close, want detail

- When far, want approximation

- *Level of Detail* (LOD) techniques draw models at different resolutions depending on their relevance to the viewer

LOD Models (I)

LOD Models (II)

LOD Models (3)

- Effect of LOD less obvious if smooth shaded
- Note also that shading changes with LOD
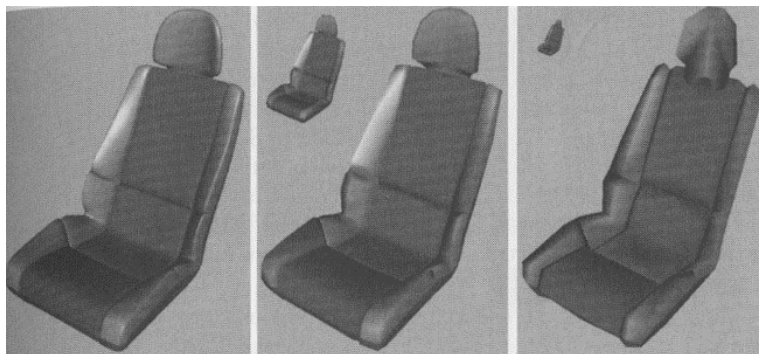
Standard LOD

- Used in games
  - Finite set of models
  - Hand generated
  - Or using automatic decimation/simplification

- Use of LODs: based on distance from eye or projected screen size

Issues with LODs

- LOD Creation

- LOD Selection: Choosing which model to display

- LOD Switching: Artifact-free switching between LODs

Selecting LOD Models

- For any given frame, decide what resolution to display

- Option 1: Choose each object's resolution independently
  - For example, use projected area or distance
  - The current standard practice: fast, simple
  - Potential problem: total # polygons may be large

- Option 2: Fix number of polygons and choose models to fit in budget
  - Ensures near constant frame rate, but harder to implement

LOD Switching

- Popping when models switched
- Popping is visually disturbing
  - Why?
  - Flickering when at threshold

- Solutions:
  - Have more resolutions in hierarchy
  - Blend two resolutions
    - Image blend: draw both resolutions
   - Geometry blend: morph between resolutions

Creating the LOD

- Convert high resolution *base mesh* to hierarchy of lower resolution meshes

- Desirable properties:
  - Fast (although not real-time)
  - Generates "good" approximations in some sense
  - Handles a wide variety of input meshes
  - Allows for geometric blending

Vertex Clustering

- Spatial partition
- Merge all vertices in a cell
- Fast but not good quality

Before   After
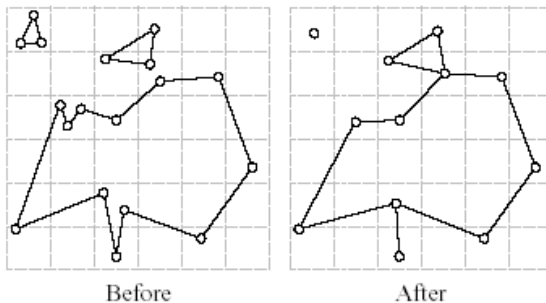
Vertex Decimation

- Starting with original model, iteratively
  - rank vertices according to importance
  - Remove unimportant vertex, retriangulate
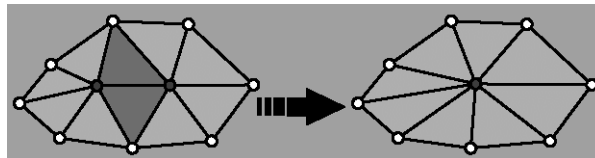 - a fairly common technique
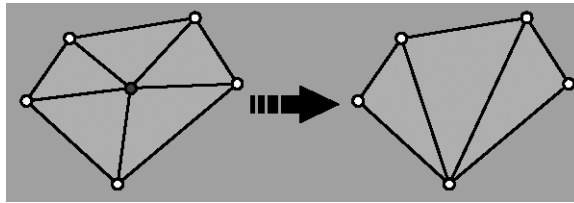
Slide courtesy Michael Garland, http://graphics.cs.uiuc.edu/~garland

Iterative Edge Contraction

- Contraction can operate on any set of vertices
  - edges (or vertex pairs) are most common

- Starting with the original model, iteratively
  - rank all edges with some cost metric
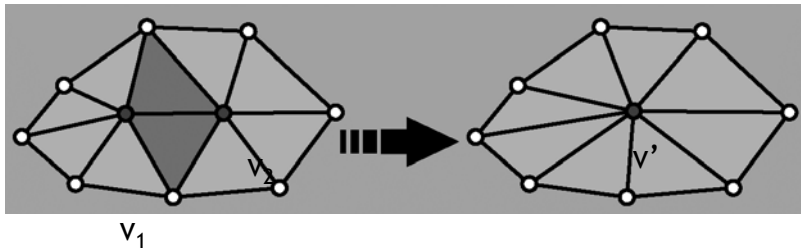 - contract minimum cost edge
  - update edge costs

Slide courtesy Michael Garland, http://graphics.cs.uiuc.edu/~garland

Edge Contraction

- Single edge contraction $(v_1, v_2) \rightarrow v'$ is performed by
  - moving v_1 and v_2 to position v'
  - replacing all occurrences of v_2 with v_1
  - removing v_2 and all degenerate triangles

Slide courtesy Michael Garland, http://graphics.cs.uiuc.edu/~garland

Operations to Algorithms

- One operation doesn't reduce a mesh!

- Use greedy algorithm: *iterative edge contractions*
  - Rank each possible edge contraction
  - Contract edge that introduces the least error
  - Repeat until done

- Does NOT produce optimal meshes
  - An optimal mesh has lowest error
  - But optimal mesh computation is intractable (NP-hard)

Summary

- LODs used extensively in interactive applications

- Other work
  - Progressive meshes

- But, can handle only single objects

- What about trees?