Progressive Meshes

- Basic simplification
  - \( M^0 \)
  - \( M^1 \)
  - \( M^2 \)
  - \( M^3 \)
  - \( M^4 \)
  - \( M^5 \)
  - \( M^6 \)
  - \( M^7 \)

- Progressive mesh
  - \( M^0 \)
  - \( M^1 \)
  - \( M^2 \)
  - \( M^3 \)
  - \( M^4 \)
  - \( M^5 \)
  - \( M^6 \)
  - \( M^7 \)
Progressive Meshes

- Simplification

\[ M^n \rightarrow M^c \rightarrow M^c \rightarrow M^c \rightarrow M^0 \]

13,133 ver 10,103 ver 8903 ver 503 ver 489 ver

- Refinement

\[ M^n \rightarrow M^c \rightarrow M^c \rightarrow M^c \rightarrow M^0 \]

13,133 ver 10,103 ver 8903 ver 503 ver 489 ver

Definition of vsplit and ecol

- ecol changes
  - Connectivity - \( \{v_u, v_t\} \) is removed
  - Geometry - \( v_t \) and \( v_u \) disappear & \( v_s \) is created (somewhere)
- \( v_s \) is parent of \( v_u \) & \( v_t \)
- \( v_u \) and \( v_t \) are children of \( v_s \)
Progressive Transmission

- Base mesh ($M^0$) transmitted first
- Refinement records transmitted later & mesh reconstructed progressively

View-Dependent Refinement

- Problem:
  - Large parts of rendered models are hidden
    - Hidden faces are view dependent
  - Basic simplification: all faces rendered at same LOD
    - Waste effort rendering hidden faces
**View-Dependent Refinement**

- **Goal:**
  - Generate progressive representation of mesh s.t. *only some* faces are simplified & others are fully detailed

**Selective Refinement**

- Refine only in desired area
**Vertex Hierarchy**

- Generate Parent-Child forest by combining Progressive Mesh representation with Parent-Child relationship

- $M^0$: Set of root vertices (most simplified mesh)

- $M^n$: Set of leaves of the forest (original mesh)
Selective Refinement

- Given vertex hierarchy forest - selective refinement mesh generated by using selective, out-of-order vsplits and ecols operations.

- Current refined/simplified mesh is vertex front in the forest.
Is This Enough?

- Is Selective-Refinement data-structure enough for View-Dependent operations?

Legal Operations

- Face/vertex is active if it exists in current front

**Legal vsplit:**
- $V_s$ is active vertex
- Faces $\{f_{n0}, f_{n1}, f_{n2}, f_{n3}\}$ are all active

**Legal ecol:**
- $V_t$ & $V_u$ are both active
- Faces $\{f_{n0}, f_{n1}, f_{n2}, f_{n3}\}$ are adjacent to $f_l$ & $f_r$
**Data-Structure Implementation**

- To achieve a real-time View-Dependent algorithm need efficient data-structure to maintain vertices & faces information
  - Vertex list array - holds vertices that participate in View-Dependent model
  - Active vertices list - holds current mesh front
  - Faces & active faces lists

**View-Dependent Algorithm**

- Traverse active-vertex-front before each rendering operation
- For each vertex test if vertex should be
  - Refined
  - Simplified
  - Left as it is
- Perform simplification operations (ecol) **only if legal**
- Perform **all** refine operations (vsplit)
  - to make it legal (sometimes) perform additional vsplit operations
Refinement Criteria

- **View-Frustum Criterion:**
  - Each original mesh vertex is center of sphere containing all its neighbors.
  - Vertex considered outside view-frustum if its associated sphere is outside
    \[ a_i v_x + b_i v_y + c_i v_z + d_i < -r_v \]
    \[ \| (a_i, b_i, c_i) \| = 1 \]
  - \((v_x, v_y, v_z)\) - vertex \(v\) position
  - \((a_i, b_i, c_i, d_i)\) for \(i = 1...4\) - frustum faces
  - \(r_v\) - radius of sphere associated with \(v\)
Refinement Criteria

- View-Frustum Criterion:

- Surface Orientation:
  - Each mesh vertex associated with cone in direction of its normal with angle $\alpha_v$
  - Given viewpoint $e$ - it is unnecessary to split $v$ if:
    \[
    (v - e) \cdot \hat{n}_v > 0 \quad \& \quad ((v - e) \cdot \hat{n}_v)^2 > \| v - e \|^2 \sin^2 \alpha_v
    \]
Refinement Criteria

- Screen-Space Geometric error:
  - Refine mesh only if distance between approximated & original surfaces when projected on the screen is larger than screen-space tolerance $\tau$.

Examples

Original  Simplified  Top-View
Examples (cont’)

Original  Simplified  Top-View