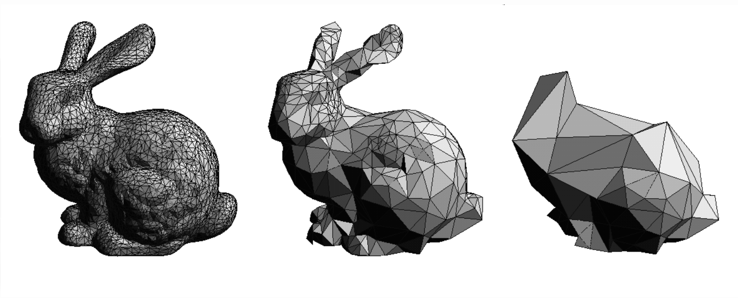
Simplification

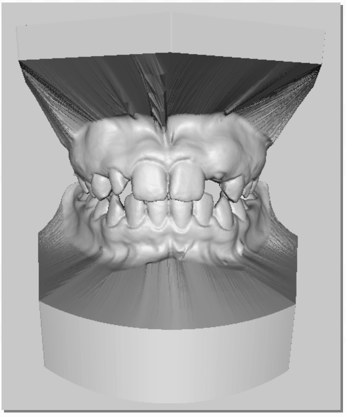


Stolen from various places

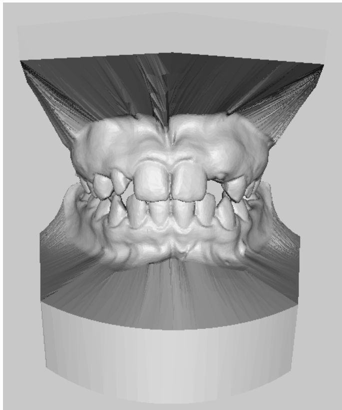
"The Problem of Detail"

- Graphics systems are awash in model data:
  - highly detailed CAD models
  - high-precision surface scans
  - surface reconstruction algorithms
- Available resources are always constrained:
  - CPU, space, graphics speed, network bandwidth
- We need economical models:
  - minimum level of detail (LOD) required

Non-Economical Model



424,376 faces



60,000 faces

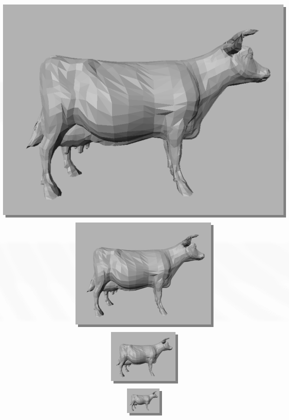
Motivation

Simplification is an active field of research in various domains:

- Cartography: large scale maps generalization
- Computer vision: large range data processing
- Computer graphics: real time rendering, compression, progressive transmission

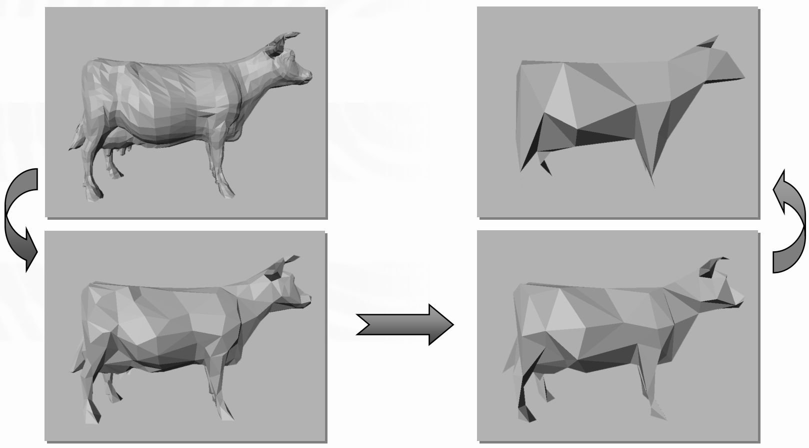
<u>We focus on polygonal (triangular) surface simplification</u>

Single Resolution is Not Enough



- Models used in variety of contexts
- Context dictates required detail ↔ LOD should vary with context
- Context changes dynamically
- With what level of coherence?
  - generally high coherence in view

Surface Simplification



Triangular meshes

- Common:
  - widely supported in hardware
  - near-universal support in software packages
  - output of most scanning systems
  - pragmatic
- Flexible
- Switching representations:
  - many applications to convert to and from triangular mesh surface

Surface Simplification Goal

Goal: produce approximations with fewer triangles:
- should be as similar as possible to original
- computationally efficient process

Need criteria for assessing model similarity (some error metric):
- similarity of appearance: for display is the ultimate goal
- similarity of shape:
  - generally easier to compute
  - lends itself to more applications other than display

Simplification Criteria

- Size in scene
- Complexity:
  - Geometric (curvature etc.)
  - Attributes (color, derivatives)

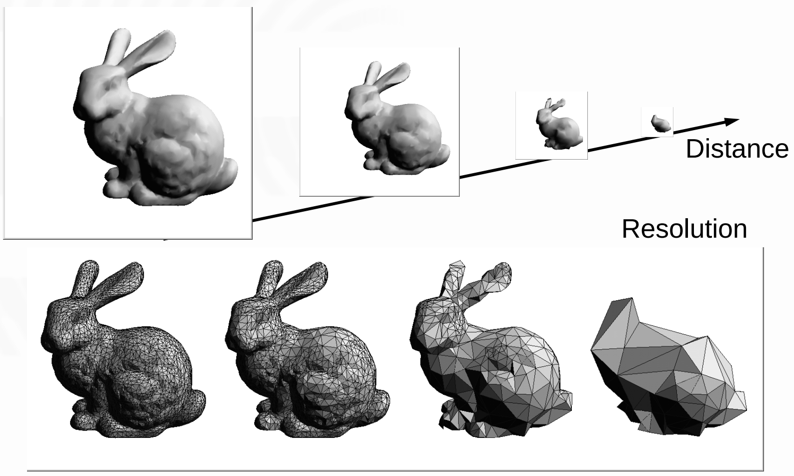
View-independent
Off-line

- Distance
- Screen size
- Visibility/Illumination
- Transmission time

View-dependent
On-line

<u>We would like to use models that can change their complexity and adapt according to different parameters.</u>

Distance & Resolution



Point of View

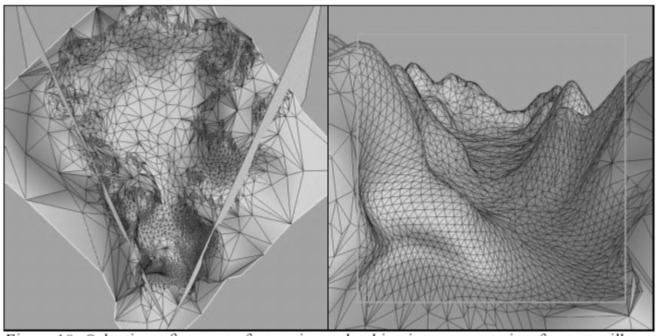
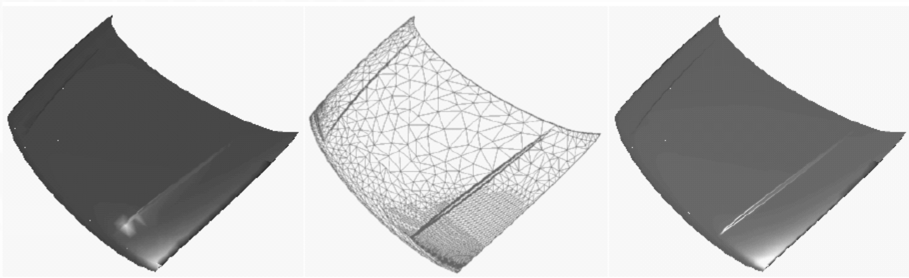


Figure 10: Selective refinement of a terrain mesh taking into account view frustum, silhouette regions, and projected screen size of faces (7,438 faces).

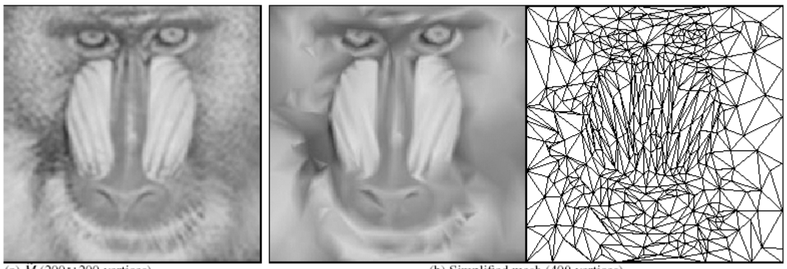
Illumination & Silhouette

Low resolution

Adaptive resolution

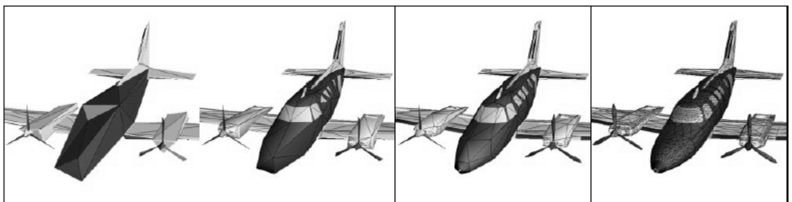


Color Attributes



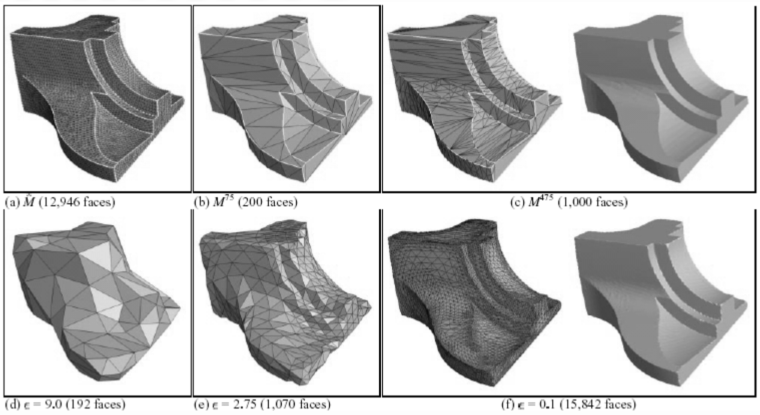
(a) M (200×200 vertices) (b) Simplified mesh (400 vertices)

Transmission…



(a) Base mesh M^0 (150 faces) (b) Mesh M^{175} (500 faces) (c) Mesh M^{425} (1,000 faces) (d) Original $\hat{M} = M^n$ (13,546 faces)

Constrained



Multiresolution Models

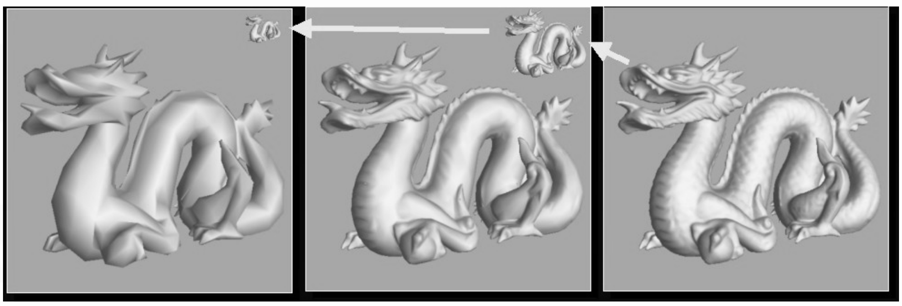
- Encode wide range of levels of detail
  - extract appropriate approximations
    - can be generated via simplification process
  - must have low overhead
    - space consumed by representation
    - cost of changing level of detail while rendering
  - must support the reconstruction of a wide range of levels of details to accommodate a wide range of viewing contexts

- Image pyramids (mip-maps) a good example
  - very successful technique for raster images

Discrete Multiresolution Models

- Given a model, build a discrete set of approximations (offline)
  - can be produced by any simplification system
  - at run time, simply select which to render using a threshold parameter
- Inter-frame switching causes “popping”
  - smooth transition with image blending (cross-disolve)
  - use geometry blending: geomorphing [Hoppe]
- Supported by several software packages: RenderMan, Open Inventor, IRIS Performer, ...

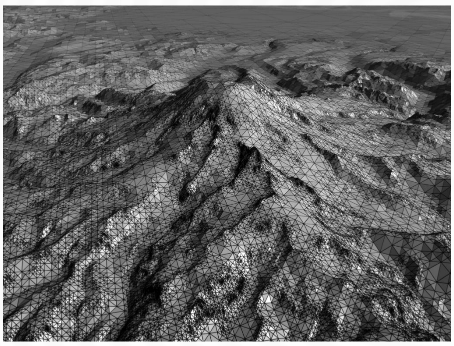
Discrete Multiresolution



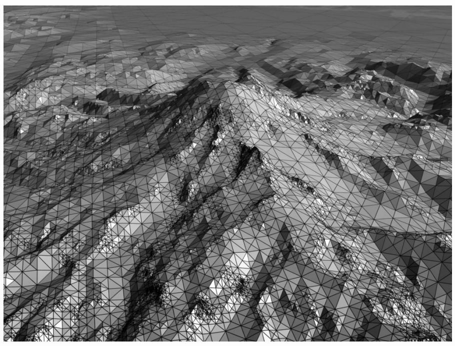
Limits of Discrete Models

- We may need varying LOD over surface
  - E.g.: large surface, oblique view (e.g. on terrain)
    - need high detail near the viewer
    - need less detail far away
  - single LOD will be inappropriate
    - either excessively detailed in the distance (wasteful)
    - or insufficiently detailed near viewer (visual artifacts)
- Doesn’t really exploit available coherence
  - small view change may cause large model change

Visualization of Terrains Made Easy



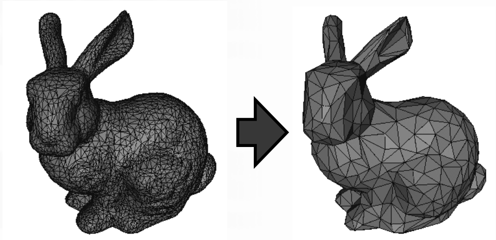
79,382 triangles



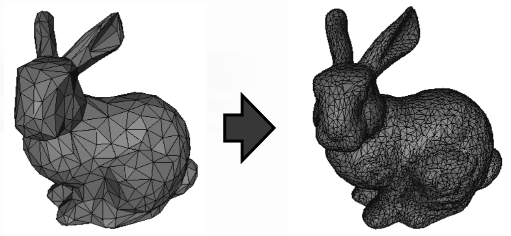
25,100 triangles

Simplification

Bottom up



Top Down



Optimal Approximations

Goal:

- Achieve given error with minimal number of triangles
- Achieve given number of triangles with minimal error

Computationally feasible for curves

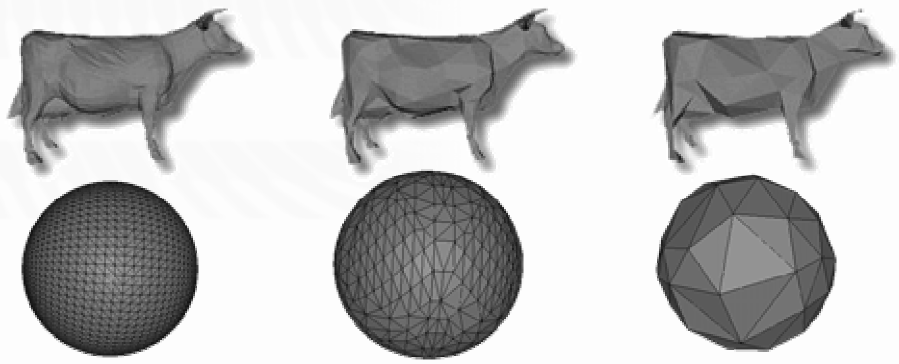
- *O(n)* for functions of one variable
- but $O(n^2 \log n)$ for plane curves

Intractable for surfaces

- NP-hard to find optimal height field [Agarwal–Suri 94]
- must also be the case for manifold surfaces

Level of Details (LOD)

Each model is in fact a set of models in different level of details. LOD usually means a discrete set of different models:



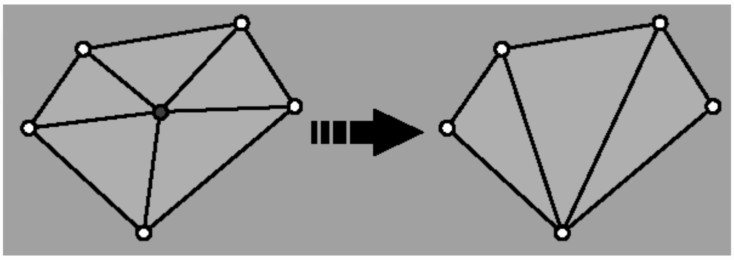
Decimation primitives

Concentrate locally on part of the mesh and reduce its complexity by a decimation operation (primitive):

- Involve near neighbors. Only a small *patch* of the mesh is affected in each operation
- Each operation introduces error
- Apply operation which introduces the least error
- Incremental

Vertex Removal

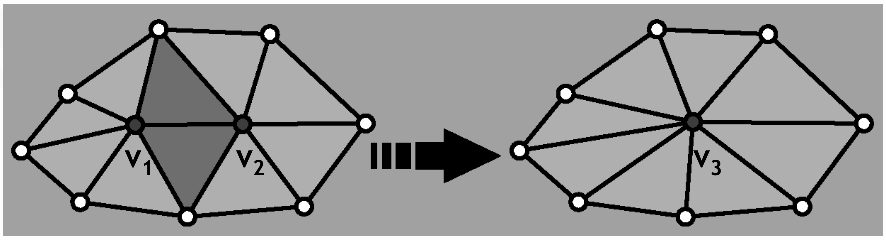
- Starting with original model, iteratively:
  - rank vertices according to their importance
  - select unimportant vertex, remove it, and its edges.
  - retriangulate hole
- Remaining vertices are subset of the original set
- A fairly common technique
- Schroeder *et al*, Soucy *et al.*, Klein *et al.*



Edge Contraction (Edge Collapse)

- <u>Edge Contraction</u>: two vertices are replaced with one new vertex removing one edge and two triangles.

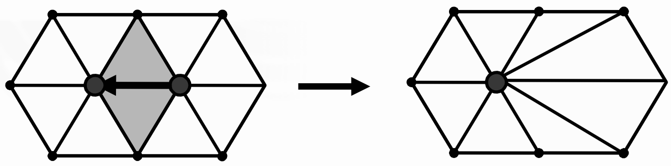
- General edge contraction $(v_1,v_2) \rightarrow v'$ is performed by
  - moving v_1 and v_2 to position v'
  - replacing all occurrences of v_2 with v_1
  - removing v_2 and all degenerate triangles



Half Edge Contraction

Issue: where should we put the new vertex when contracting?

Solution: just use one of the two vertices and remove only one neighbor.



Edge Contraction

Starting with the original model, iteratively

- rank all edges with some cost metric
- contract minimum cost edge
- update edge costs

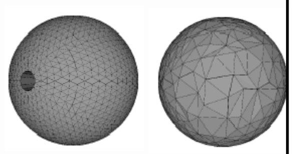
Currently the most popular technique

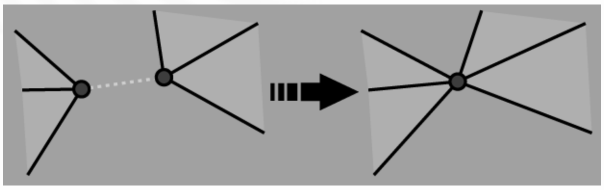
- Hoppe, Garland–Heckbert, Lindstrom-Turk, Ronfard-Rossignac, Guéziec, and several others
- simpler than vertex removal (no re-triangulation)
- well-defined on any mesh

Vertex Pair Contraction

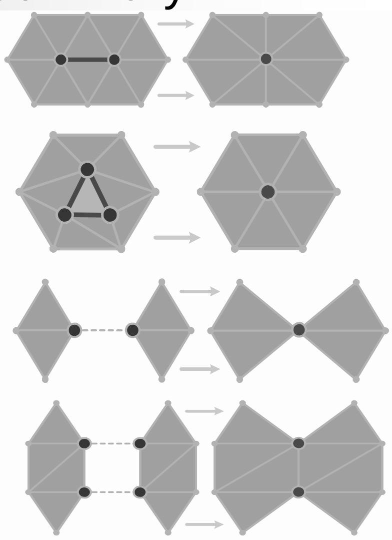
Can also easily contract any pair of vertices

- fundamental operation is exactly the same
- joins previously unconnected areas
- can be used to achieve topological simplification





Contraction Operators Summary



- Edge collapse (v ← v-1, f ← f-2)
- Triangle collapse (v←v-2, f ← f-4)
- Vertex pair contraction
- Vertex cluster contraction (set of vertices)

<u>We want slow simplification!</u>

Outline of Decimation Algorithm

```
While (object not coarse enough)
     choose best decimation
     apply to object
```

How do we measure coarseness?
How do we choose priority?

Error Metrics for Contraction

Used to sort edges during simplification
- reflects amount of geometric error introduced
- main differentiating feature among algorithms

Must address two interrelated problems
- what is the best contraction to perform?
- what is the best position v' for remaining vertex?
  - can just choose one of the endpoints
  - but can often do better by optimizing position of v'

Error Metrics

Define shape distance by:

- Sum
- Max
- Norm L_2, L
- Hausdorff Distance

Error computation

Geometric error:

- Position difference (vertices triangles)
- Volume difference

Attribute errors:

- Normal difference
- Color or function values difference

Error calculation or estimation?

Error Computation (2)

Local error: Compare the new patch with the previous iteration.

- \+ Fast
- \+ Memory-less
- – Accumulates error

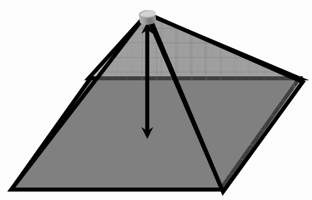
Global error: Compare the new patch with the original mesh.

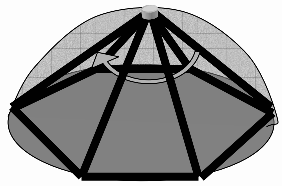
- \+ Better quality control
- – Slow
- – Must remember the original mesh throughout the algorithm

Simplification Error Metrics

Measures:

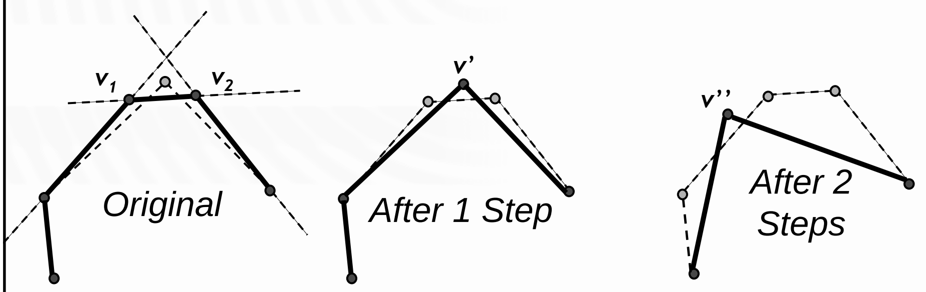
- Edge length
- Distance to plane
- Curvature
- Volume
- Quadric error metrics
  [Garland–Heckbert]





Example: Contraction & “Planes” in 2D

Lines defined by neighboring segments
- Determine position of new vertex
- Accumulate lines for ever larger areas



Measuring Error with Infinite Planes

Why base error on planes?
- Faster, but less accurate than distance-to-face
- Simple linear system for minimum-error position
- Drawback: unlike surface, planes are infinite

Related error metrics
- Ronfard & Rossignac — max vs. sum
- Lindstrom & Turk — similar form; volume-based

Quadric Error

- Each vertex has a (conceptual) set of planes
  - Error ≡ sum of squared distances to planes in set

$$Error(v) = \sum_i \left(n_i^T v + d_i\right)^2 = \sum_i \left(a_i x + b_i y + c_i z + d_i\right)^2$$

- Initialize with planes of incident faces
  - Consequently, all initial errors are 0
- When contracting pair, use plane set union
  - *planes(v') = planes(v_1) ∪ planes(v_2)*

Quadric Error

Given a plane, we can define a *quadric* Q:

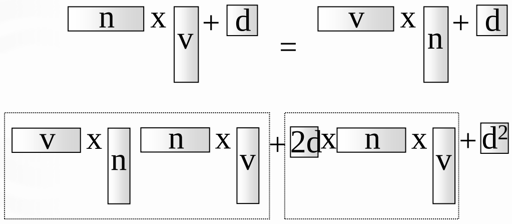
$$Q = (\mathbf{A}, \mathbf{b}, c) = (\mathbf{n}\mathbf{n}^\mathsf{T}, d\mathbf{n}, d^2)$$

Measuring squared distance to the plane is equivalent to:

$$Q(\mathbf{v}) = \mathbf{v}^\mathsf{T}\mathbf{A}\mathbf{v} + 2\mathbf{b}^\mathsf{T}\mathbf{v} + c$$

Quadric Error

$$Q(v) = \left(n^T v + d\right)^2 =$$
$$\left(v^T n + d\right)\left(n^T v + d\right) =$$
$$\left(v^T n n^T v + 2dn^T v + d^2\right) =$$
$$\left(v^T (nn^T) v + 2(dn)^T v + d^2\right)$$



$$Q(\mathbf{v}) = \begin{bmatrix} x & y & z \end{bmatrix} \begin{bmatrix} a^2 & ab & ac \\ ab & b^2 & bc \\ ac & bc & c^2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + 2\begin{bmatrix} ad & bd & cd \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + d^2$$

The Quadric Error Metric

Sum of quadrics represents set of planes

$$\sum_i (\mathbf{n}_i{}^T \mathbf{v} + d_i)^2 = \sum_i Q_i(\mathbf{v}) = \left(\sum_i Q_i\right)(\mathbf{v})$$

$$Q = Q_i + Q_j = (\mathbf{A}_i + \mathbf{A}_j,\ \mathbf{b}_i + \mathbf{b}_j,\ c_i + c_j)$$

Each vertex has <u>one</u> associated quadric

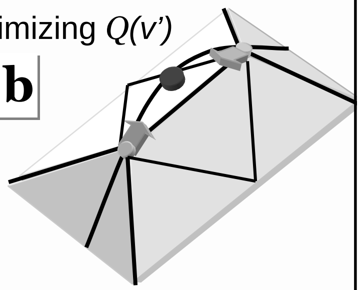
- Error(v_i) = Q_i (v_i)
- Sum quadrics when contracting $(v_i, v_j) \rightarrow v'$
- Cost of contraction is $Q(v')$
- Exclude duplicate planes from the sum

The Quadric Error Metric

Sum of endpoint quadrics determines v'

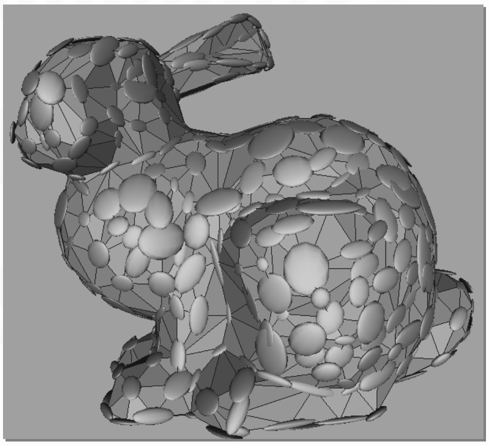
- Optimal placement: choose *v'* minimizing *Q(v')*

$$\nabla Q(\mathbf{v}') = 0 \;\Rightarrow\; \mathbf{v}' = -\mathbf{A}^{-1}\mathbf{b}$$



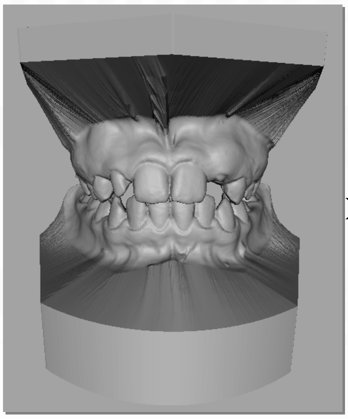
- Fixed placement: select v_1 or v_2
- Fixed placement is faster but lower quality
- It also gives smaller progressive meshes
- Fallback to fixed placement if A is non-invertible

Visualizing Quadrics in 3-D

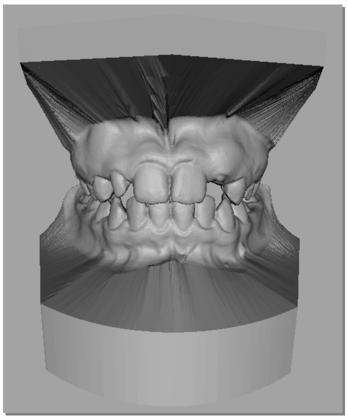


Quadric isosurfaces

- Are ellipsoids (maybe degenerate)
- Centered around vertices
- Characterize shape
- Stretch in least-curved directions

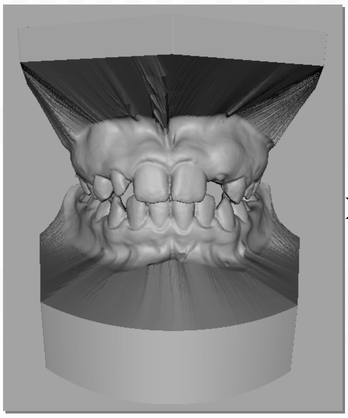


50 sec

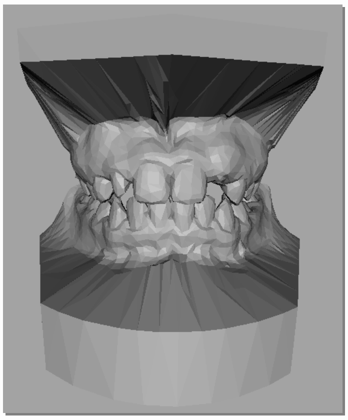


424,376 faces

60,000 faces

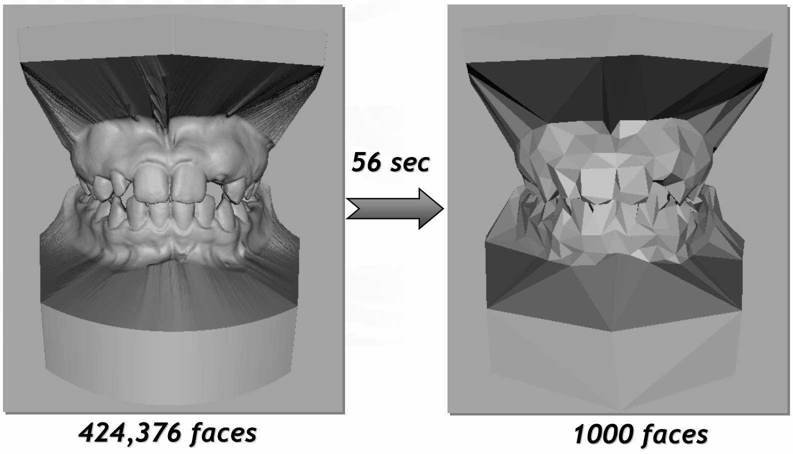


55 sec



424,376 faces

8000 faces



Simplification Summary

Spectrum of effective methods developed

- high quality; very slow *[Hoppe et al, Hoppe]*
- good quality; varying speed
  [Schroeder et al; Klein et al; Ciampalini et al; Guéziec Garland-Heckbert; Ronfard-Rossignac; Lindstrom-Turk]
- lower quality; very fast *[Rossignac–Borrel; Low–Tan]*

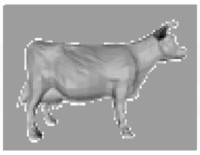
Various other differentiating factors:

- is topology simplified ?
- restricted to manifolds?

Progressive Transmission

Base mesh (M_0) is transmitted first.

Refinement (e.g. vsplit) records are transmitted later and the mesh reconstructed progressively.



Progressive Meshes

Q: How do we use the model for creating different level of details?

A: Multi-resolution model: store the coarsest level object and the encoding of all the decimation operations.

Decimation operations order: $\{d_0, d_1, d_2, \ldots\}$

Each d_i defines: edge contraction

$$M_i \xrightarrow{d_i} M_{i+1}$$

But can also define: vertex split

$$M_i \xleftarrow{d^{-1}_i} M_{i+1}$$

Model Traversal (view independent)

Coarsening
- Order: $d_{i+1}, d_{i+2}, \ldots$
- History: $d_0, d_1, d_2, \ldots, d_i,$

Refinement
- Order: $d_{i+1}, d_{i+2}, \ldots$
- History: $d_0, d_1, d_2, \ldots, d_i,$

Problems:
- No flexibility in order of operations!
- No adaptiveness in level of details!

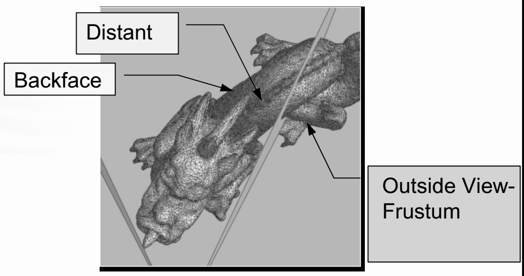
<u>A solution should be able to apply coarsening and refinement not necessarily in the order they were produced!</u>

View-Dependent Refinement

<u>Problem:</u>

While rendering, there are always faces, which are hidden or far from the viewpoint.

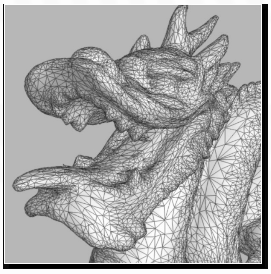
Using traditional simplification techniques, those faces are rendered in the same LOD as the complete mesh.

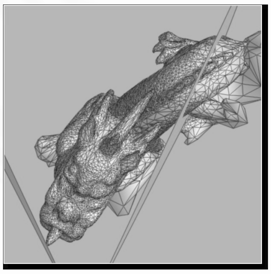


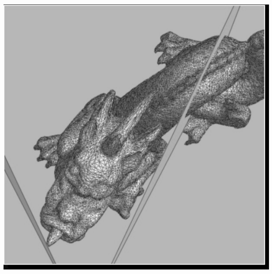
cont'

Goal:

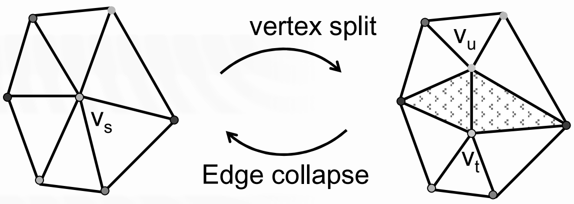
To generate a progressive representation of a mesh, in which only some of its faces are simplified and the rest are fully detailed.







Vertex split and edge collapse



Vertex Hierarchy

By combining the PM presentation with the Parent-Child relationship, a Parent-Child forest can be generated.

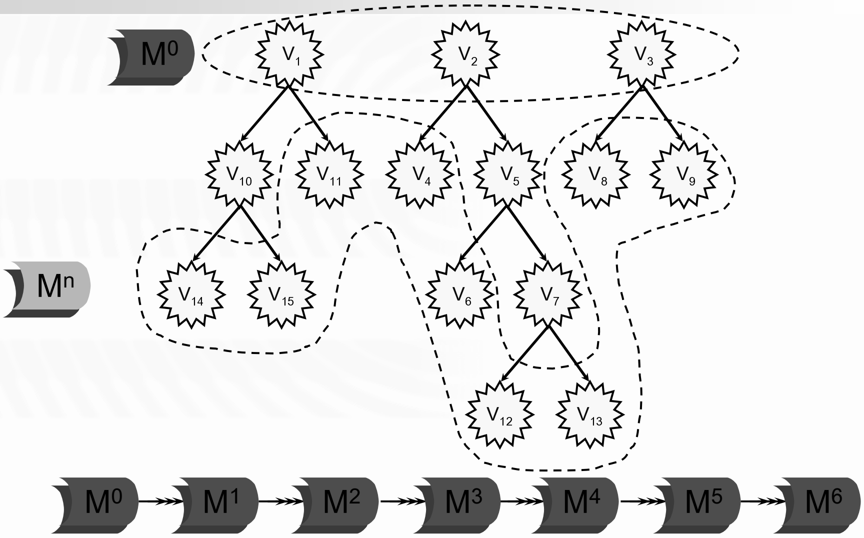
The root vertices of the forest are M0, which is the most simplified mesh.

The leaves of the forest are Mn, which is the original mesh.

A cut is defined as a set of edges with the following properties:

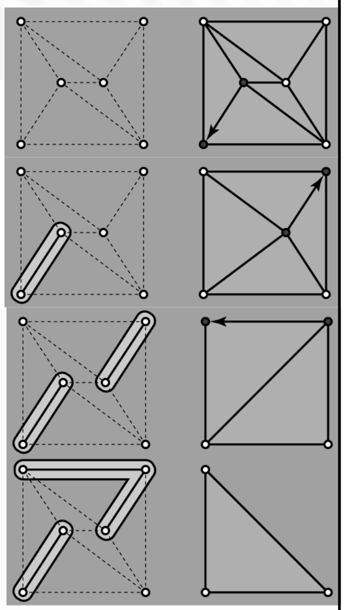
- Every path from the roots to the leaves is intersected.
- Every path is intersected only once.

cont'

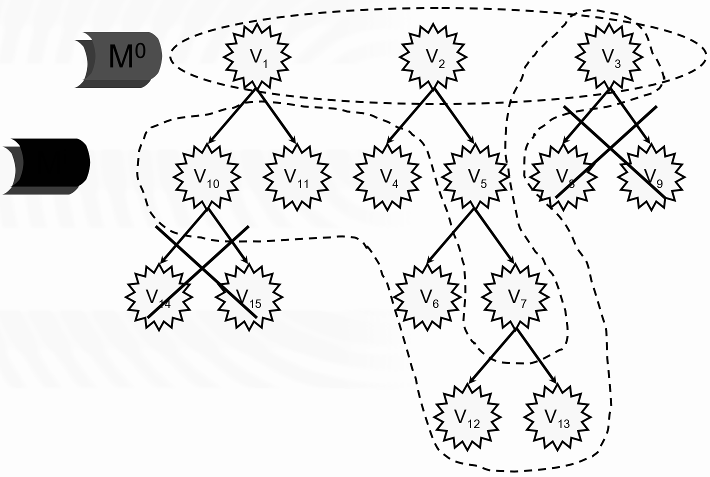


Selective Refinement

- Tree encodes dependency of contractions
- Given a vertex hierarchy forest, a selective refinement mesh can be generated by using selective, out-of-order vsplits and ecols operations.
- The current refined/simplified mesh is a vertex front in the forest.

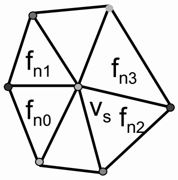


cont'



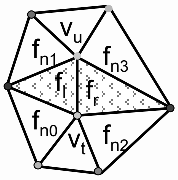
Legal Operations

A face or a vertex are called "*Active*" if they exist in the current front.



Legal vertex split:

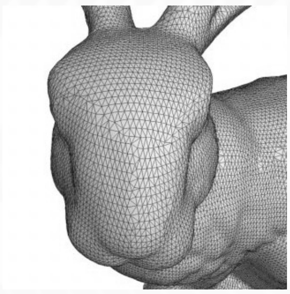
- V_s is an active vertex
- The faces $\{f_{n0}, f_{n1}, f_{n2}, f_{n3}\}$ are all active



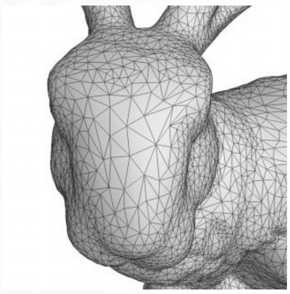
Legal edge collapse:

- V_t and V_u are both active
- The faces $\{f_{n0}, f_{n1}, f_{n2}, f_{n3}\}$ are all active

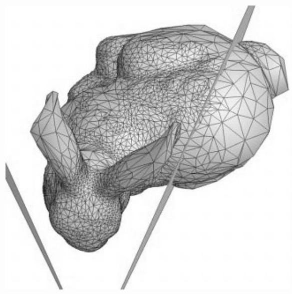
Examples



Original

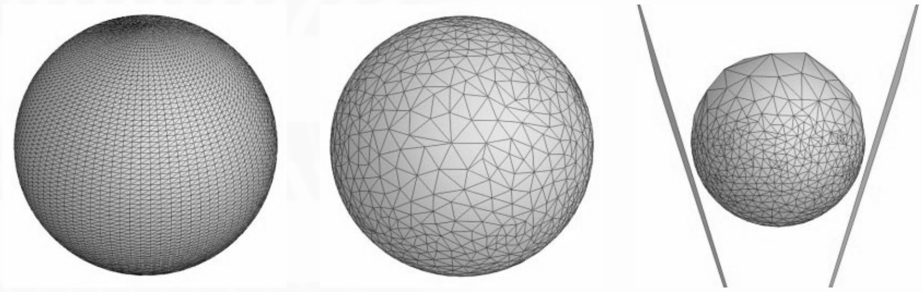


Simplified



Top-View

Examples (cont')



Original Simplified Top-View

Applications Beyond Display

Other important applications are appearing

- surface editing [Guskov *et al* 99]
- surface morphing [Lee *et al* 99]
- hierarchical bounding volumes
- object matching
- shape analysis / feature extraction

Multiresolution Model Summary

Representations are available to support
- progressive transmission
- view-dependent refinement
- hierarchical computation

But limitations remain
- vertex hierarchies may over-constrain adaptation
- adaptation overhead not suitable for all cases

Looking Ahead

We've reached a performance plateau
- broad range of methods for certain situations
- incremental improvement of existing methods

Major progress may require new techniques
- broader applicability of simplification
- higher quality approximations

Needs better understanding of performance
- how well, in general, does an algorithm perform?

Greater Generality

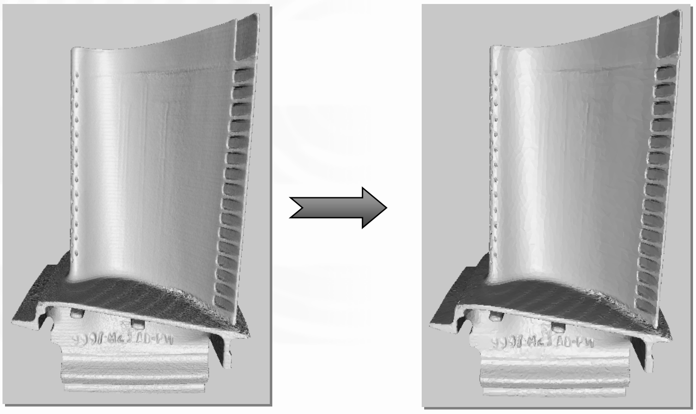
Model types have complexity issues

- tetrahedral volumes, spline patch surfaces, ...

Need to handle extremely large data sets

- precise scans on the order of 10^9 triangles
- this is where simplification is needed the most
- even at 10^6 triangles, many algorithms fail

Too Large for Many Methods



1,765,388 faces → *80,000 faces*

Really Too Large …

Complete data sets: 0.3–1.0 billion triangles

8.2 million triangles (2 mm resolution)

6.8 million triangles (¼ mm resolution)

Better Topological Simplification

Imperceptible holes & gaps can be removed
- most methods do this only implicitly

Few if any methods provide good control
- when exactly are holes removed?
- will holes above a certain size be preserved?

Requires better understanding of the model
- when to simplify geometry vs. topology
- seems to benefit from more volumetric approach

Better Performance Analysis

Better criteria for evaluating similarity

- image-based metric more appropriate for display
- metrics which accurately account for attributes

Most analysis has been case-based

- measure/compare performance on 1 data set

More thorough analysis is required

- theoretical analysis of quality [Heckbert-Garland 99]
- provably good approximations possible?

Conclusions

Substantial progress since 1992

- simplification of 3D surfaces
- multiresolution representations (PM, hierarchies)
- application of multiresolution in different areas

There remains much room for improvement

- more effective, more general simplification
- better analysis and understanding of results
- other multiresolution representations