Overview

- Simplification using quadric error metrics (QEM) [Garland et al. 1997]
  - Greedy algorithm using local error metrics
- Variational shape approximation [Cohen-Steiner et al. 2004]
  - Approximate error minimization

Simplification using quadric error metrics

- Main idea: simplify mesh using a sequence of vertex pair contractions

Overview

- Valid vertex pairs
- Error quadrics
- Evaluation and results
**Valid vertex pairs**
- Two criteria
  - Edges $(v_1, v_2)$
  - Proximity $\|v_1 - v_2\| < \text{threshold}$
- May join disconnected regions
  - Does not preserve topology if $\varepsilon > 0$
  - Keep track of valid pairs during contractions
    - Valid pairs of $\tilde{v}$ is union of valid pairs of $v_1, v_2$
    - Replace $v_1, v_2$ by $\tilde{v}$ in all other valid pairs

**Error quadrics**
- Idea: error of a new vertex is distance to a set of planes that locally describe the surface
- A plane is represented by vector $p = (n_x, n_y, n_z, d)^T$
- Squared distance of point $v$ to plane is squared dot product
  $$\Delta_p(v) = (p^T v)^2$$
- In matrix form
  $$\Delta_p(v) = v^T(pp^T)v = v^TK_pv, \quad K \in \mathbb{R}^{4 \times 4}$$

**Error quadrics**
- Error of vertex to a set of planes is sum of error for each plane
  $$\Delta(v) = \sum_{p \in \mathcal{P}(v)} v^T K_p v = v^T \left( \sum_{p \in \mathcal{P}(v)} K_p \right) v = v^T Q v$$
- Matrix $Q$ represents a quadric surface
- Visualization of iso-surfaces $\Delta(v) - \varepsilon$ of error quadrics [Garland et al.]
  - Set of neighboring planes $\mathcal{P}(v)$ are planes adjacent to vertex $v$

**Cost for contraction pair**
- Initial error quadric for each vertex is computed using triangles adjacent to vertex $i$
- $p_l(v)$ - planes given by triangles adjacent to $v$
- Error for contraction pair is smallest possible error for any new vertex with respect to sets of planes of both vertices in pair
  - Given quadrics $Q_1, Q_2$ for $v_1, v_2$
  - Error for contraction pair is
    $$\min v^T(Q_1 + Q_2)v$$
- Update error quadric after contraction
  - Instead of keeping track of sets of planes, simply add up error quadrics $Q_1, Q_2$ of $v_1, v_2$
  - Some planes are double counted

**Results**
- 5804 faces → 994 faces → 532 faces → 248 faces → 64 faces
- 69451 faces → 1000 faces → 100 faces

[Garland et al.]
Evaluation

- Need error metric to evaluate quality of simplified shape
- Garland et al. propose to measure average squared distance between input and simplified shape

\[ E = \frac{1}{|X_0| + |X_i|} \left( \sum_{i \in X_0} d^2(x_i, M_i) + \sum_{i \in X_i} d^2(x_i, M_i) \right) \]

- Sample points on input, simplified shape \( X_0, X_i \)
- Minimum distance between point on one shape and the other shape

\[ d(v, M) = \min_{p \in M} \| v - p \| \]

Evaluation

- Effect of optimal vertex placement

<table>
<thead>
<tr>
<th>Faces</th>
<th>Final</th>
<th>Optimal</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.0062</td>
<td>0.0054</td>
<td>13.4%</td>
</tr>
<tr>
<td>100</td>
<td>0.00032</td>
<td>0.00025</td>
<td>21.7%</td>
</tr>
<tr>
<td>500</td>
<td>2.4e-05</td>
<td>1.3e-06</td>
<td>47.6%</td>
</tr>
<tr>
<td>1000</td>
<td>5.7e-06</td>
<td>3.4e-06</td>
<td>40.3%</td>
</tr>
<tr>
<td>2000</td>
<td>1.2e-06</td>
<td>7.9e-07</td>
<td>32.4%</td>
</tr>
<tr>
<td>3000</td>
<td>3.6e-07</td>
<td>2.6e-07</td>
<td>28.2%</td>
</tr>
</tbody>
</table>

- Instead of greedy approach, wouldn’t it be better to directly minimize a geometric error?

Overview

- Simplification using quadric error metrics (QEM) [Garland et al. 1997]
  - Greedy algorithm using local error metrics
- Variational shape approximation [Cohen-Steiner et al. 2004]
  - Approximate error minimization

Variational shape approximation

- Idea
  - Given a number of approximation primitives, and an error metric, find best shape approximation that minimizes error
  - Ignore topology
- Need
  - Approximation primitives
  - Error metric
  - Optimization algorithm

Approach

- Partition input geometry into given number of parts, or regions
- Use planar proxies as approximation primitives for each region
- Define metric that measures error between regions and their proxies
- Use optimization algorithm to find best partition and proxies

Partition into regions
Proxies, in one-to-one correspondence with regions

[Cohen-Steiner et al.]
**Error metrics**

- Measure error between cluster and proxy
- Squared error between positions
  \[ L^2(R, P_i) = \int_{R_i} ||x - \Pi(x)||^2 dx \]
  - Cluster surface \( R_i \)
  - Projection onto planar proxy \( \Pi_i(x) \)
- Problems
  - Best proxy is not unique for hyperbolic neighborhoods (saddle points)
  - Bad for subsequent optimization step

**Normal based metric**

- Normal of proxy \( n_i \)
- Best proxy is unique for all types of infinitesimal neighborhoods
- Best proxy normal is simply the average of normals in cluster

\[ L^2(R, P_i) = \int_{R_i} ||n(x) - n_i||^2 dx \]

**Optimization algorithm**

**Problem statement**

Given an error metric, a desired number of proxies, find the optimal set of regions and associated proxies that minimize the error

- Input geometry uses triangles, regions are clusters of triangles
  - Clustering problem
- Use Lloyd’s clustering algorithm
  - Also used to approximate k-means clustering

**Lloyd clustering**

- Iteration over two phases
  - Geometry partitioning
  - Proxy fitting
- “Pseudocode”
  - Input parameter: \( k \) proxies/regions

Fit \( k \) initial proxies

Compute initial geometry partitioning

While error is reduced

  - Update proxy fitting
  - Update geometry partitioning

**Fit initial proxies**

- Randomly pick \( k \) triangles in input mesh
- One proxy \( P_i \) assigned to each triangle
  - Proxy defined by barycenter and normal of triangle

**Geometry partitioning**

**Distortion minimizing flooding**

- For each seed triangle, add three neighboring triangles \( T_j \) to priority queue
  - Priority value in queue is error between triangle and proxy \( E(T_j, P_i) \)
  - Entry in queue is tagged with proxy index \( i \)
- Each triangle can appear in queue up to three times
  - Different proxy tags and priority values
- Iteratively pop triangles from queue until empty
  - If triangle has already been popped before, do nothing
  - Otherwise, label triangle with proxy tag; add up to two neighboring triangles to queue, use current proxy tag
- When queue is empty
  - Each triangle has a proxy tag
  - Regions for each proxy are connected and non-overlapping
**Proxy fitting**

- $L^2$ metric
  - Proxy is least squares fitting plane to triangles in region associated with proxy
- $L^{2,1}$ metric
  - Normal of proxy is area weighted average of normals
  - Position of proxy plane is irrelevant, pick barycenter region associated with proxy

---

**Update geometry partitioning**

- Given proxies, find new seed triangles
  - Triangles that have smallest error with respect to their current proxy
- Restart distortion minimizing flooding with new seeds
- Iterate

---

**Partitioning and proxies**

[Image: Partitioning and Proxies]

- [Cohen-Steiner et al.]

---

**Convergence**

- Global convergence not guaranteed
- Well behaved in practice
  - Only small changes after a few iterations

[Graph: Error vs Iteration]

---

**Extensions**

- Region teleportation
- Farthest point initialization
- Meshing

---

**Region teleportation**

- Avoid getting stuck in local minima
- At regular intervals during iteration
  - Look at all pairs of neighboring regions
  - Determine which pair, if merged into single region, leads to smallest increase in error
  - Merge that pair
  - Add new region, use triangle with largest error as seed triangle

---

[Image: Graph: Error vs Iteration]
### Farthest point initialization
- Add regions one by one
- Use triangle with largest error as seed triangle for each new region

---

### Meshing
- So far, have only planar proxies and associated regions, but no simplified triangle mesh
- Several steps
  - Generate vertices
  - Triangulation of vertices
  - Construction of simplified polygonal mesh from triangle mesh

---

### Generating vertices
- Each original vertex that borders three or more different regions becomes a new anchor vertex
- Position of anchor vertex is average of projection of original vertex onto proxies
- Construct edges between anchor vertices by following region boundaries on original mesh
- Refine edges by adding more anchor vertices if necessary

---

### Triangulation
- Flood fill original mesh vertices by assigning shortest distance to next anchor
  - Label each original vertex with closest anchor
- Each triangle that has vertices labeled with three different anchors gives rise to a triangle connecting the three anchors

---

### Simplification
- Greedily merge triangles to quads
- Greedily remove edges to form larger polygons
- Quads and polygons are not planar in general

---

### Results
- 346k triangles to 300 proxies
- 100k triangles to 50 proxies
Comparison
- Comparison to simplification using quadric error metrics
- Same number of edges

Comparison
- Comparison of Hausdorff distance to simplification using quadric error metrics [Garland et al.]

Summary
- Greedy algorithm using iterative vertex contractions
- Define local error to prioritize vertex pairs for contraction
- Simple rule to update error after contraction

Variational shape approximation
- Formulate problem as error minimization problem (variational problem)
  - Representation of simplified shape
  - Error metrics
  - Approximate error minimization algorithm
- Separate meshing step

Discussion
- Variational shape approximation nicely captures anisotropy
- Hausdorff error for given number of vertices is usually lower than QEM
- Slower than QEM
- For same number of triangles, error often lower with QEM
  - Variational approximation is not driven by per triangle error

Further topics
- Out-of-core simplification
- Level-of-detail rendering

Out-of-core simplification
- Input geometry may not fit into main memory
  - Need algorithm that doesn’t require loading full input geometry
- (Random) example: Out-of-Core Simplification of Large Polygonal Models, Lindstrom SIGGRAPH 2000
  - Based on spatial grid of cells
  - In-core memory consumption limited by grid resolution
  - Read input triangles one by one, accumulate error quadrics in cells
  - Use error quadrics to determine vertex position for each cell
## Level of detail rendering

- Idea: adjust simplification of geometry to given viewing configuration
  - Increase/regulate framerate
- Usually precomputed discrete or continuous levels of simplification
- View-dependent, on-the-fly rendering of appropriate level
- (Random) examples
  - Geometry clipmaps: terrain rendering using nested regular grids, Lossasso et al. 2004
  - View dependent progressive meshes, Hoppe et al. 1997

## Next time

- Parameterization