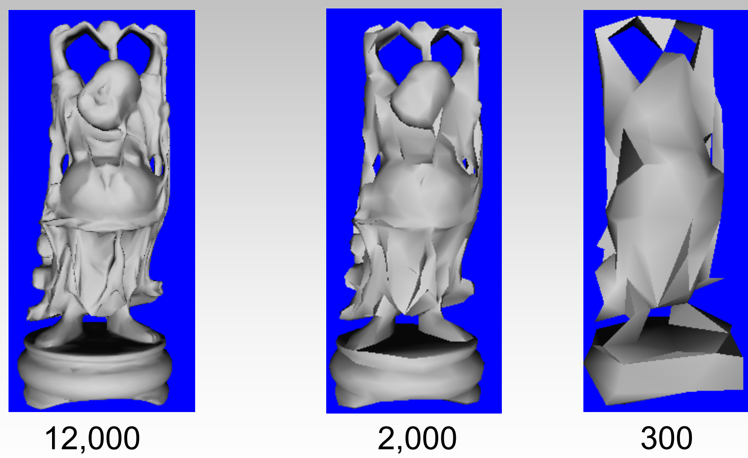
Mesh Simplification

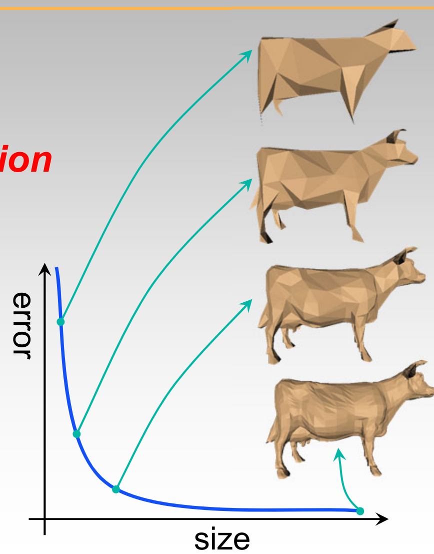


Simplifier

1

Motivation

Reduce information content
Accelerate rendering & computation
Multi-resolution models

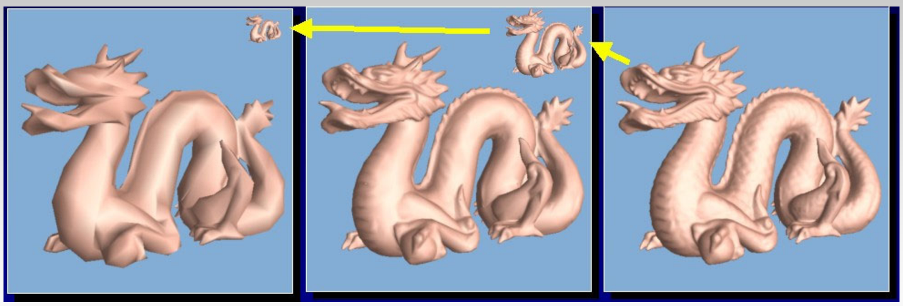


2

Level of Detail (LOD)

Refined mesh for close objects

Simplified mesh for far

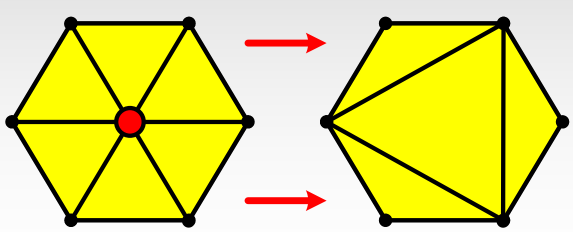


3

Methodology

Sequence of local operations

- Involve near neighbors - only small *patch* affected in each operation
- Each operation introduces error
- Find and apply operation which introduces the least error

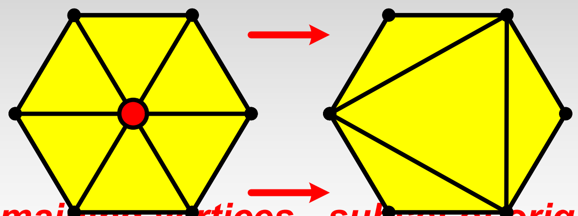


6

Simplification Operations (1)

Decimation

- Vertex removal:
  - $v \leftarrow v\text{-}1$
  - $f \leftarrow f\text{-}2$



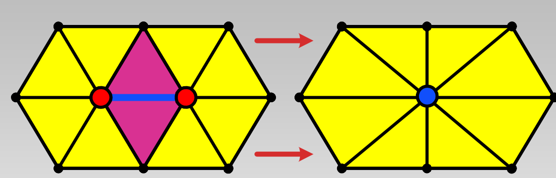
Remaining vertices - subset of original vertex set

7

Simplification Operations (2)

Decimation

- Edge collapse
  - $v \leftarrow v\text{-}1$
  - $f \leftarrow f\text{-}2$



Vertices may move

8

The Basic Algorithm

Repeat

- Select the element with minimal error
- Perform simplification operation (remove/contract)
- Update error (local/global)

Until mesh size / quality is achieved

10

Error Control

Local error: Compare new patch with previous iteration

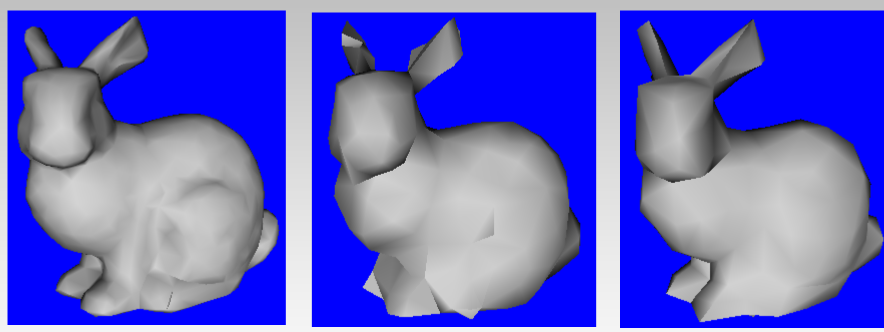
- Fast
- Accumulates error
- Memory-less

Global error: Compare new patch with original mesh

- Slow
- Better quality control
- Can be used as termination condition
- Must remember the original mesh throughout the algorithm

11

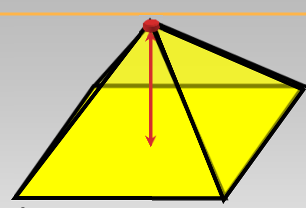
Local vs. Global Error



2000 faces 488 faces 488 faces

12

Simplification Error Metrics

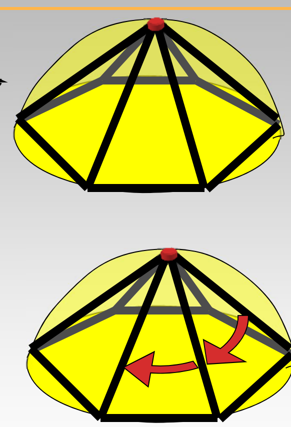


Measures

- Distance to plane
- Curvature

Usually approximated

- Average plane
- Discrete curvature



$\Sigma\alpha / 2\pi$

13

Implementation Details

Vertices/Edges/Faces data structure

- Easy access from each element to neighboring elements

Use priority queue (e.g. heap)

- Fast access to element with minimal error
- Fast update

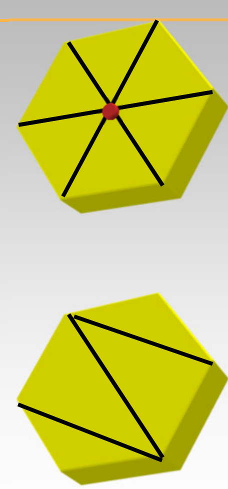
14

Vertex Removal Algorithm

Simplification operation: Vertex removal

Error metric: Distance to average plane

May preserve mesh features (creases)



15

Algorithm Outline

Characterize local topology/geometry

Classify vertices as removable or not

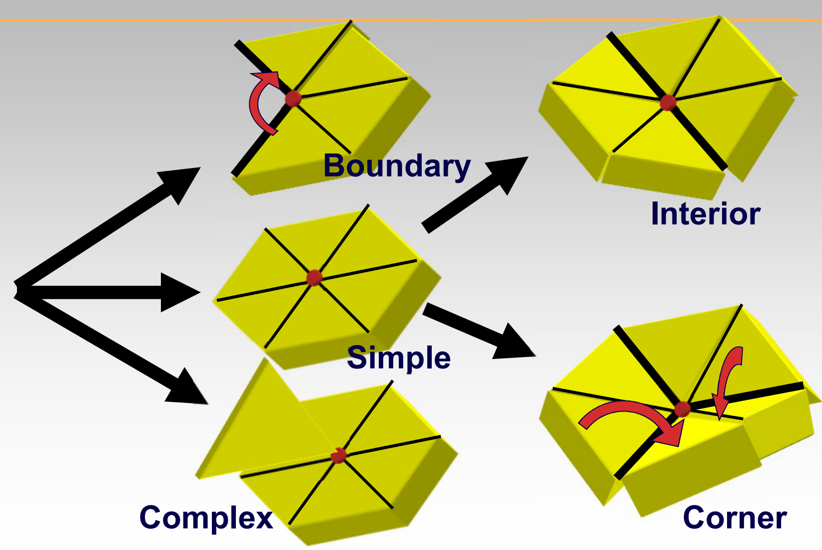
Repeat

- Remove vertex
- Triangulate resulting hole
- Update error of affected vertices

Until reduction goal is met

16

Characterizing Local Topology/Geometry



17

Decimation Criterion

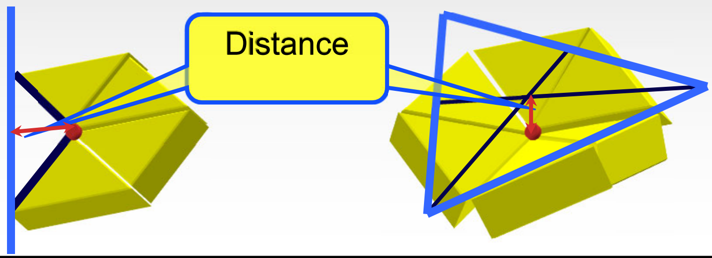
E_{MAX} – user defined parameter

Simple vertex:

- Distance of vertex to the face loop average plane < E_{MAX}

Boundary vertices:

- Distance of the vertex to the new boundary edge < E_{MAX}



Triangulating the Hole

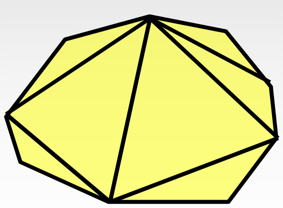
Vertex removal produces non-planar loop

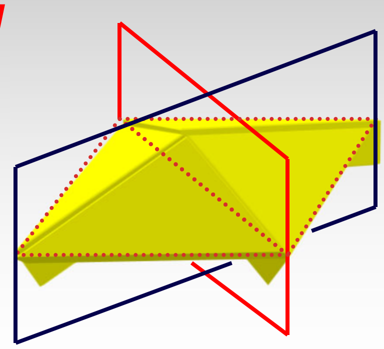
- Split loop recursively
- Split plane orthogonal to the average plane

Control aspect ratio

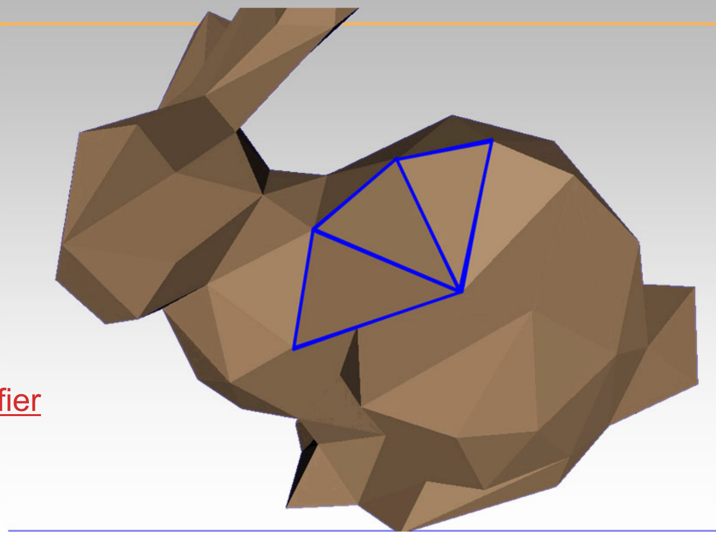
Triangulation may fail

- Vertex is not removed





Example



Simplifier

20

Pros and Cons

Pros:

- Efficient
- Simple to implement and use
  - *Few input parameters to control quality*
- Reasonable approximation
- Works on very large meshes
- Preserves topology
- Vertices are a subset of the original mesh

Cons:

- Error is not bounded
  - *Local error evaluation causes error to accumulate*

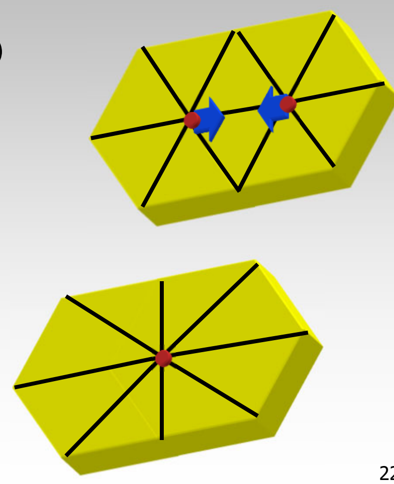
21

Edge Collapse Algorithm

Simplification operation:

- Edge collapse (pair contraction)

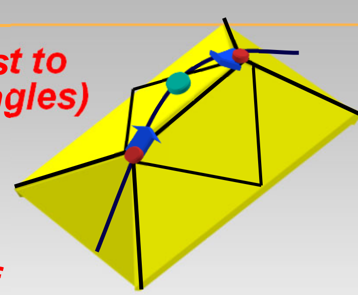
Error metric:
distance, pseudo-global



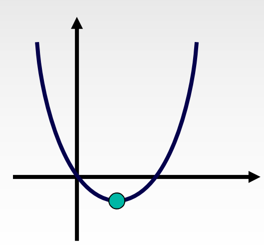
22

Distance Metric: Quadrics

Choose point closest to set of planes (triangles)



Sum of squared distances to set of planes is quadratic ⇒ has a minimum



23

Quadrics

Plane

- $Ax + By + Cz + D = 0$, where $A^2 + B^2 + C^2 = 1$
- $p = [A, B, C, D]$, $v = [x, y, z, 1]$, $v\,p^T = 0$

Quadratic distance between v and p:

$$\Delta_p(v) = (v\,p^T)^2 = (v\,p^T)(p\,v^T) = v\,(p^T p)\,v^T = v\,K_P\,v^T$$

$$K_P = \begin{bmatrix} A^2 & AB & AC & AD \\ AB & B^2 & BC & BD \\ AC & BC & C^2 & CD \\ AD & BD & CD & D^2 \end{bmatrix}$$

24

© Alla Sheffer

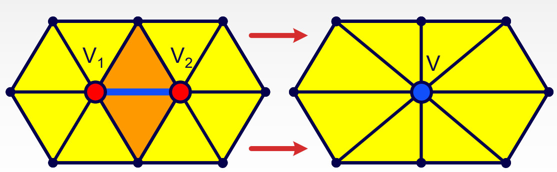
Distance to Set of Planes

$$\Delta(v) = \sum_{p \in planes(v)} \Delta_p(v) = \sum_{p \in planes(v)} (v\,K_p\,v^T) = v\,(\sum_{p \in planes(v)} K_p)\,v^T = v\,Q_v\,v^T$$

After v_1, v_2 are contracted to v,
$Q_v \leftarrow Q_{v1} + Q_{v2}$

Pseudo-global

All original planes persist during the entire simplification process



25

© Alla Sheffer

Contracting Two Vertices

Goal: Given edge $e = (v_1,v_2)$, find contracted $v = (x,y,z,1)$ that minimizes $\Delta(v)$:

$$\partial\Delta/\partial x = \partial\Delta/\partial y = \partial\Delta/\partial z = 0$$

Solve system of linear normal equations:

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} v = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

If no solution - select the edge midpoint

26

Algorithm

Compute Q_V for all the mesh vertices

Identify all valid pairs

Compute for each valid pair (v_1,v_2) the contracted vertex v and its error $\Delta(v)$

Store all valid pairs in a priority queue (according to $\Delta(v)$)

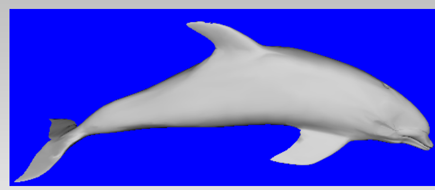
While reduction goal not met

- Contract edge (v_1,v_2) with the smallest error to v
- Update the priority queue with new valid pairs

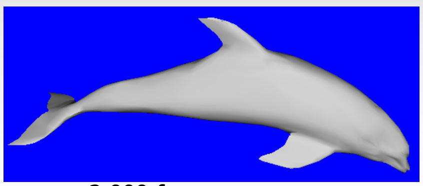
28

Examples

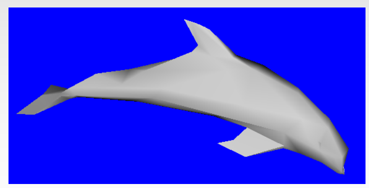
Dolphin (Flipper)



Original - 12,337 faces



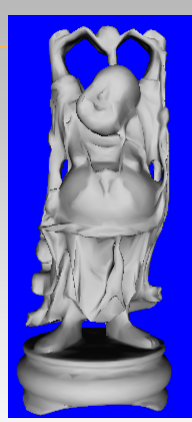
2,000 faces



300 faces (142 vertices)

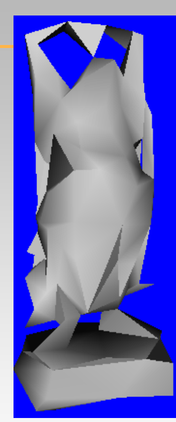
29

Examples



Original - 12,000

2,000 faces



298 faces (140 vertices)

Simplifier

30

Pros and Cons

Pros

- Error is bounded
- Allows topology simplification
- High quality result
- Quite efficient

Cons

- Difficulties along boundaries
- Difficulties with coplanar planes
- Introduces new vertices not present in the original mesh

31