Graphics&Geometric Computing Lab @USTC

Mesh Simplification

Ligang Liu
Graphics&Geometric Computing Lab
USTC
http://staff.ustc.edu.cn/~lgliu

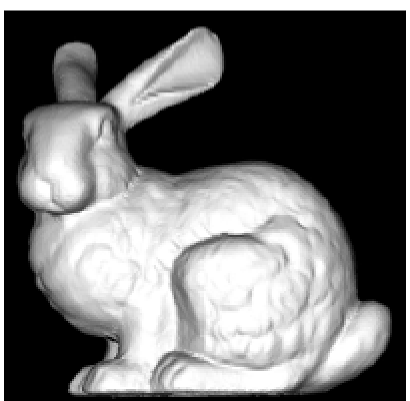
Problems

- High resolution meshes becoming increasingly available
  - 3D active scanners
  - Computer vision methods
  - Meshes extracted from volumetric data
  - Terrain data

Motivation

- Reduce information content
- Accelerate rendering
- Improved sampling
- Multi-resolution models

69451 faces
35947 vertices

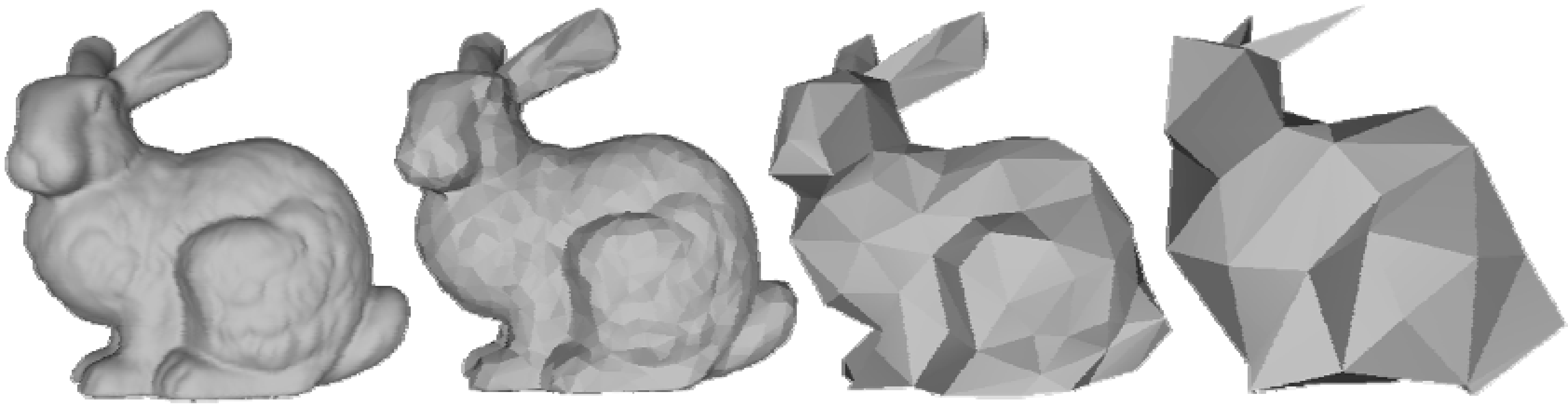


871414 faces
437645 vertices

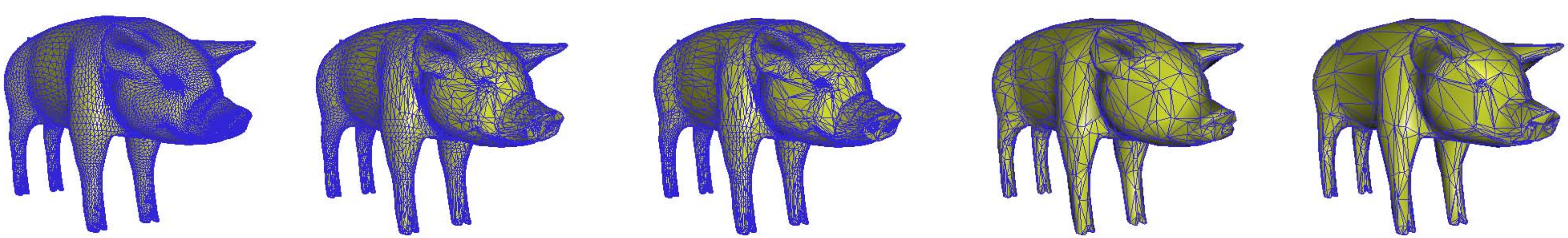
1087716 faces
543652 vertices

1765388 faces
882954 vertices

Simplification Examples



69,451 polys   2,502 polys   251 polys   76 polys

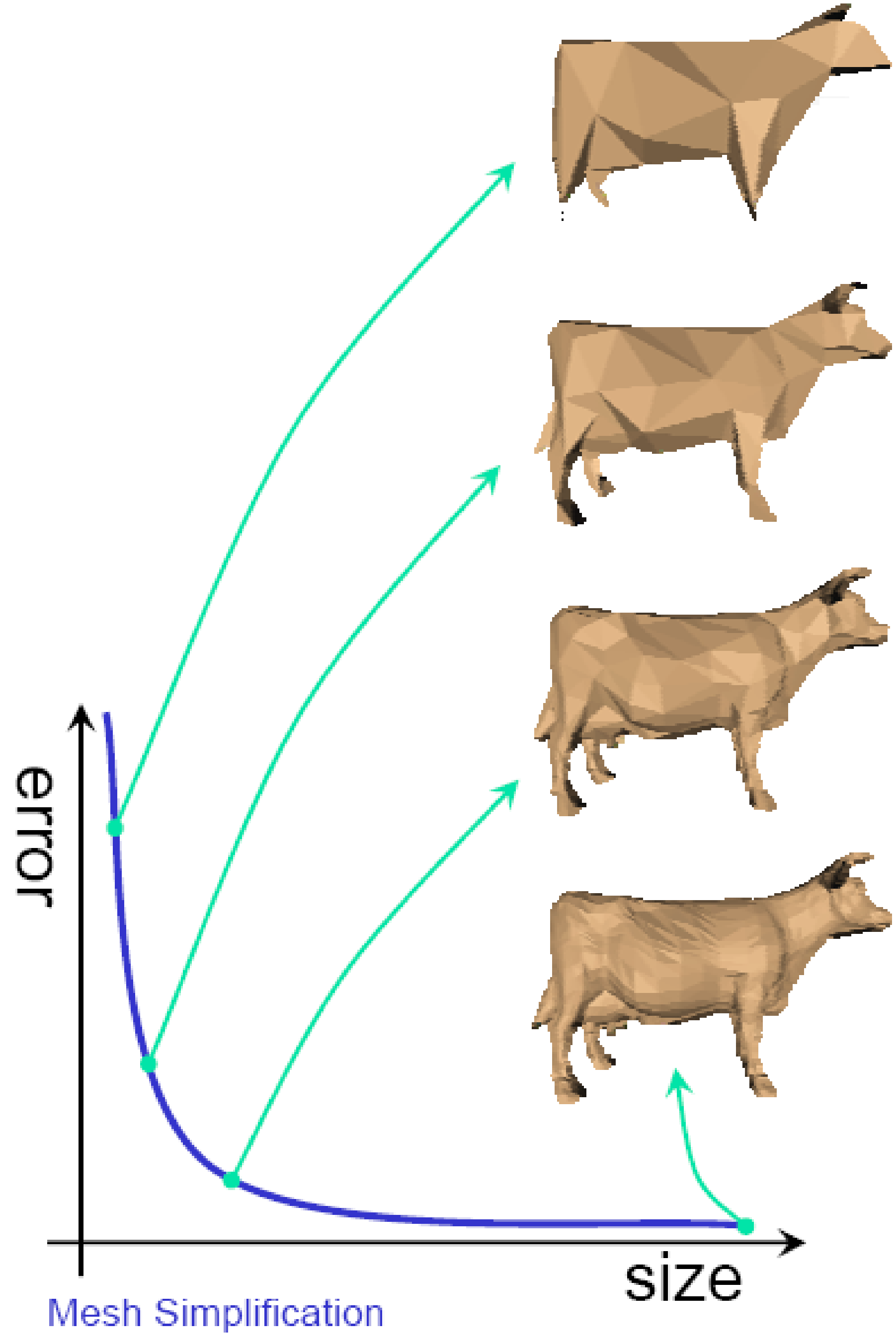


Simplification Applications

- Level-of-detail modeling
  - Generate a family of models for the same object with different polygon counts
  - Select the appropriate model based on estimates of the object's projected size
- Simulation proxies
  - Run the simulation on a simplified model
  - Interpolate results across a more complicated model to be used for rendering

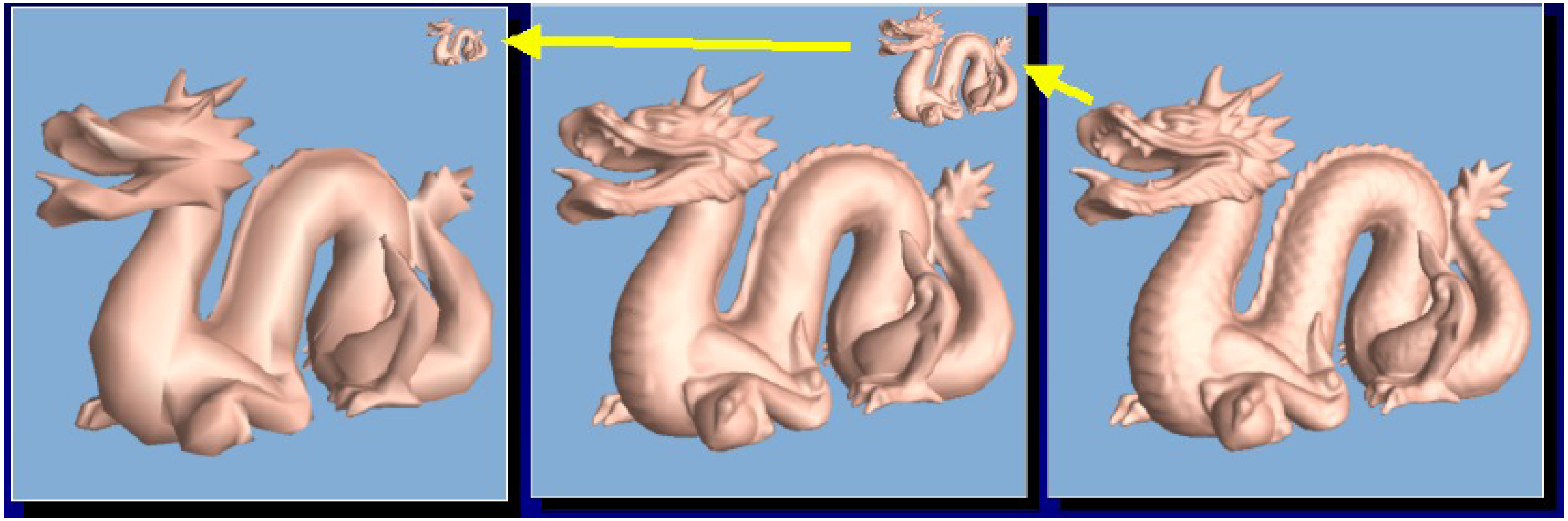
Trad

- Size
- Error



Level of Detail (LOD)

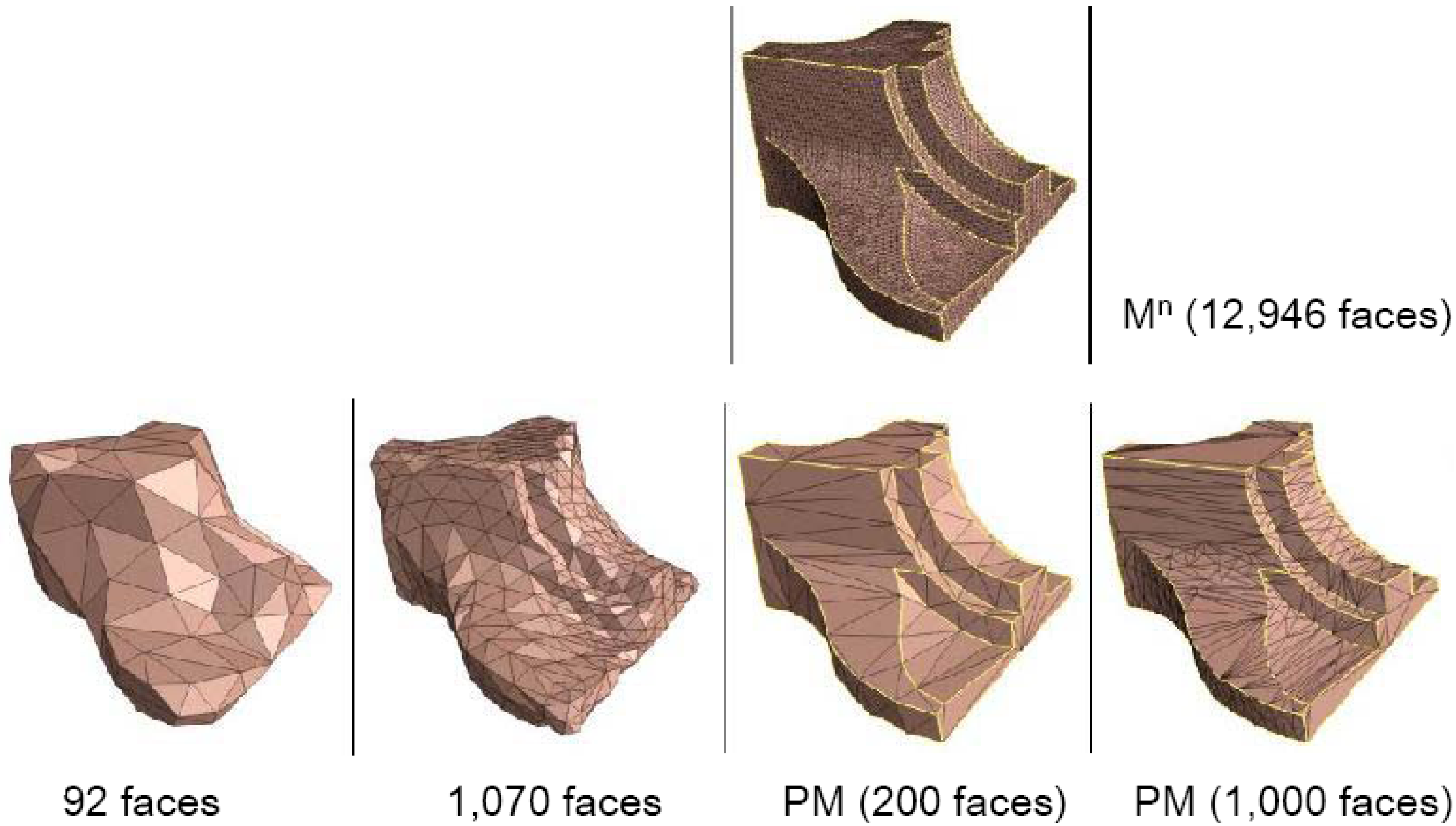
- Refined mesh for close objects
- Simplified mesh for far



Performance Requirements

- Offline
  - Generate model at given level(s) of detail
  - Focus on quality
- Real-time
  - Generate model at given level(s) of detail
  - Focus on speed
  - Requires preprocessing
  - Time/space/quality tradeoff

Quality



Classification

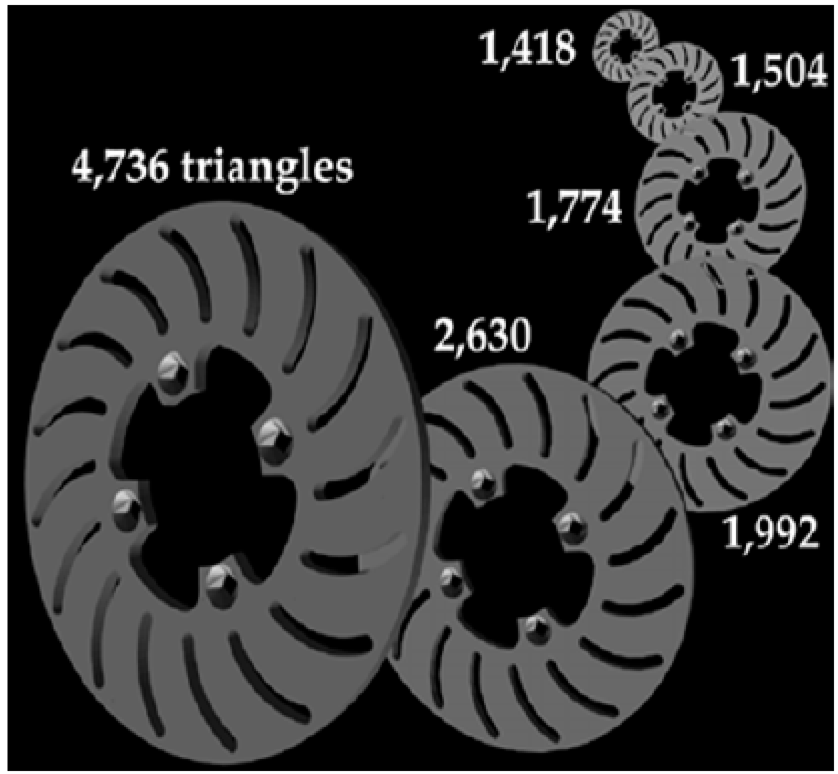
- Topology-preserving vs. topology-modifying
- Refinement-based vs. decimation-based
  - Refinement = top-down: e.g., subdivision
  - Decimation-based = bottom-up up – most common for meshes with irregular connectivity (unstructured)
- Local vs. global. If local,
  - Which decimation operator?
  - Vertices, edges, or faces, what to remove and in what order?
  - Computation of new vertex/edge/face locations

Classification

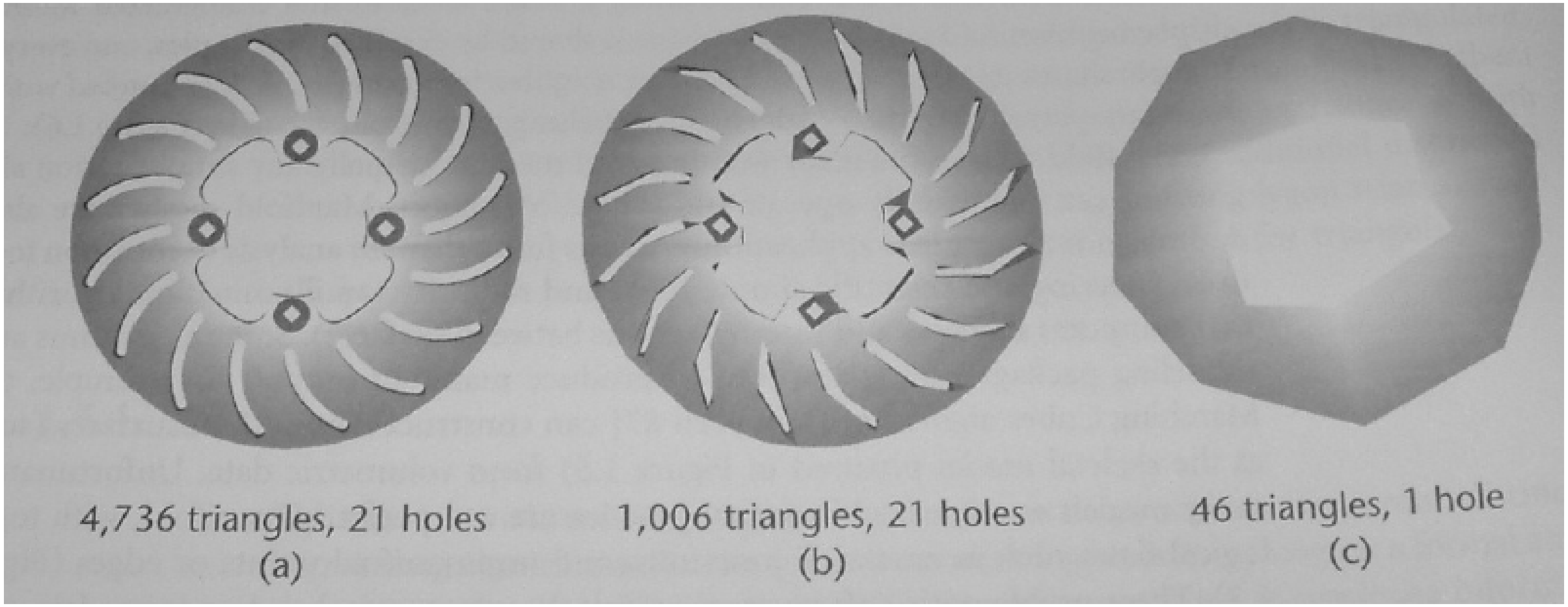
- Fidelity-based or budget-based– remember?
- What fidelity measure to use?
  – Object-space: view-independent; several approaches
  – Image-space: view-dependent; this is what matters
  – Perceptual concerns: not fully understood, at least in computer graphics
  – Guaranteed error bound?

Topology-Preserving Simplification

- Limits drastic simplification if genus of the model is high
- Solution: also simplify mesh topology – e.g., fill those holes



Simplifying Mesh Topology



- How can this be done? – *hole collapsing*? – that is one idea …

Comparison

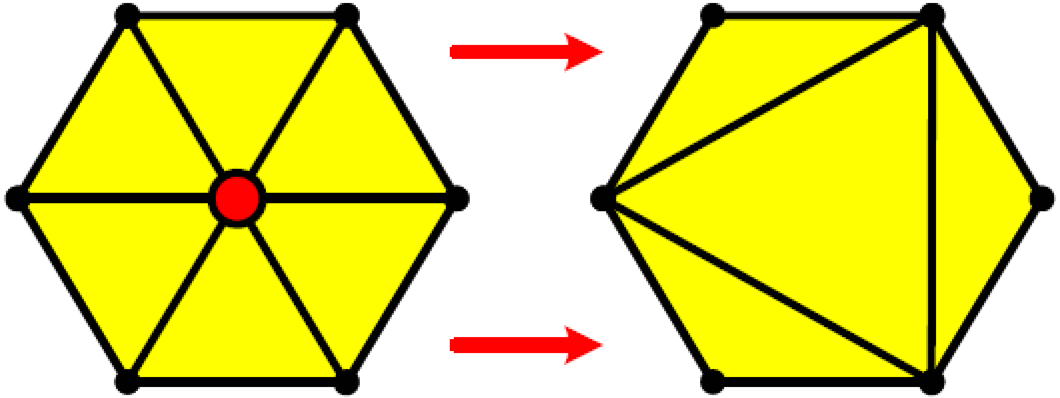
- Topology-preserving
  - Better visual fidelity with less change to the mesh
  - Smoother transitions between levels – small changes
  - Limits drastic simplification when topology is complex
  - Cannot merge small objects
  - Mostly deal with 2D manifold mesh, but not all acquired models are manifolds due to *noise* in data
- Topology-modifying
  - Can have more drastic simplification – e.g., fill holes
  - Poorer visual fidelity and *popping* when filling a hole

Algorithms

- Vertex Removal/Decimation
- Edge Collapse
- Appearance-Preserving Simplification

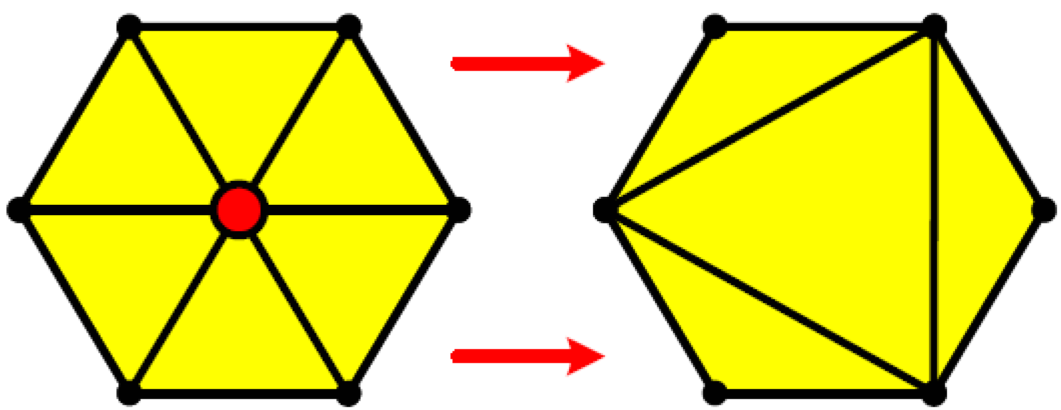
Methodology

- 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



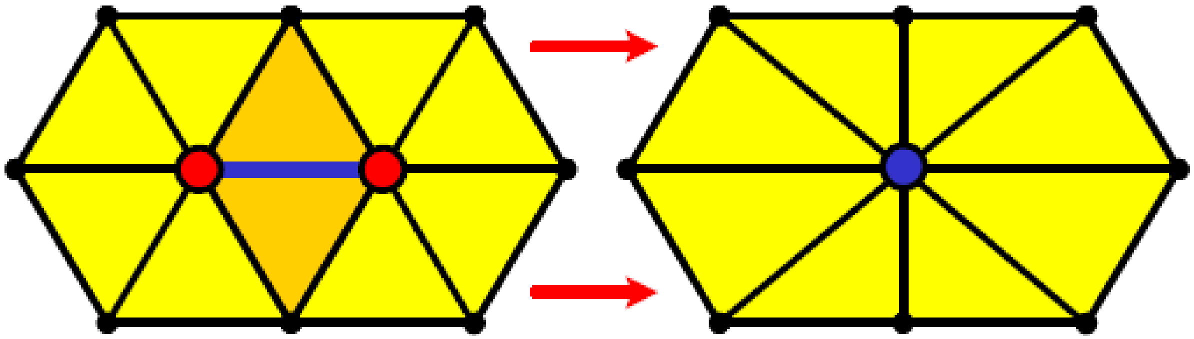
Simplification Operations (1)

- Decimation
  - Vertex removal:
    - v ← v-1
    - f ← f-2
- Remaining vertices - subset of original vertex set



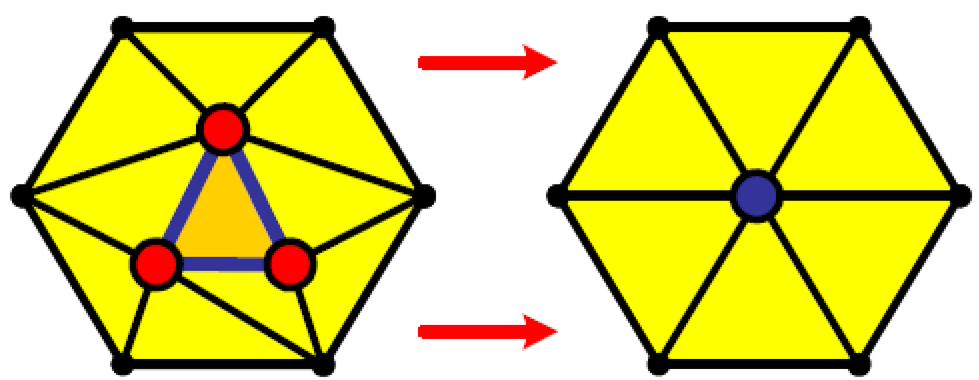
Simplification Operations (2)

- Decimation
  - Edge collapse
    - v ← v-1
    - f ← f-2
- Vertices may move



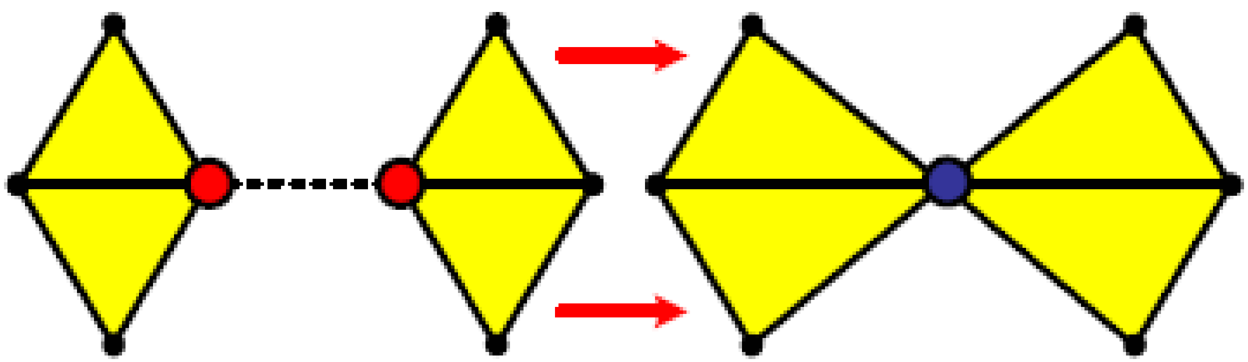
Simplification Operations (3)

- Decimation
  - Triangle collapse
    - v ← v-2
    - f ← f-4
- Vertices may move



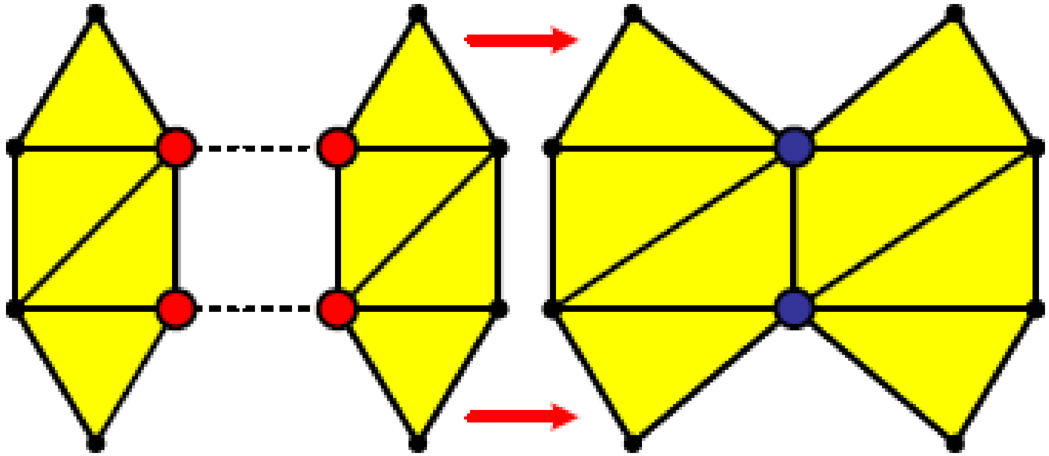
Simplification Operations (4)

- Contraction
  - Pair contraction
- Vertices may move



Simplification Operations (5)

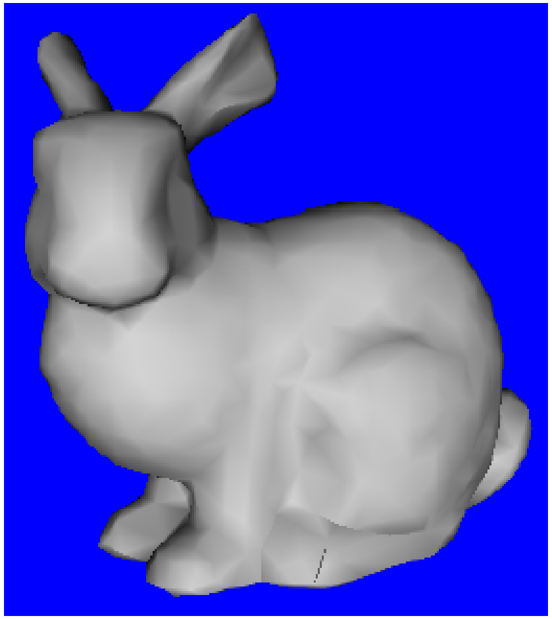
- Contraction
  - Cluster contraction (set of vertices)
- Vertices may move



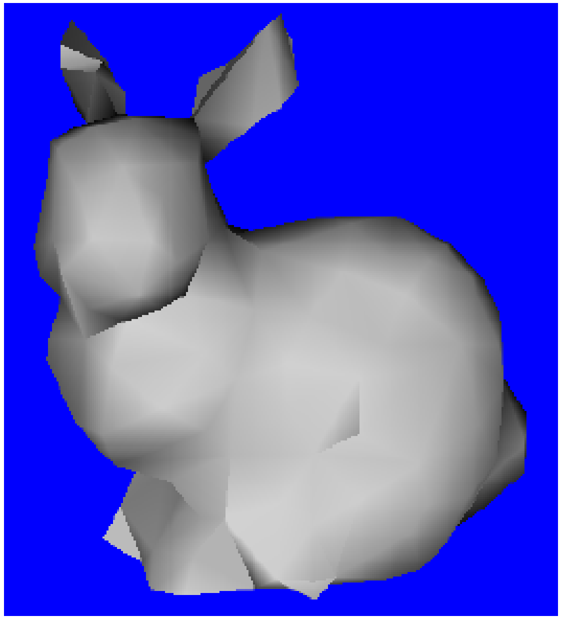
Error Control

- 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

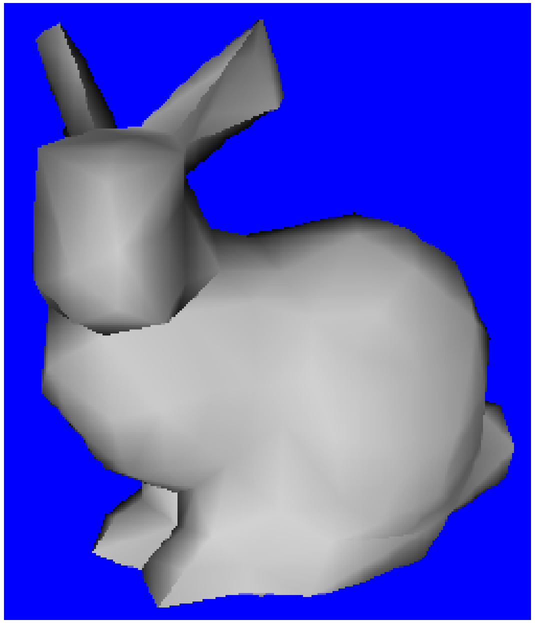
Local vs. Global Error



2000 faces



488 faces



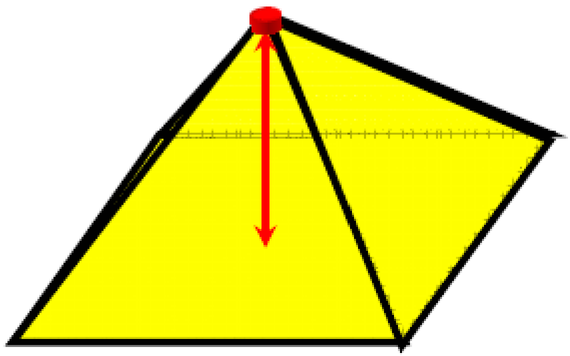
488 faces

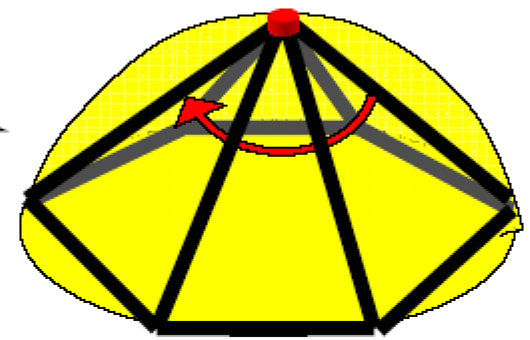
1. Local Simplification Strategies

The Basic Algorithm

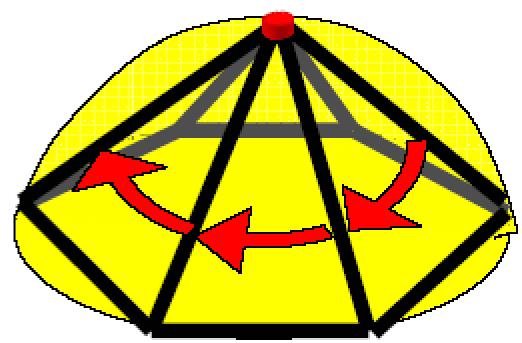
- Repeat
  - Select the element with minimal error
  - Perform simplification operation (remove/contract)
  - Update error (local/global)
- Until mesh size / quality is achieved

Simplification Error Metrics





- Measures
  - Distance to plane
  - Curvature
- Usually approximated
  - Average plane
  - Discrete curvature



$\Sigma\alpha / 2\pi$

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

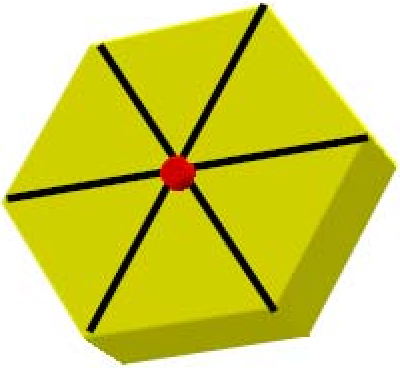
1.1 Vertex Removal Algorithm

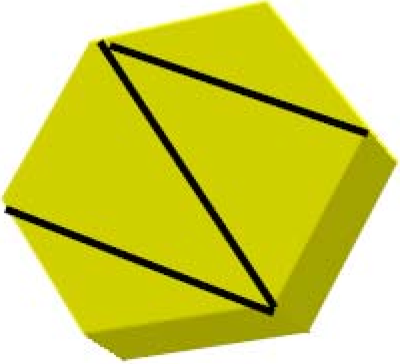
Mesh Decimation

[Schroeder et al 92]

Algorithm Overview

- Simplification operation: Vertex removal
- Error metric: Distance to average plane
- May preserve mesh features (creases)

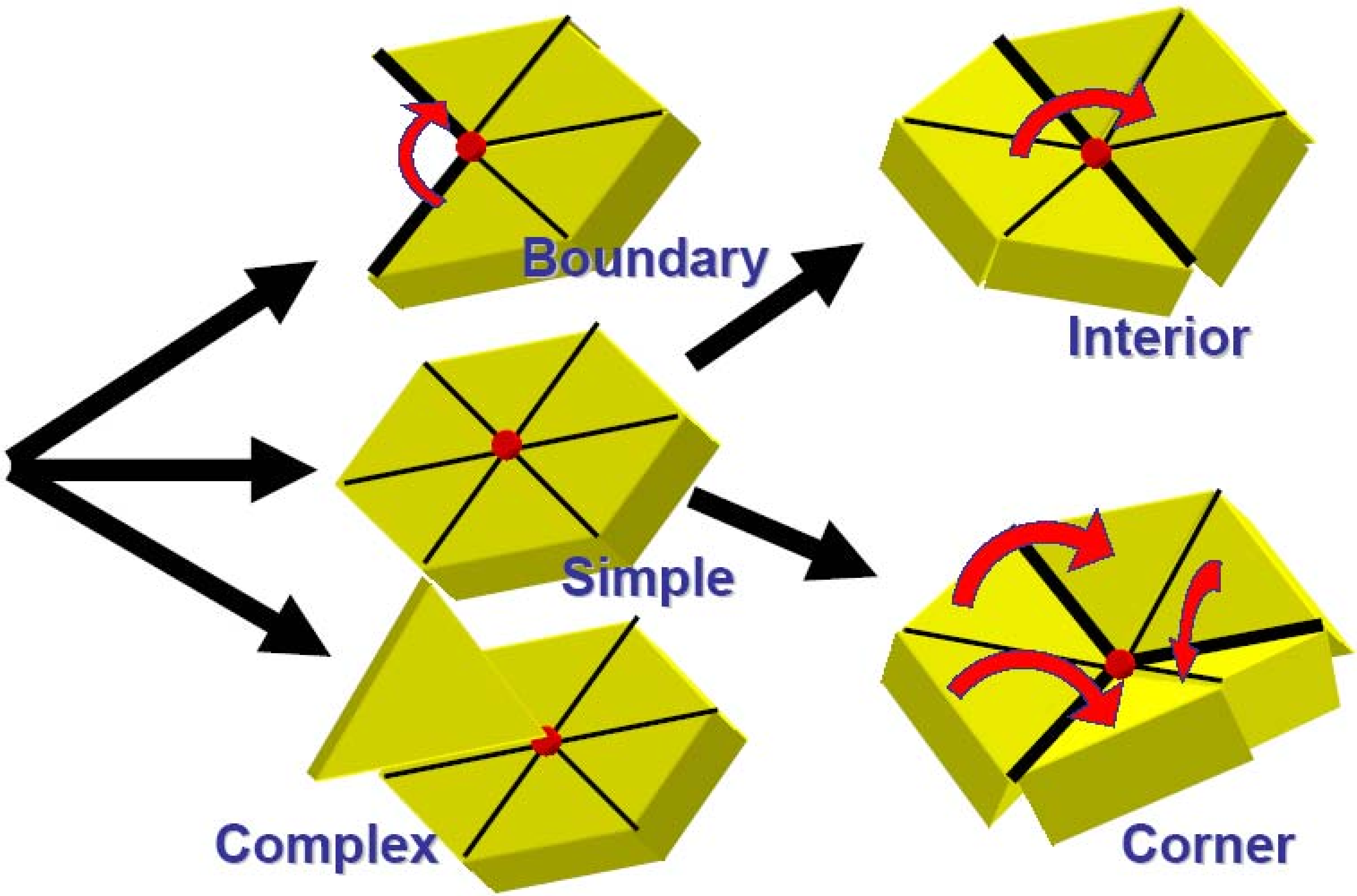




Algorithm Outline

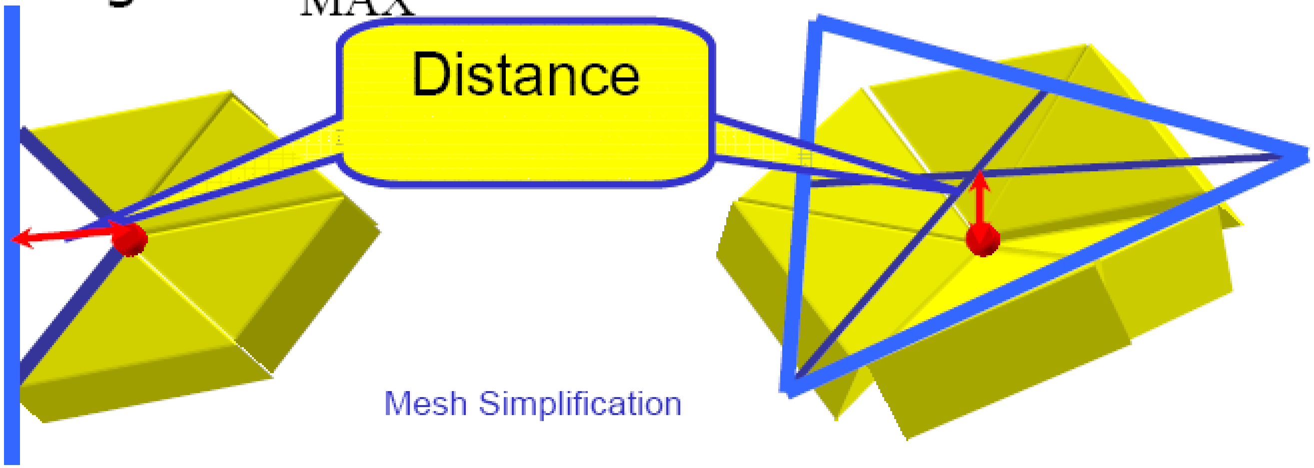
- Characterize local topology/geometry
- Classify vertices as removable or not
- ***Repeat***
  - Remove vertex
  - Triangulate resulting hole
  - Update error of affected vertices
- ***Until*** reduction goal is met

Characterizing Local Topology/Geometry



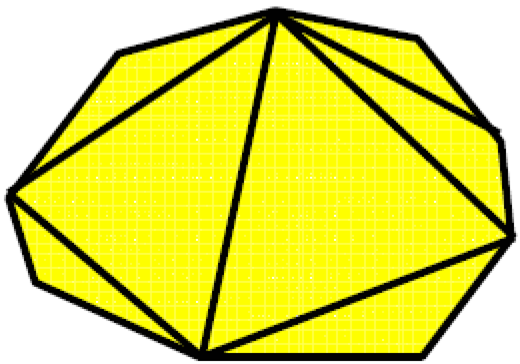
Decimation Criterion

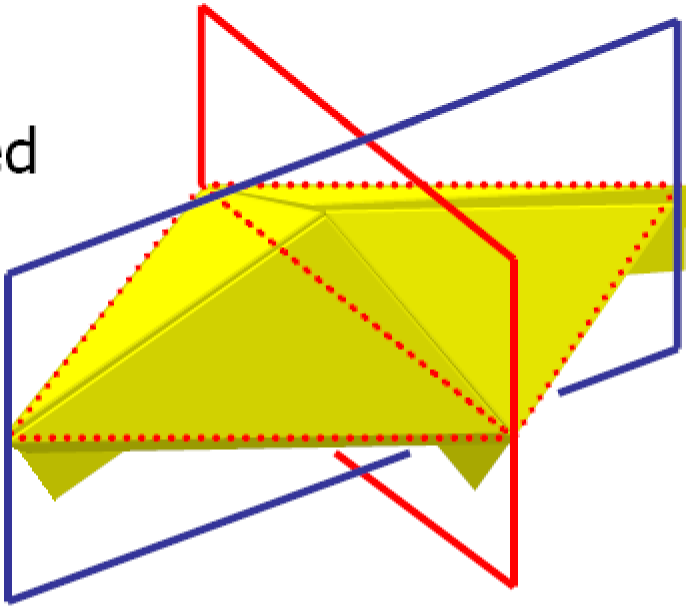
- E_{MAX} – user defined parameter
- Simple vertex:
  - Distance of vertex to the face loop average plane < E_{MAX}
- Boundary vertices:
  - Distance of the vertex to the new boundary edge < E_{MAX}



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





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

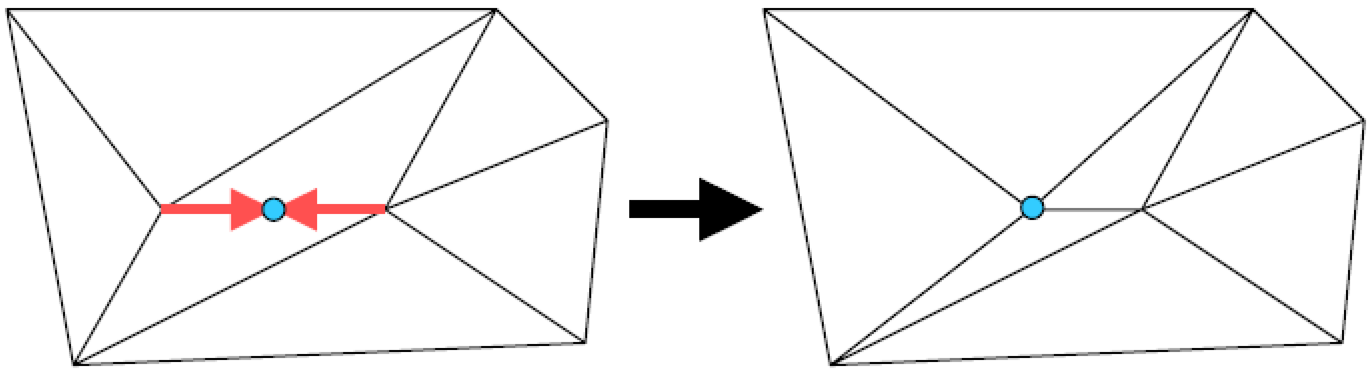
1.2 Edge Collapse Algorithm

Edge Contraction

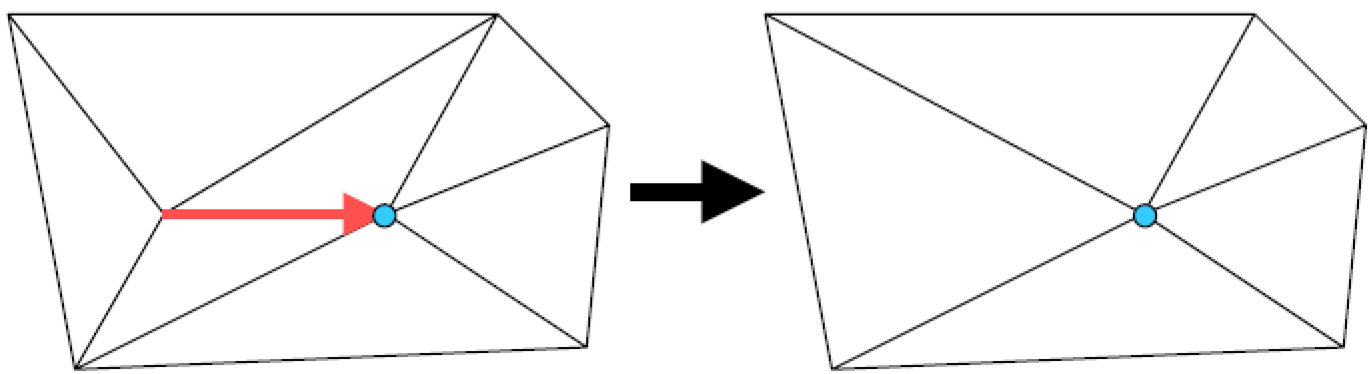
[Hoppe el al 93]

Edge Collapse

General edge collapse



Half-edge collapse (does not introduce new vertices)

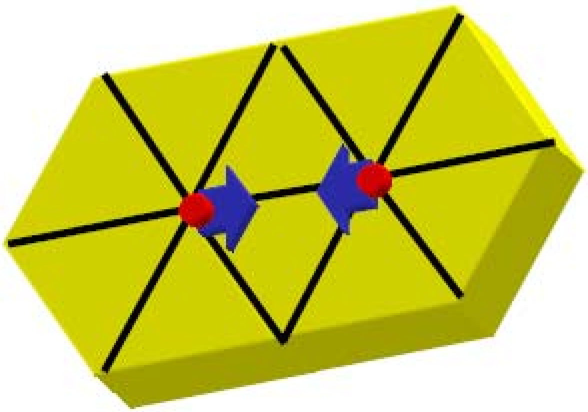


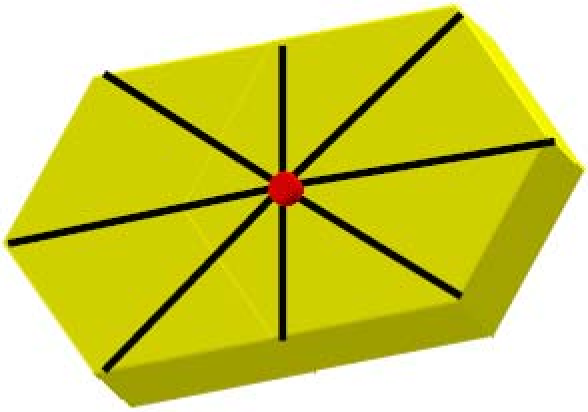
Edge Collapse

- Currently the most popular technique
  - Hoppe, Garland–Heckbert, Lindstrom-Turk, Ronfard-Rossignac, Guéziec, and several others
  - simpler operation than vertex removal
  - well-defined on any simplicial complex

Algorithm Overview

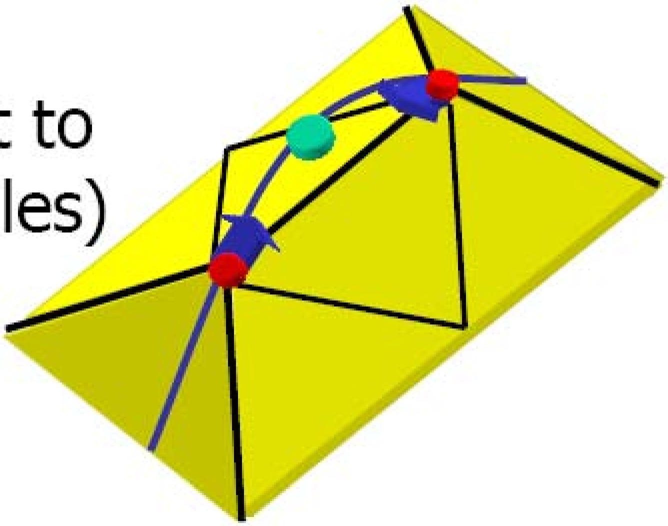
- Simplification operation: Pair contraction
- Error metric: distance, pseudo-global
- Simplifies also topology



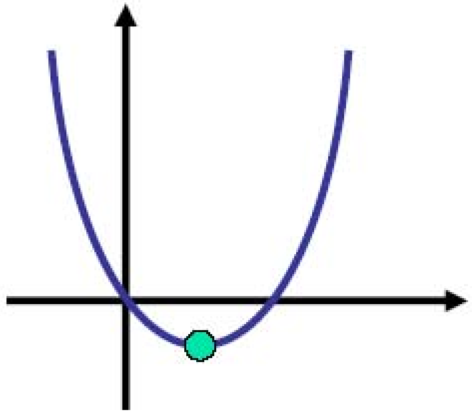


Distance Metric: Quadrics

- Choose point closest to set of planes (triangles)



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



The Quadric Error Metric [Garland & Heckbert 1997]

- Given a plane, we can define a quadric Q

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

measuring squared distance to the plane as

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

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

The Quadric Error Metric

- Sum of quadrics represents set of planes

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

- Each vertex has an associated quadric
  - Error(v_i) = $Q_i\,(v_i)$
  - Sum quadrics when contracting $(v_i, v_j) \rightarrow v'$
  - Cost of contraction is $Q(v')$

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

The Quadric Error Metric

- Sum of endpoint quadrics determines v'
  - Fixed placement: select v_1 or v_2
  - Optimal placement: choose *v'* minimizing *Q(v')*

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

  - Fixed placement is faster but lower quality
  - But it also gives smaller progressive meshes
  - Fallback to fixed placement if A is non-invertible

Contracting Two Vertices

- **Goal**: Given edge $e = (v_1, v_2)$, find contracted

  $v = (x,y,z)$ that minimizes $\Delta(v)$:

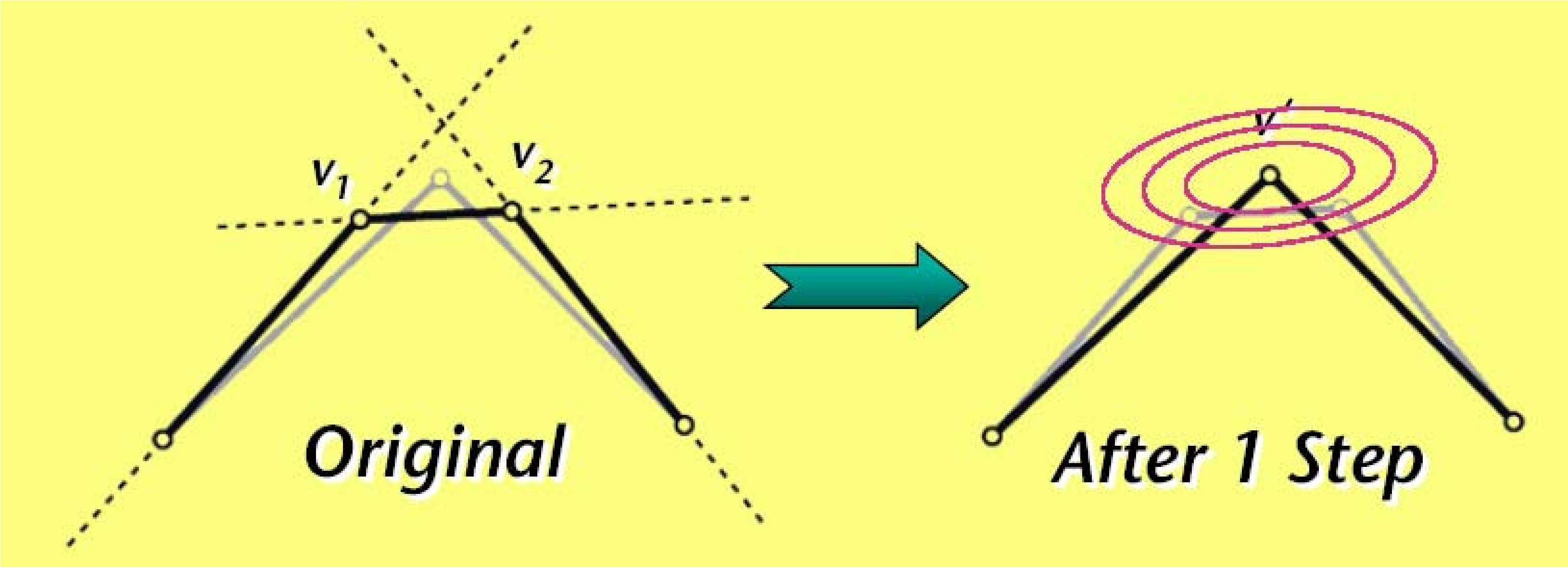
  $\partial\Delta/\partial x = \partial\Delta/\partial y = \partial\Delta/\partial z = 0$
- Solve system of linear *normal equations*:

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

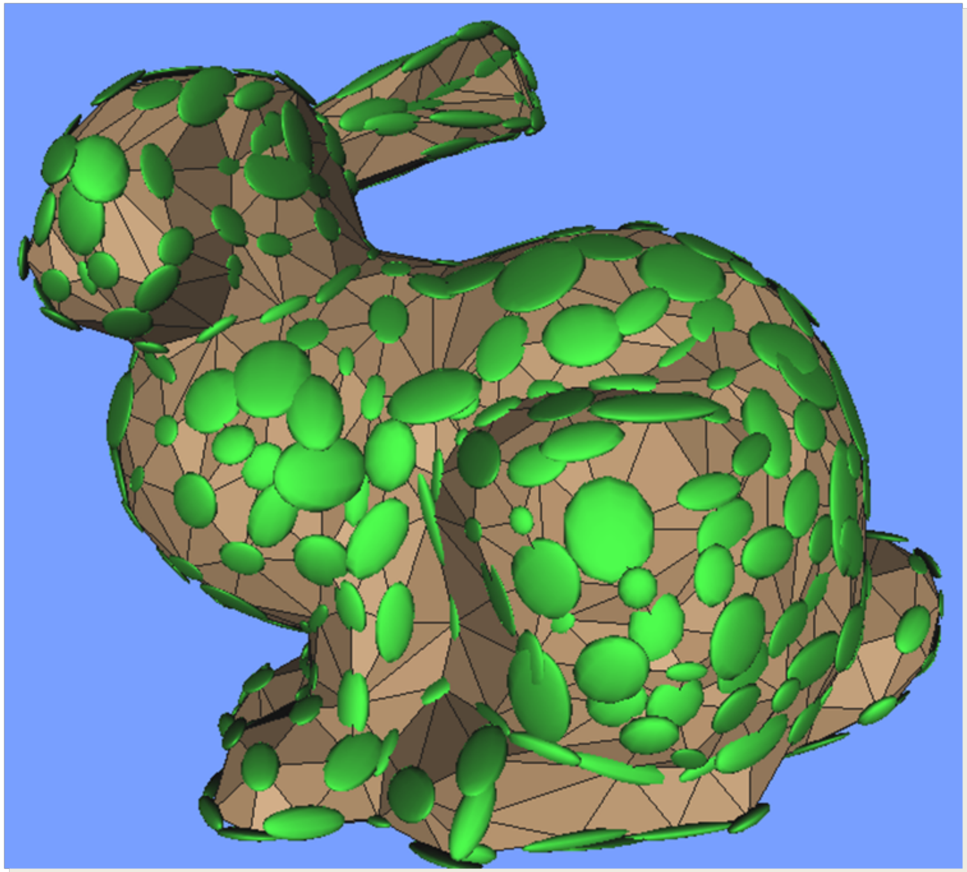
If no solution - select the edge midpoint

A Simple Example: Contraction & "Planes" in 2D

- Lines defined by neighboring segments
  - Determine position of new vertex
  - Error iso-contours shown on right



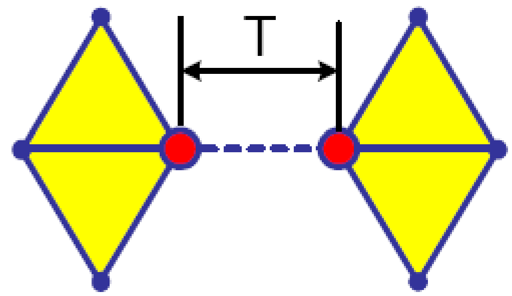
Visualizing Quadrics



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

Selecting Valid Pairs for Contraction

- Edges:
  $\{(v_1,v_2) : (v_1v_2) \text{ is in the mesh}\}$

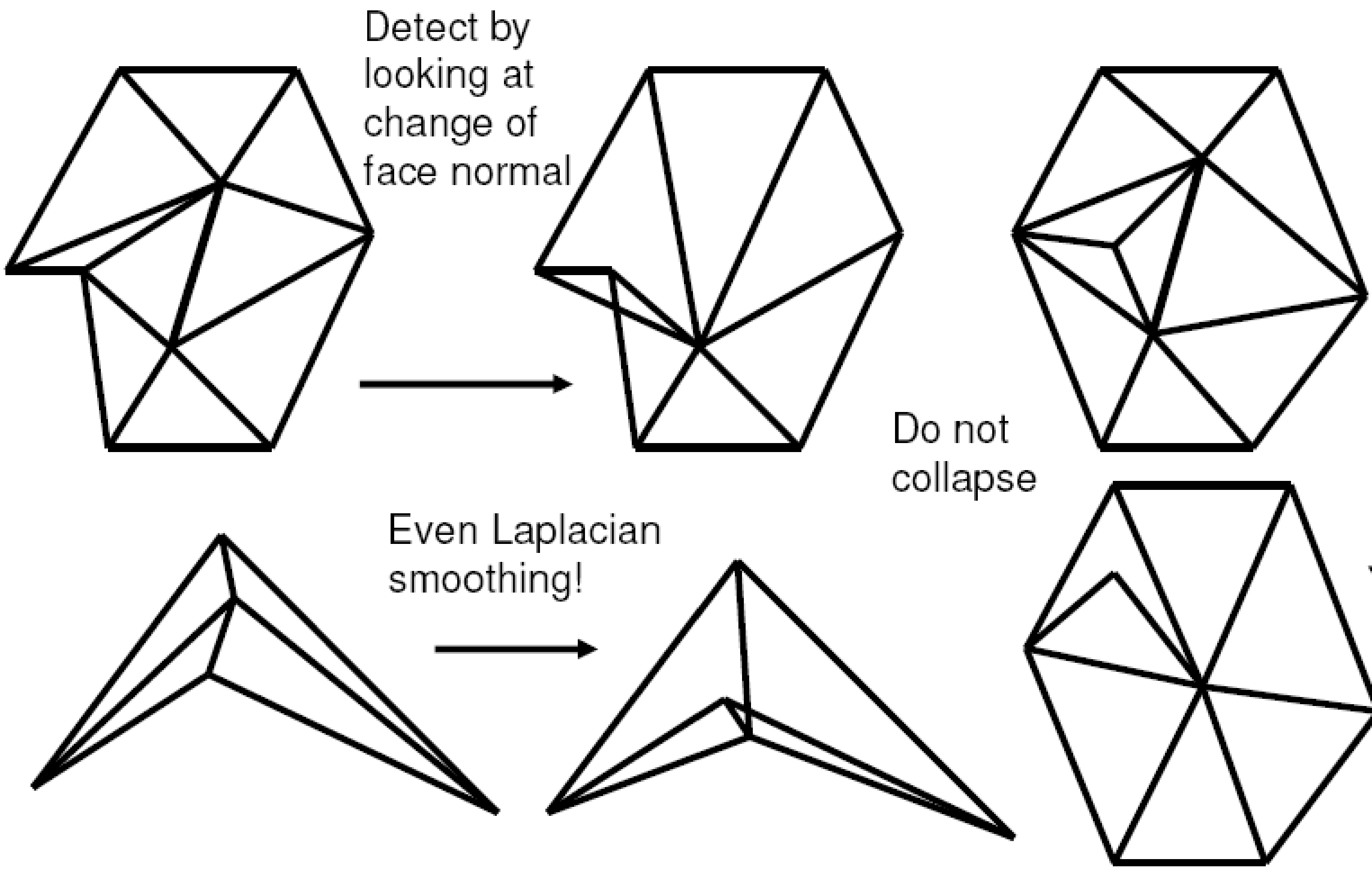


- Close vertices:
  $\{(v_1,v_2) : ||v_1 - v_2|| < T\}$
  - Threshold T is input parameter

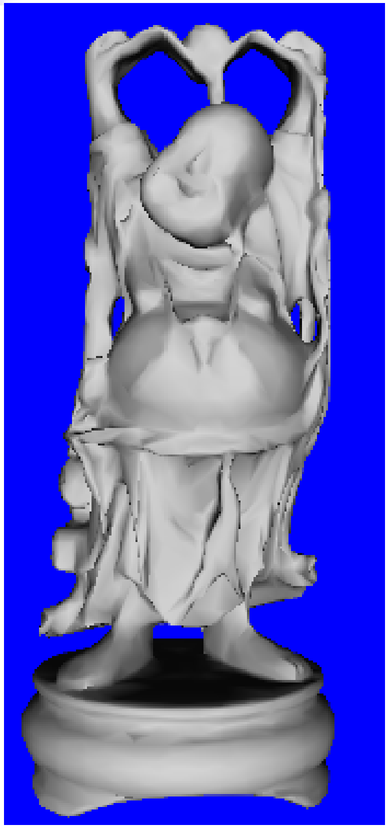
Algorithm

- 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

Artifacts by Edge Collapse

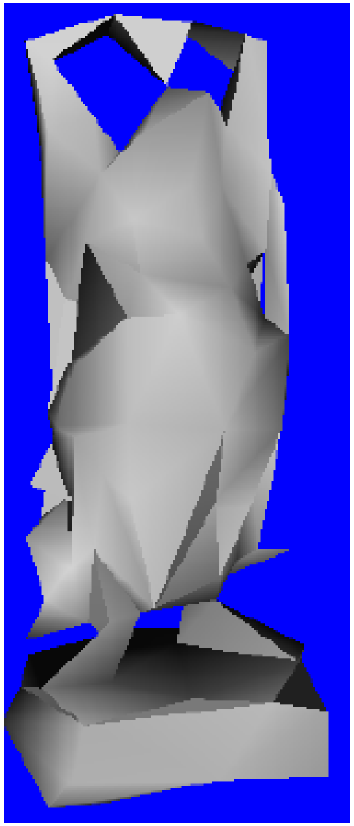


Examples



Original - 12,000

2,000 faces



298 faces (140 vertices)

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

1.3 Appearance-Preserving Simplification

Motivation

- Generalization required to handle appearance properties
  - color
  - texture
  - normals
  - etc.

Surface Properties as Vertex Attributes

- Each Vertex has a set of properties
  - Each property has one unique value per vertex
  - Attributes are linearly interpolated over faces
  - Primary example: one RGB color per vertex
- Can't treat geometry & color separately
  - Position and color are correlated
  - Optimal position may lie off the surface
  - Must synthesis new color for this position

Vertex Attributes Become Added Dimensions

- Treat each vertex as a 6-vector [x,y,z,r,g,b]
  - Assume this 6D space is Euclidean
    - Of course, color space is only roughly Euclidean
  - Scale xyz space to unit cube for consistency
- Triangle determines a 2-plane in 6D space
  - Can measure squared distance to this plane
  - Distance along all perpendicular directions
    - Generalized Pythagorean Theorem

Generalized Quadric Metric

- Squared distance to 2-plance has same form:

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

  - A: 6x6 matrix, v,b: 6-vectors c: scalar (for RGB)
  - Underlying algorithm remains the same

Generalized Quadric Metric

- Common property types

	Vertex	Dimension
Color	[x y z r g b]	6x6 quadrics
Texture	[x y z s t]	5x5 quadrics
Normal	[x y z u v w]	6x6 quadrics
Color+Normal	[x y z r g b u v w]	9x9 quadrics

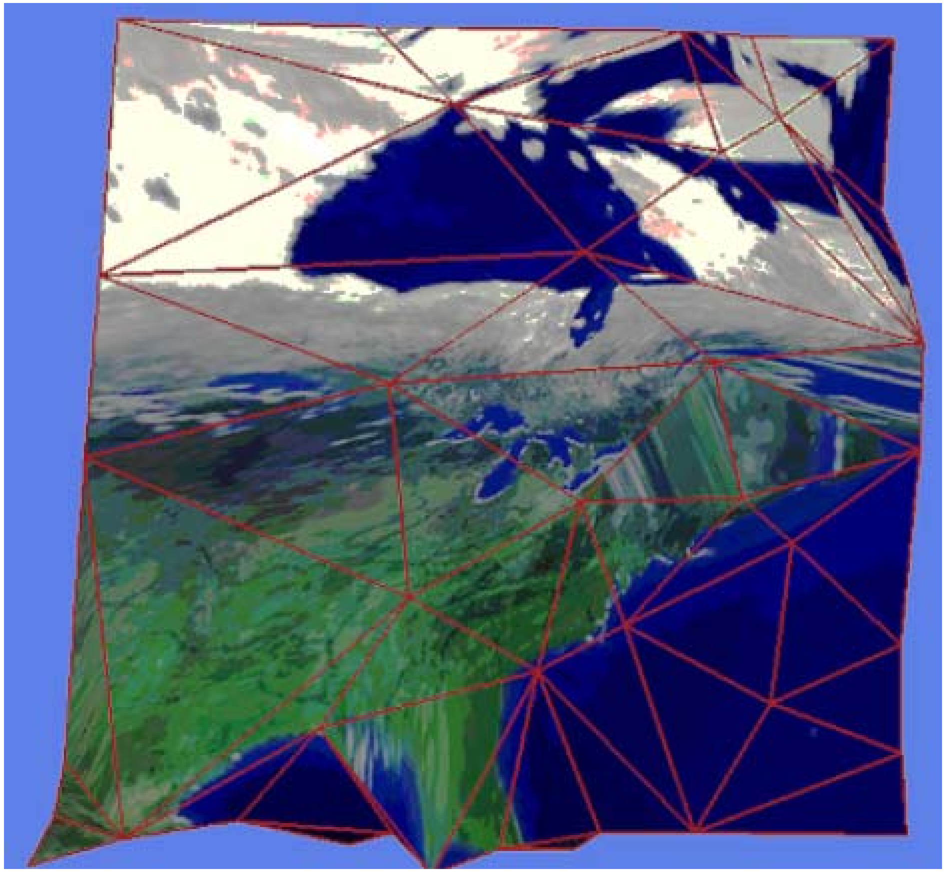
Example

50761 triangles

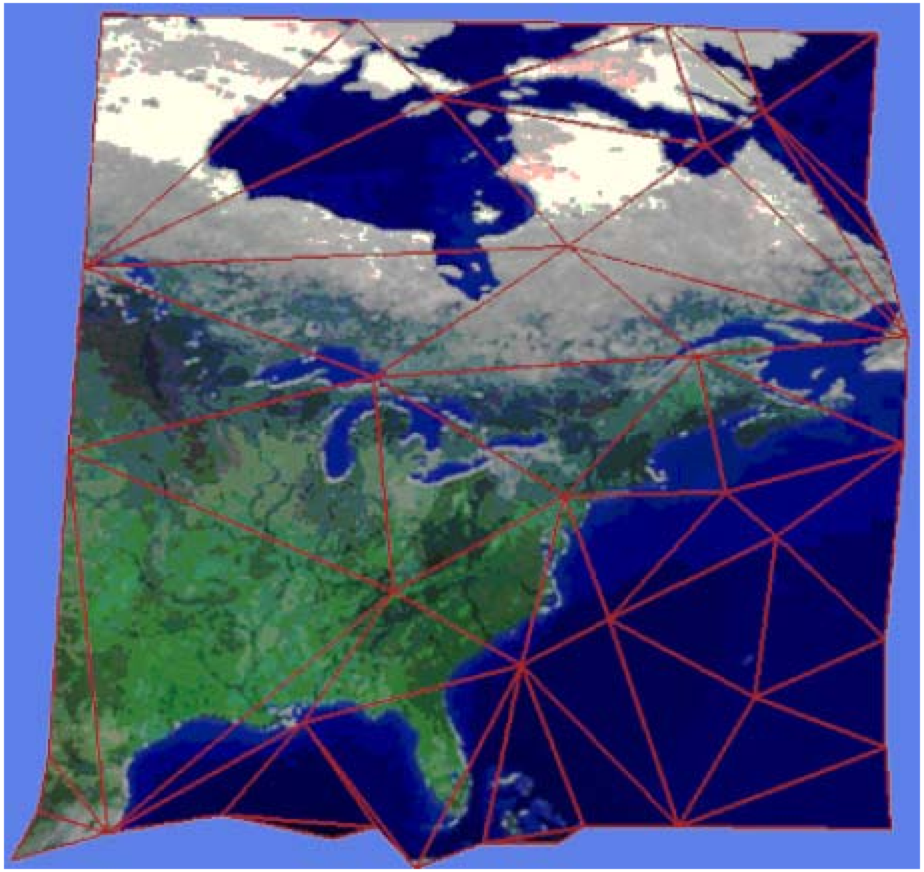
1500 triangles

A Sample Textured Surface

Comparison



Simplifying geometry only



Simplifying geometry + texture coordinates

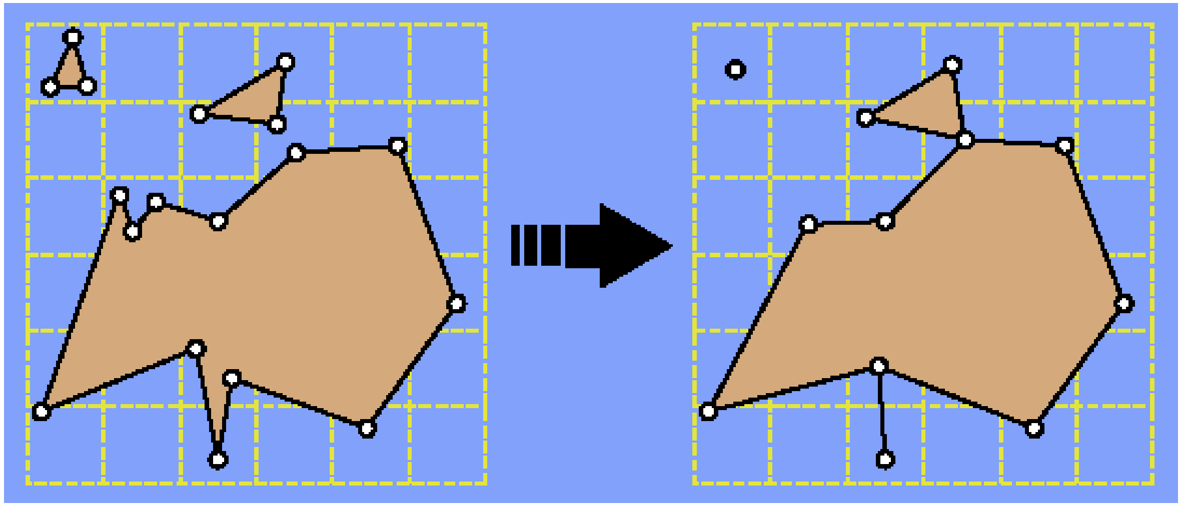
2. Global Simplification Strategies

Algorithms

- Vertex Clustering
- Re-Tiling
- Mesh optimization

2.1 Vertex Clustering

- Merge all vertices within the same cell



Steps

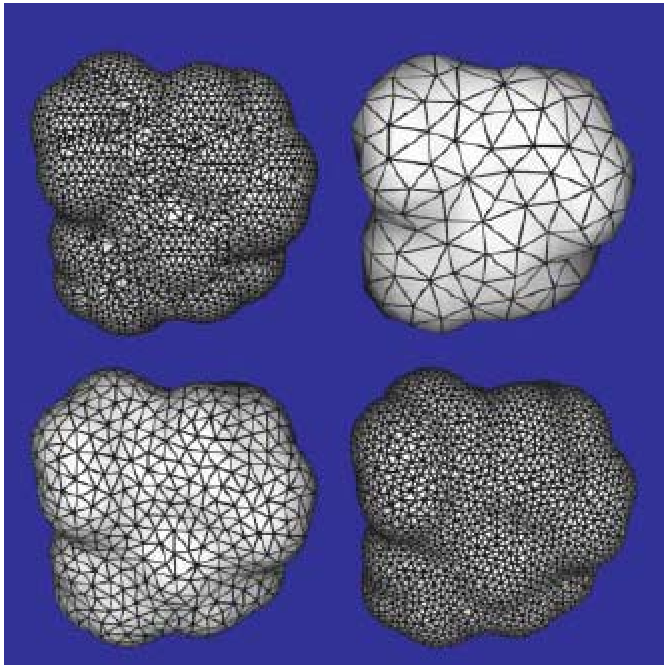
- Partition space into cells
  - grids [Rossignac-Borrel], spheres [Low-Tan], octrees, ...
- Merge all the vertices falling within a single cell together and replace with a single representative vertex
- Form triangles with resulting vertices that attempt to preserve the original topology

Pros and Cons

- Advantages
  - Does not require manifold models
  - Can handle multiple objects
  - Fast
- Disadvantages
  - Low quality
  - Hard to control

2.2 Mesh Re-Tiling [Turk 92]

- Re-tiling attempts to simplify as well as improve meshing by introducing new "uniformly spaced" vertices

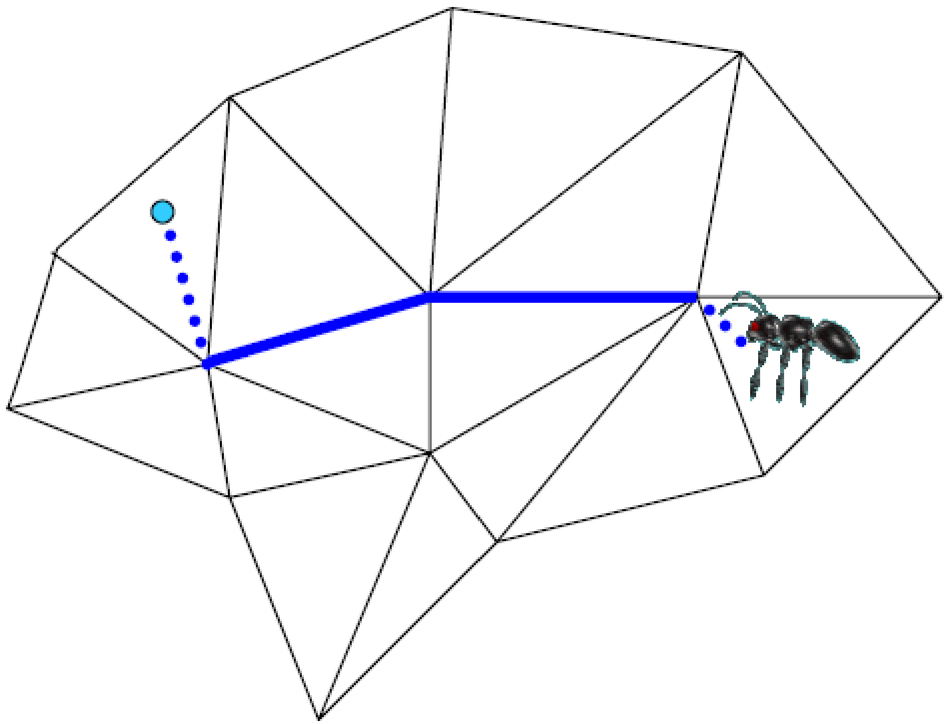


Steps

- Generate random points on surface
- Use a diffusion/repulsion to spread the points out uniformly
- Add new set of points to the surface and mutually tessellate
- Remove old vertices one by one yielding a new triangulation

Geodesic Distances

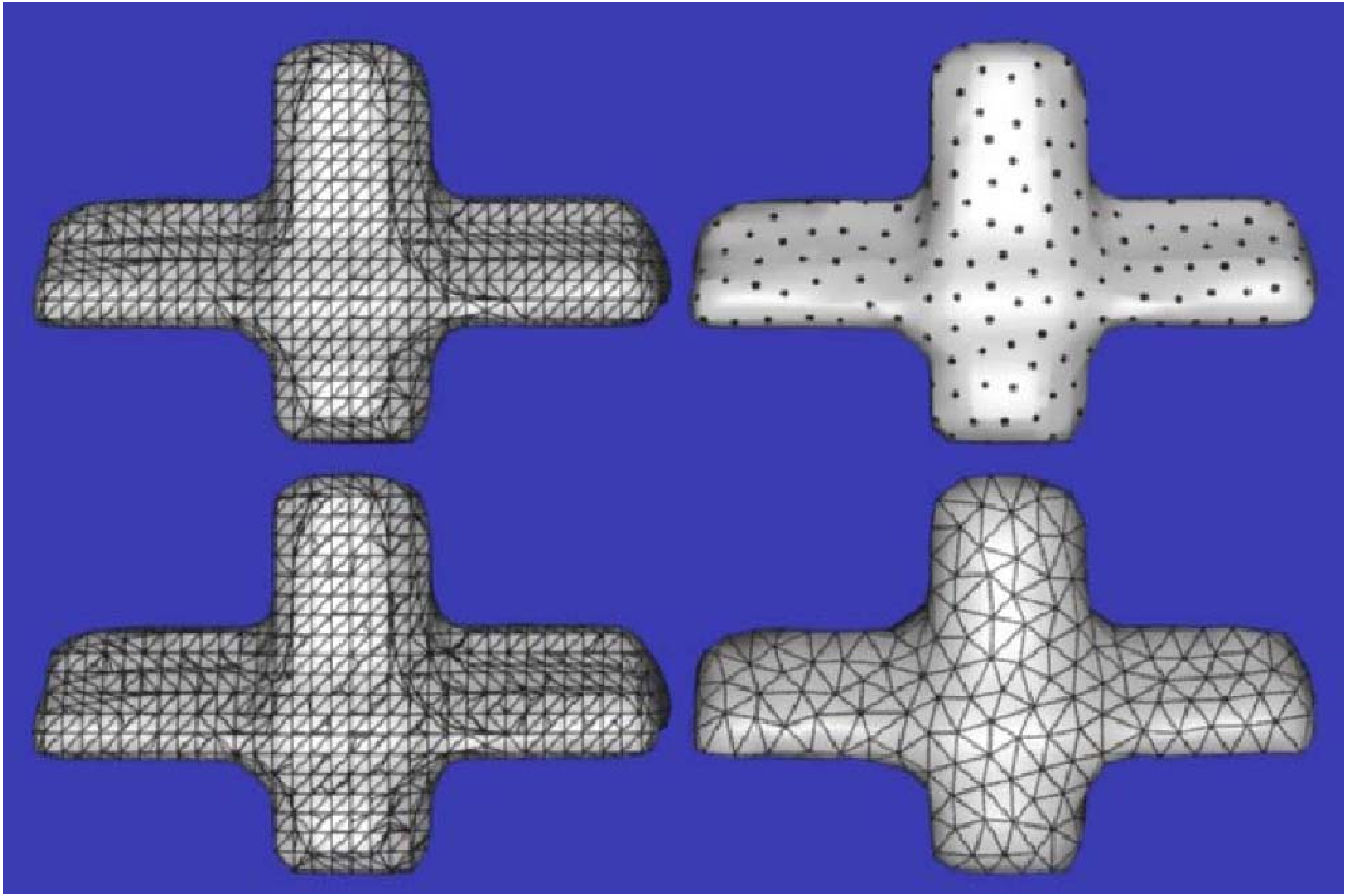
- Shortest path "on the manifold" between two points



Pros and Cons

- Advantages
  - High quality triangles
  - Maintains topology
- Disadvantages
  - Slow
  - Tends to blur sharp features (resampling)

Re-Tiling Example

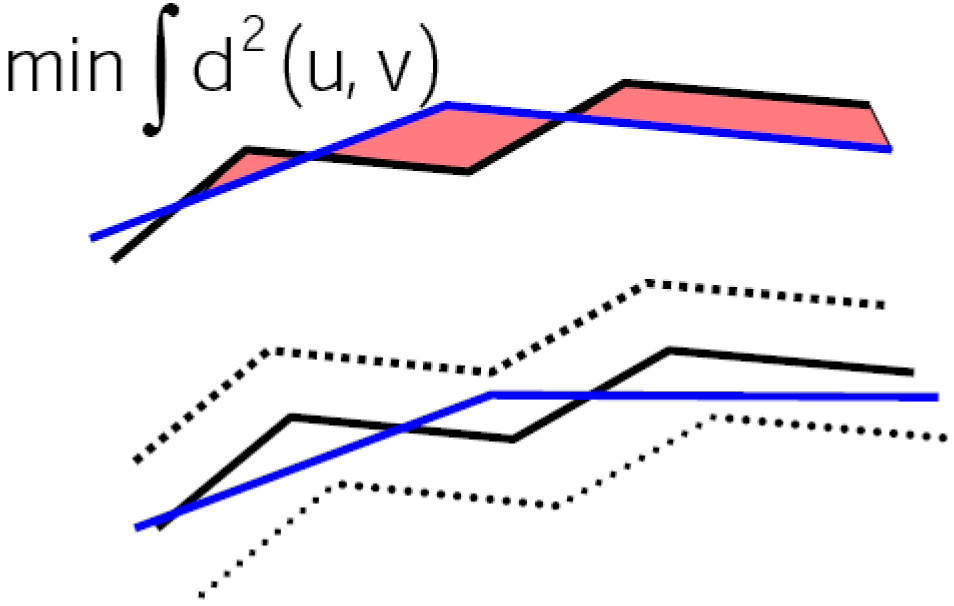


2.3 Mesh Optimization [Hoppe et al 93]

- Frames simplification as an optimization problem
  - Minimizes some energy function
  - Make simple changes to the topology of the mesh
  - Evaluate the energy before and after the change
  - Accept any change that reduces the energy

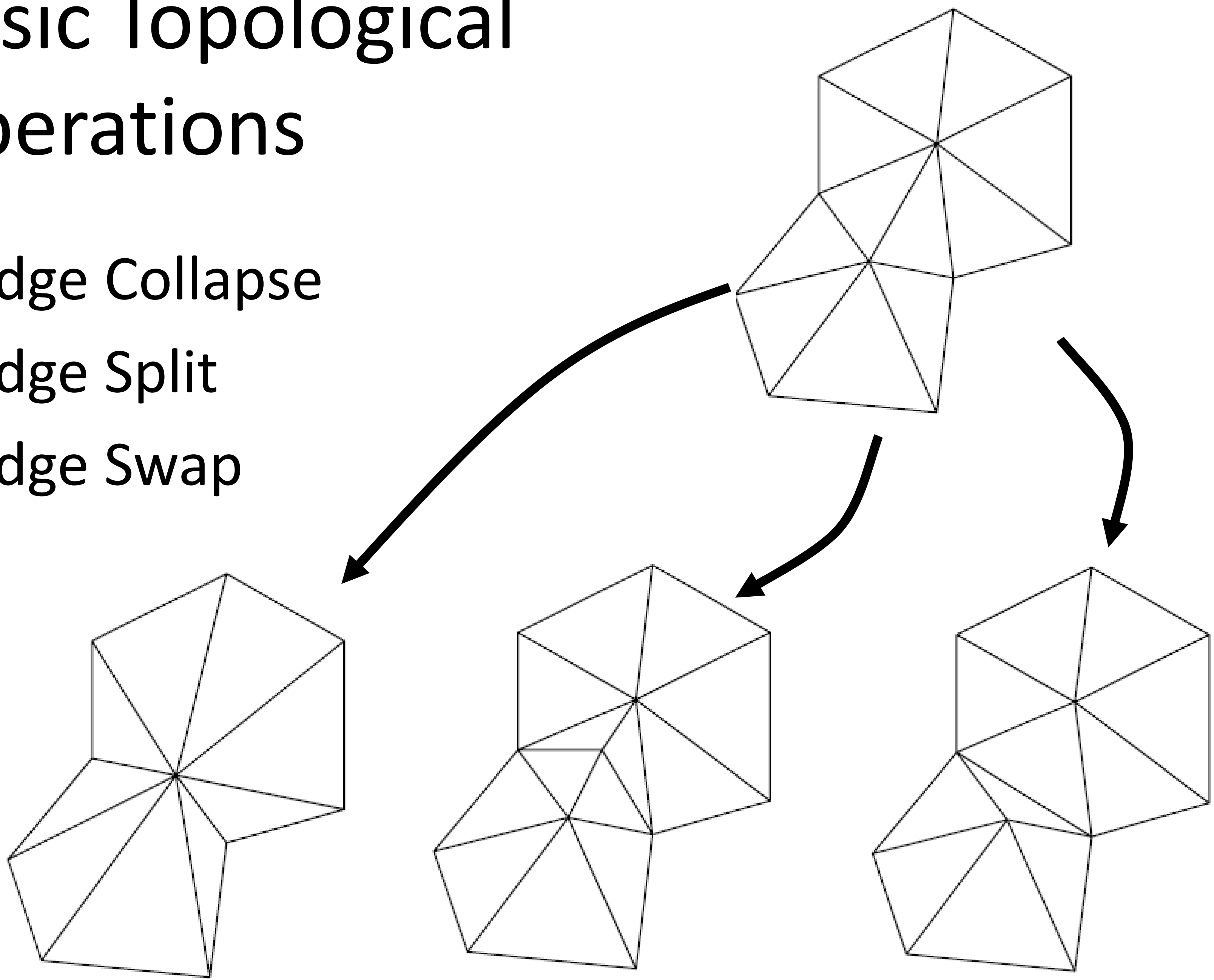
Energy Functions

- Geometric measures
- Topological measures
- Localized fits



Basic Topological Operations

- Edge Collapse
- Edge Split
- Edge Swap



Discussions