Level of Detail: View-Dependent Simplification

Daniel G. Aliaga
(slides based on those of David Luebke @ NVidia)

View-Dependent LOD: Algorithms

- Many good published algorithms:
  - Progressive Meshes
  - Hierarchical Dynamic Simplification
  - Multitriangulation
  - Others...

Overview: The VDS Algorithm

- Overview of the VDS algorithm:
  - A preprocess builds the vertex hierarchy, a hierarchical clustering of vertices
  - At run time, clusters appear to grow and shrink as the viewpoint moves
  - Clusters that become too small are collapsed, filtering out some triangles

Data Structures

- The vertex tree
  - Represents the entire model
  - Hierarchy of all vertices in model
  - Queried each frame for updated scene

- The active triangle list
  - Represents the current simplification
  - List of triangles to be displayed
  - Triangles added and deleted by operations on vertex tree

The Vertex Tree: Folding And Unfolding

- Folding a node collapses its vertices to the proxy
- Unfolding the node splits the proxy back into vertices

Vertex Tree Example

Triangles in active list  Vertex hierarchy
The Vertex Tree

- At runtime, folds and unfolds create a cut or boundary across the vertex tree:

  ![Diagram showing high detail and low detail parts of a model]

  This part of the model is represented at high detail.
  This part in low detail.

View-Dependent Simplification

- Any run-time criterion for folding and unfolding nodes may be used.
- Examples of view-dependent simplification criteria:
  - Screenspace error threshold
  - Silhouette preservation
  - Triangle budget simplification
  - Gaze-directed perceptual simplification

Screen Space Error Threshold

- Nodes chosen by projected area
- User sets screenspace size threshold
- Nodes which grow larger than threshold are unfolded

Silhouette Preservation

- Retain more detail near silhouettes
- A silhouette node supports triangles on the visual contour
- Use tighter screenspace thresholds when examining silhouette nodes

Triangle Budget Simplification

- Minimize error within specified number of triangles
- Sort nodes by screenspace error
- Unfold node with greatest error, putting children into sorted list
  Repeat until budget is reached

Asynchronous Simplification

- Algorithm partitions into two tasks:
  - Simplify Task
  - Render Task
  - Run them in parallel

Active Triangle List
Asynchronous Simplification

- If $S =$ time to simplify, $R =$ time to render:
  - Single process $= (S + R)$
  - Pipelined $= \max(S, R)$
  - Asynchronous $= R$

- The goal: efficient utilization of GPU/CPU

Temporal Coherence

- Exploit the fact that frame-to-frame changes are small
- Three examples:
  - Active triangle list
  - Vertex tree

Exploiting Temporal Coherence

- Active triangle list
  - Could calculate active triangles every frame
  - But...few triangles are added or deleted each frame
  - Idea: make only incremental changes to an active triangle list
    - Simple approach: doubly-linked list of triangles
    - Better: maintain coherent arrays with swapping

Exploiting Temporal Coherence

- Vertex Tree
  - Few nodes change per frame
  - Don’t traverse whole tree
  - Do local updates only at boundary nodes

VDSlib

- Implementation: VDSlib
  - A public-domain view-dependent simplification and rendering package
  - Flexible C++ interface lets users:
    - Construct vertex trees for objects or scenes
    - Specify with callbacks how to simplify, cul, and render them
    - Available at http://vdslib.virginia.edu

GLOD

- An easy-to-use library for level of detail in OpenGL
  - LOD generation
  - LOD run-time management
  - View-dependent LOD (using VDSlib)

http://www.cs.jhu.edu/~graphics/GLOD