Simplification of 3D Meshes

Addy Ngan

2/24/00

Outline

- Motivation
- Taxonomy of simplification methods
- Hoppe et al, Mesh optimization
- Hoppe, Progressive meshes

Motivation

- High detailed meshes becoming widely available in computer vision, scientific visualization, terrain data from satellite… etc.
- Need to store, transmit, analyze, edit and display them efficiently.

> 1M triangles!

Goals

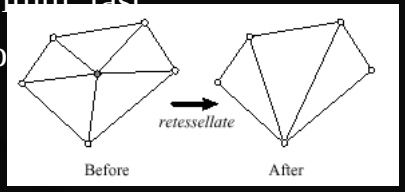
- Reduce number of polygons
  - Faster rendering
  - Less storage
  - Simpler manipulation
- Other qualities
  - General (non-manifold)
  - Efficiency
  - Preserve attributes other than geometry

Taxonomy of methods

- Manifold Simplification
  - Vertex decimation
  - Wavelet
  - Edge collapse
- Non-manifold Simplification
  - Vertex clustering

Mesh Decimation [Schroeder et al 92]

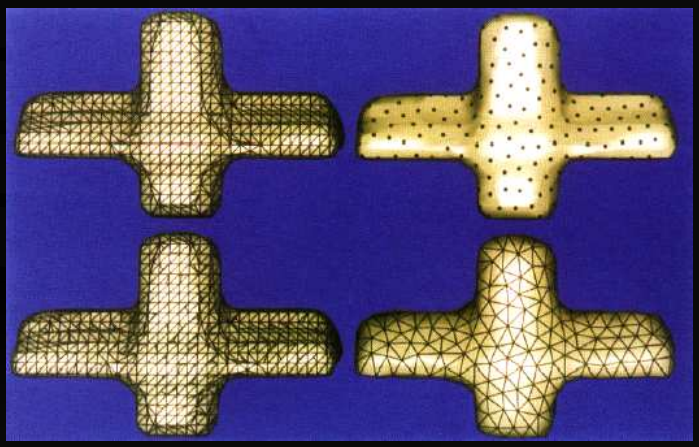
- Multiple passes
- For each pass remove all vertices that matches certain criteria and retriangulate
- Criteria: distance to plane/edge
- Advantage: simple algorithm, fast
- Disadvantage: big memo



Re-Tiling [Turk 92]

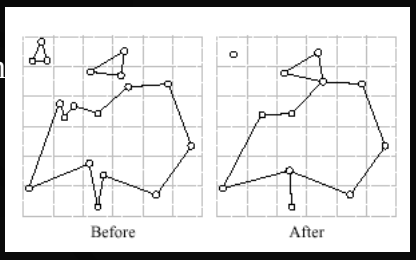
- Steps:
  - Generate random points on surface
  - Iterative repulsion spread the points out uniformly
  - Add new set of points to the surface and *mutual tessellate*
  - Remove old vertices one by one yielding a new triangulation
- Variant to put points adaptively depending on curvature
- Advantage : maintain topology
- Disadvantage : complex algorithm, blur sharp features.

Re-Tiling (con't)



3D Grid Method [Rossignac-Borrel 93]

- Steps:
  - Subdivide the bounding volume into regular grid
  - Merge all vertices within each cell together into a new vertex
  - Form triangles accordin
- Advantage: very general, fast
- Disadvantage : low quality, non-adaptive

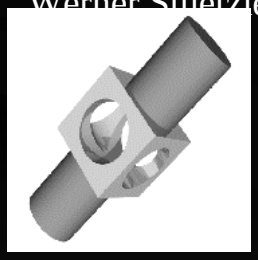


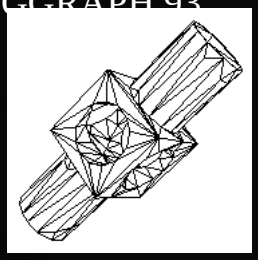
Optimization [Hoppe et al 93]

- Optimization based simplification. Minimize energy function.
- Repeat semi-random changes to topology and optimize the geometry.
- Work on manifold
- Advantages: High quality, less sensitive to noise
- Disadvantages: slow

Mesh Optimization

Hugues Hoppe, Tony DeRose,
Tom Duchamp, John McDonald,
Werner Stuetzle, SIGGRAPH 93





Optimization

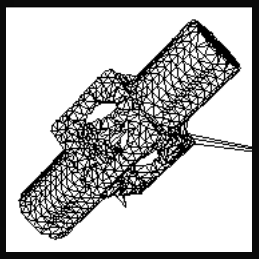
- Similar problem to simplification
- Starting with sample data points from the surface and a initial mesh, find a simpler mesh
- Minimize energy function that describes the conciseness and accuracy of the mesh

Mesh Representation

- A mesh is represented by a pair (K,V)
  - K is a simplicial complex representing the connectivity of the vertices/edges/faces.
  - $V=\{\mathbf{v}_1,\ldots,\mathbf{v}_m\}$ is a set of m vertex positions defining the shape of the mesh in $\mathbf{R}^3$
- If mesh is not self-intersecting, every point can be represented by a barycentric vector

Energy Function

- $E(K,V) = E_{dist}(K,V) + E_{rep}(K) + E_{spring}(K,V)$
- $E_{dist}(K,V) = \sum_{i=1}^{n} d^2(\mathbf{x}_i, \varphi_V(|K|))$
- $E_{rep}(K) = c_{rep} m$
- $E_{spring}(K,V) = \sum_{\{j,k\} \in K} \kappa \left\| \mathbf{v}_j - \mathbf{v}_k \right\|^2$



Without E_{spring}, spikes possible for region with no data points

Minimizing the Energy Function

- A bigger $c_{rep} \rightarrow$ sparser representation.
- Outer loop optimize over K (Discrete)
- Inner loop optimize over V for a fixed K (Continuous)

Optimization for fixed K

- = Minimize $\sum_{i=1}^{n} d^2(\mathbf{x}_i, \varphi_V(|K|)) + \sum_{\{j,k\} \in K} \kappa \left\| \mathbf{v}_j - \mathbf{v}_k \right\|^2$
- For each $\mathbf{x}_i$, distance = $\min_{\mathbf{b}_i \in |K|} \left\| \mathbf{x}_i - \varphi_V(\mathbf{b}_i) \right\|^2$
- New objective function :
  - E(K,V,B)= $\sum_{i=1}^{n} \min_{\mathbf{b}_i \in |K|} \left\| \mathbf{x}_i - \varphi_V(\mathbf{b}_i) \right\|^2 + \sum_{\{j,k\} \in K} \kappa \left\| \mathbf{v}_j - \mathbf{v}_k \right\|^2$

Optimization over fixed K

- Two subproblems :
  - For fixed vertices V, find optimal barycentric coordinate vectors B by projection
  - For fixed B, find optimal V by solving a linear least squares problem
- Find optimal solutions to both subproblems, so E(K,V,B) must converge.

Projection Subproblem

- Find optimal B by projecting $\mathbf{x}_i$ onto the mesh.
- To accelerate, build a spatial partitioning data structure so that for each point only consider nearby subset of faces
- Assume a point's projection lies in the neighborhood of its projection in previous iteration (perform well in practice)

Linear Least Squares Subproblem

- Minimize for x,y,z coordinates, $\|A\mathbf{v} - \mathbf{d}\|^2$
  - $\mathbf{v}$ m-vector
  - A (n+e)×m matrix
  - $\mathbf{d}$ (n+e)-vector

  For m mesh vertices, n data points, e edges
  - First *n* rows of $\boldsymbol{d}$ contains the n data points
  - Next *e* rows of $\boldsymbol{d}$ are zeroes
  - $\mathbf{v}$ contains the m mesh vertices

Linear Least Squares (con't)

$\|A\mathbf{v}_j - \mathbf{d}_j\|^2$

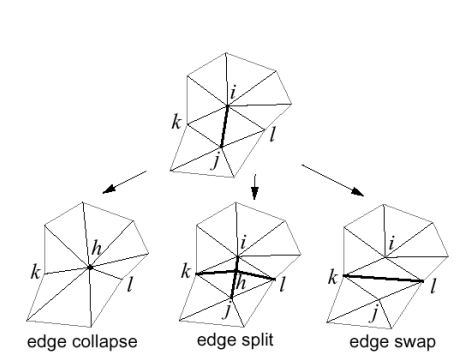
- First *n* rows of *A* contains the barycentric coordinates computed in projection (at most 3 non-zero entries)
- Next *e* rows represent the spring energy: each contains an $\sqrt{\kappa}$ and $-\sqrt{\kappa}$ entries in columns corresponding to indices of edge's endpoints (exactly 2 non-zero entries)

- A is sparse. Use "conjugate gradient method" to solve in O(n+m) time

Optimization over K

- Take a legal move and accept if it gives lower energy. Terminate if a number of trials failed to give a lower energy.
- Legal move : application of one of these that leaves the topological type of K unchanged
- Three elementary transformations:
  edge collapse/split/swap

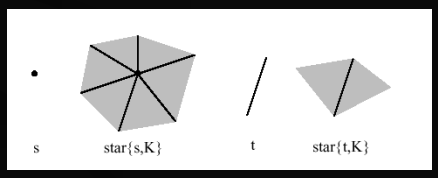
Legal move



- Split – Always legal
- Collapse : if and only if…
  - For all vertices {k} adjacent to both {i} and {j}, {i,j,k} is a face
  - If {i} and {j} are both boundary vertices, {i,j} is a boundary edge
  - K has more than 4 vertices if {i} and {j} both are not boundary vertices, or K has more than 3 vertices if either {i} or {j} are boundary vertices
- Swap – if and only if $\{k,l\} \notin K$
- Proof in Hoppe et al, TR 93-01-01 University of Washington

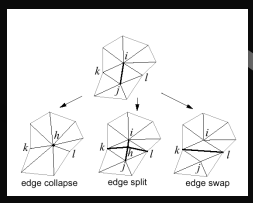
Evaluation of a legal move

- Instead of re-computing the global energy after a legal move, only compute the change in energy for the local submesh
- For s ∈ K, define star(s;K) = {s' ∈ K : s non-empty subset of s'}



Edge Collapse

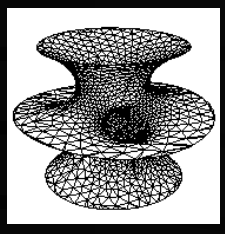
- To evaluate collapsing of an edge $\{i,j\}$, take the submesh to be star($\{i\}$;K)∪ star($\{j\}$;K). Optimize over the new vertex h while holding all other constant
- Attempt optimizations starting at $\mathbf{v}_i$, $\mathbf{v}_j$, and ½($\mathbf{v}_i$+$\mathbf{v}_j$). Accept the best one.
- Instead of checking for self-intersection after collapse which is expensive, use a threshold value for the maximum dihedral angle of edges in star($\{h\}$;K')



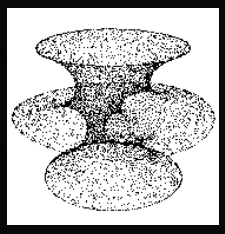
Edge split/swap

- Edge split –
  - Same procedure as collapse, using the submesh to be star($\{i,j\}$; K). Initial value of $\mathbf{v}_h$ be ½($\mathbf{v}_i$+$\mathbf{v}_j$).
- Edge swap-
  - For a swap that replace an edge $\{i,j\}$ with $\{k,l\}$, choose the best of the two optimizations, one with submesh star($\{k\}$;K') varying vertex $\mathbf{v}_k$, another with submesh star($\{l\}$;K') varying vertex $\mathbf{v}_l$.

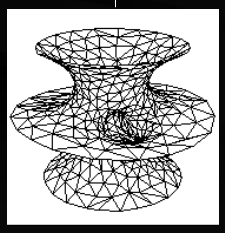
Results



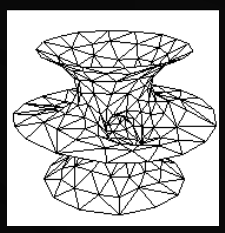
Initial mesh
(2032 vertices)



Sample Points
(6752 vertices)

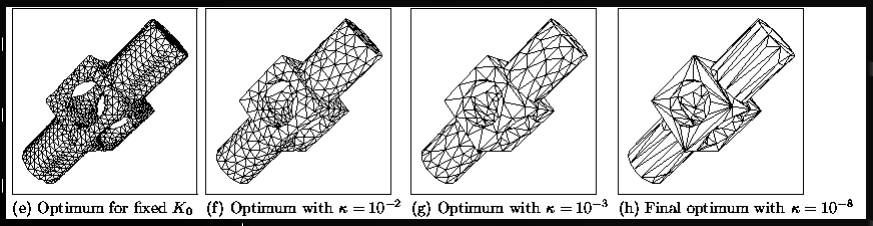


$c_{rep}=10^{-5}$
(487 vertices)



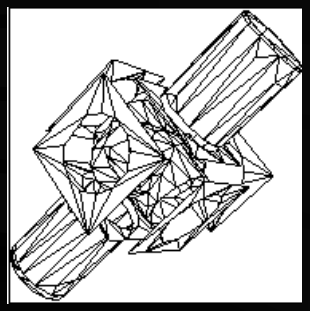
$c_{rep}=10^{-4}$
(239 vertices)

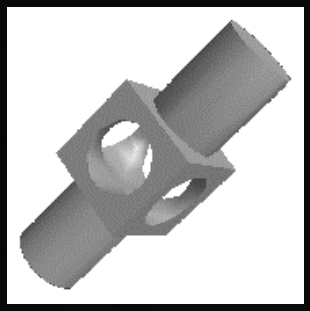
Results



(e) Optimum for fixed K_0 (f) Optimum with $\kappa = 10^{-2}$ (g) Optimum with $\kappa = 10^{-3}$ (h) Final optimum with $\kappa = 10^{-8}$

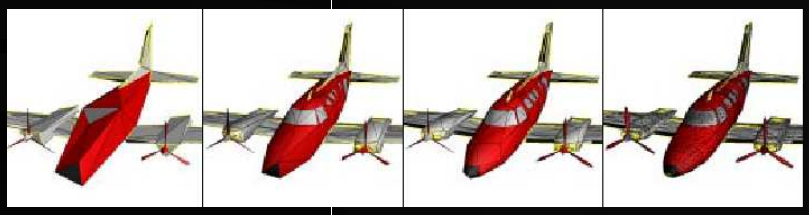
Results - Segmentation





Progressive Meshes

Hugues Hoppe
SIGGRAPH 96



Additional Goal

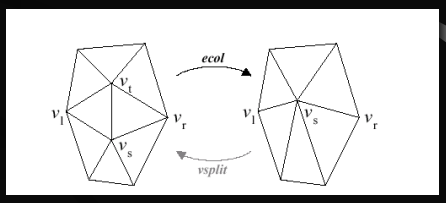
- Lossless, continuous-resolution, progressive representation
- More compact representation
- Apart from geometry, also preserve other attributes (colors, normals, materials, texture coordinates…)

Mesh Representation

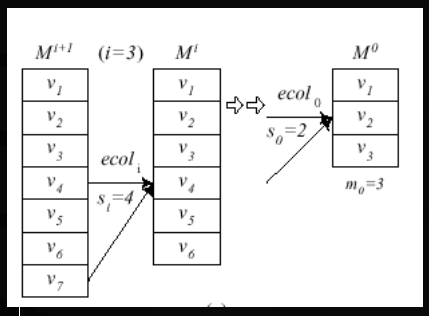
- Edge collapse only is sufficient!
- An initial mesh $\hat{M} = M^n$
  can be simplified into a coarser mesh M^0 by a sequence of edge collapse.

$$(\hat{M} = M^n) \xrightarrow{ecol_{n-1}} \ldots \xrightarrow{ecol_1} M^1 \xrightarrow{ecol_{n-1}} M^0$$

- The sequence is invertible



Progressive Mesh



$$(\hat{M} = M^n) \xrightarrow{ecol_{n-1}} \ldots \xrightarrow{ecol_1} M^1 \xrightarrow{ecol_{n-1}} M^0$$

Energy Function

- Old
  $E(K,V) = E_{dist}(K,V) + E_{rep}(K) + E_{spring}(K,V)$
- New
  $E(M) = E_{dist}(M) + E_{spring}(M) + E_{scalar}(M) + E_{discrete}(M)$ where $M = (K,V,D,S)$
- E_{rep} goes away! And so does the parameter c_{rep}

Preserving geometry

- $E_{dist} + E_{spring}$
- Same as before, but only consider edge collapse in the outer loop.
- The possible legal collapses are placed in a priority queue with its estimated energy change ΔE
- For an edge collapse $K \rightarrow K'$, $\Delta E = E_{K'} - E_K$
- After each collapse, the cost of its neighborhood edges in the priority queue are updated.

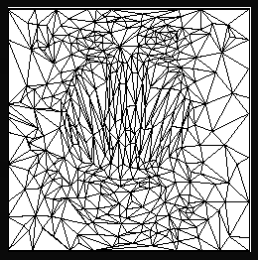
Preserving Scalar attributes

- For *d* scalar attributes, we could have added *d* dimension to E_{dist}.
- For efficiency purpose, we separate it as a different term $E_{scalar}(\underline{V}) = (c_{scalar})^2 \sum_{i=1}^{n} \left\| \underline{\mathbf{x}}_i - \varphi_{\underline{V}}(\mathbf{b}_i) \right\|^2$
- First solve for the vertex positions. Then using the same $\mathbf{b}_i$ to find vertex attributes minimizing E_{scalar} by linear least square

Preserving Scalar Attributes

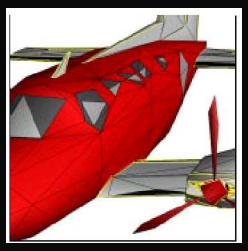
200x200 vertices

400 vertices



Preserving discontinuity

- We want to preserve discontinuity because they often form noticeable features
- sample an additional set of points $\mathbf{X}_{disc}$ from sharp edges of initial mesh. Compute E_{disc} by projecting X_{disc} onto the corresponding sharp edges
- disallow/penalize collapse of boundary and discontinuity edges



Without E_{disc}

References

- Survey of Polygonal Surface Simplification Algorithms, Paul Heckbert and Michael Garland, tech. report, CS Dept.,Carnegie Mellon U., to appear
- H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, Mesh Optimization, SIGGRAPH 93, 19-26.
- Hugues Hoppe, Progressive Meshes, SIGGRAPH 96, 99-108.
- Hugues Hoppe, Efficient implementation of progressive meshes, Computers & Graphics, Vol. 22, No. 1, 1998, pages 27-36.
- Greg Turk, Re-Tiling Polygonal Surfaces, SIGGRAPH 92, 55-64.
- William J. Schroeder, Jonathan Zarge, and William Lorensen, Decimation of Triangle Meshes, SIGGRAPH 92, 65-70.
- J. Rossignac and P. Borrel, Multi-resolution 3D approximations for rendering complex scenes