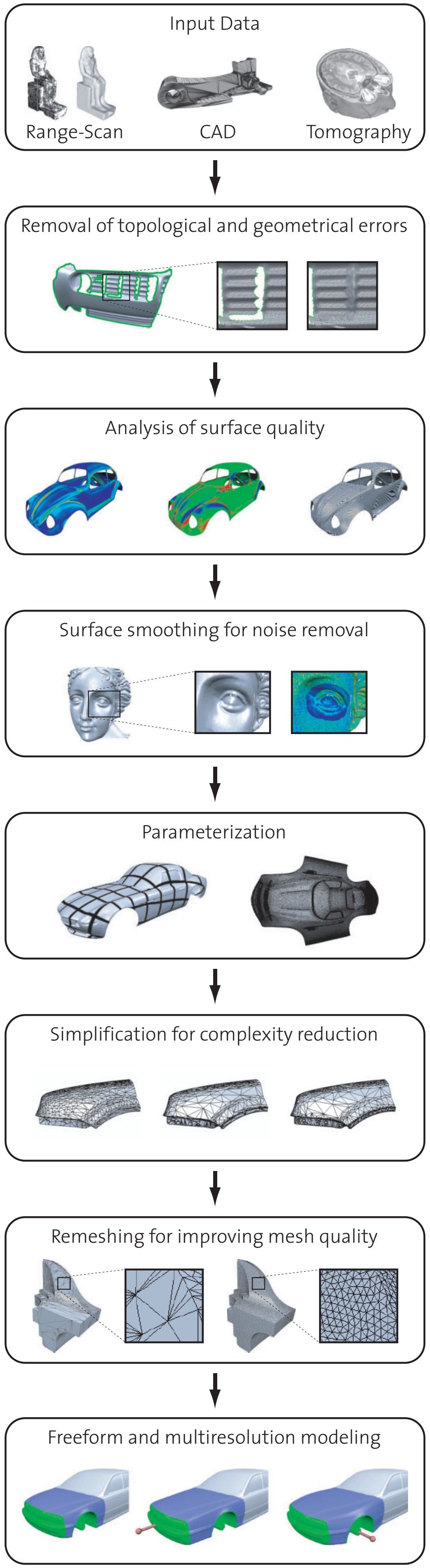


ETH
APPLIED
GEOMETRY
GROUP

Mesh Decimation

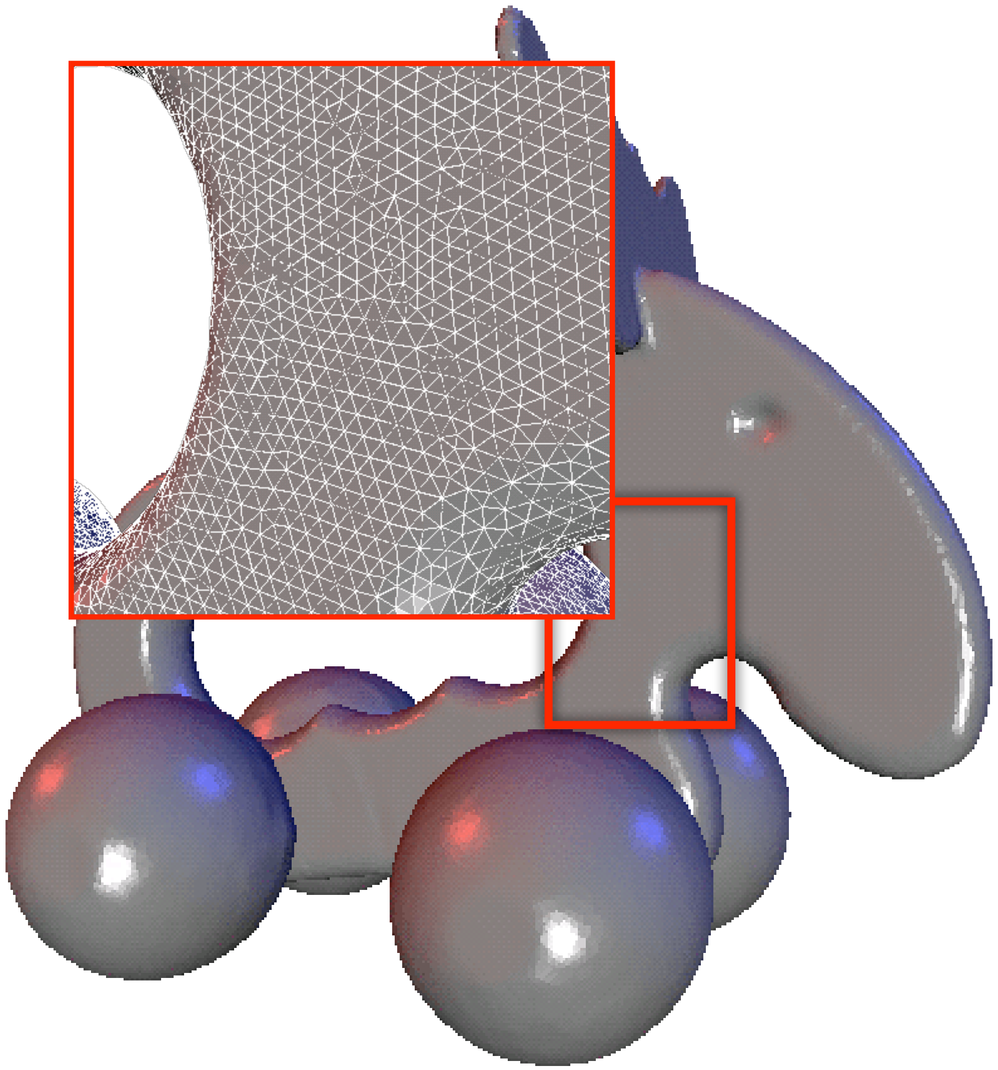
Mark Pauly

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

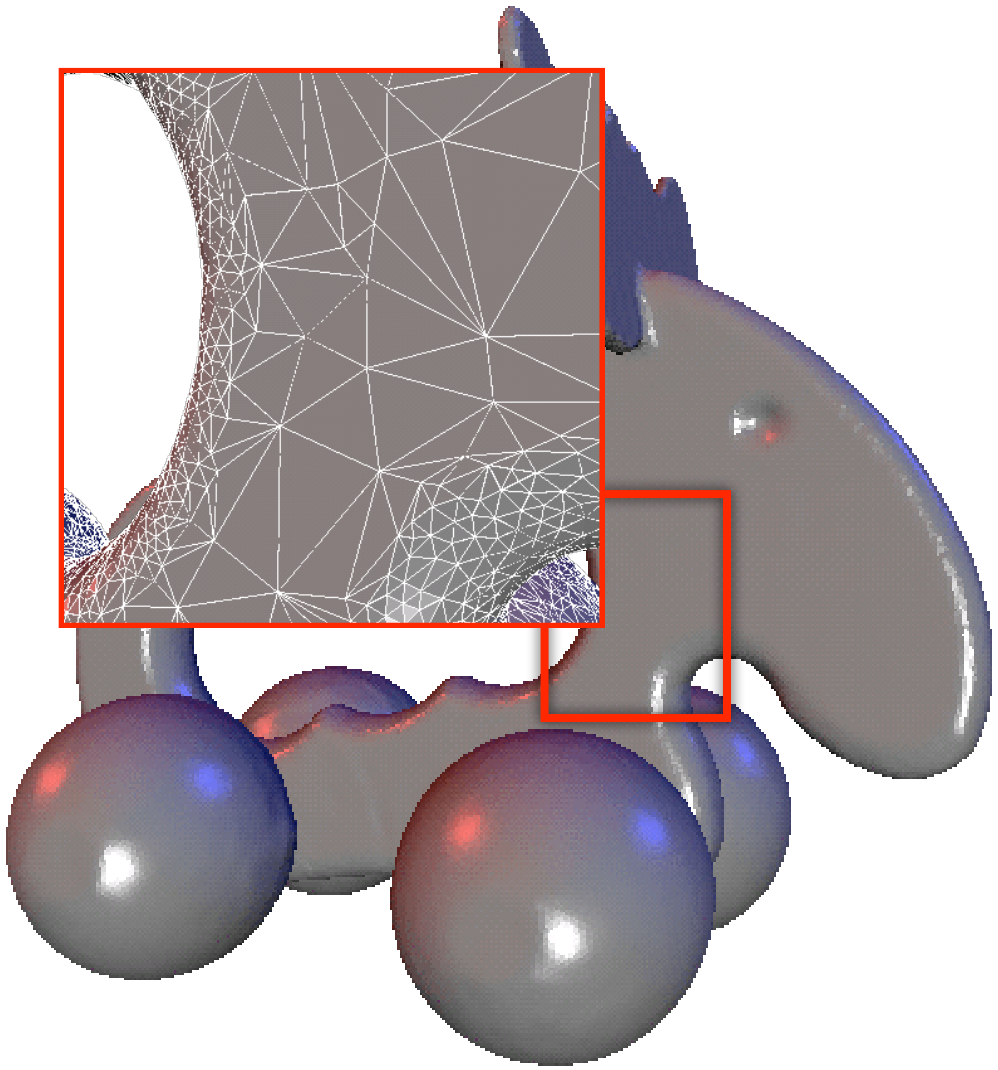
SIGGRAPH2006

Applications

- Oversampled 3D scan data



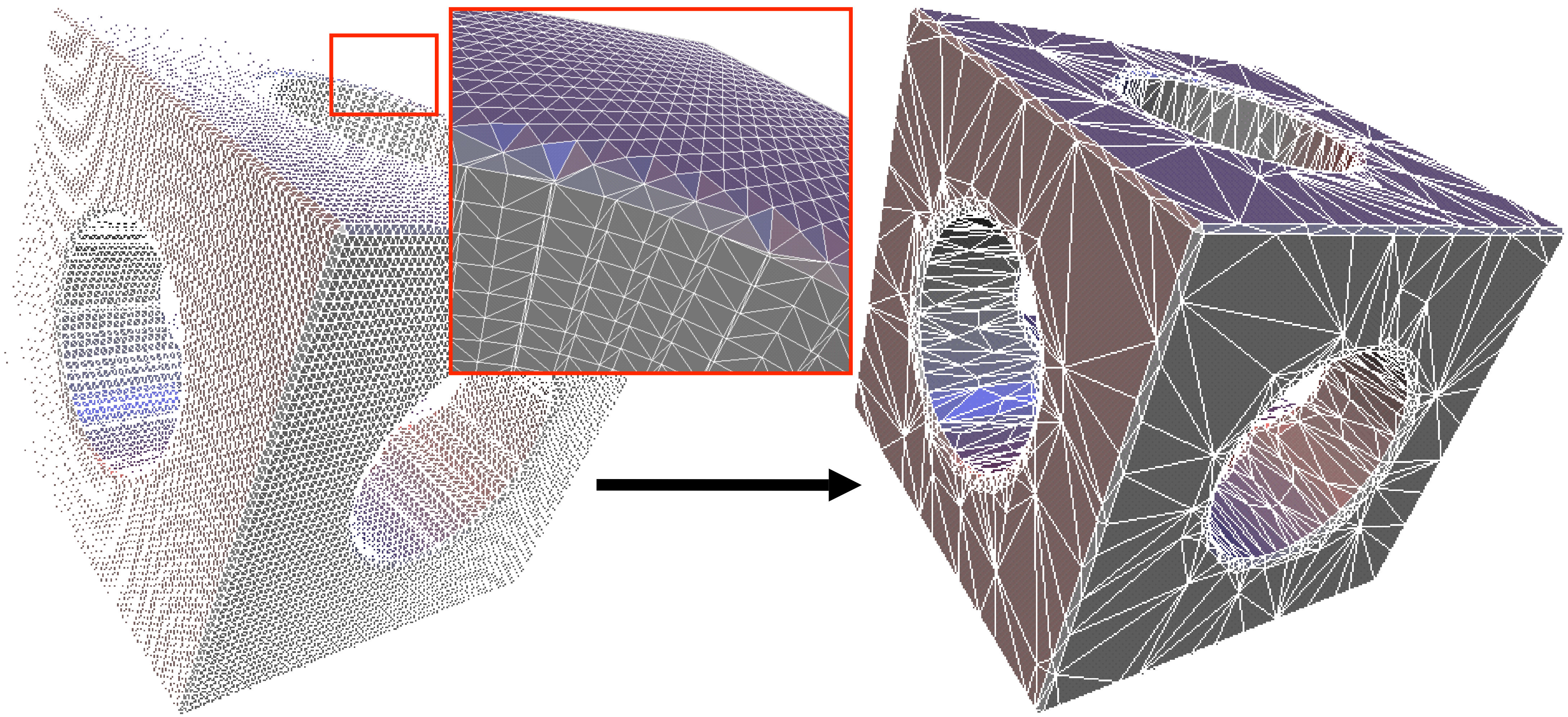
~150k triangles



~80k triangles

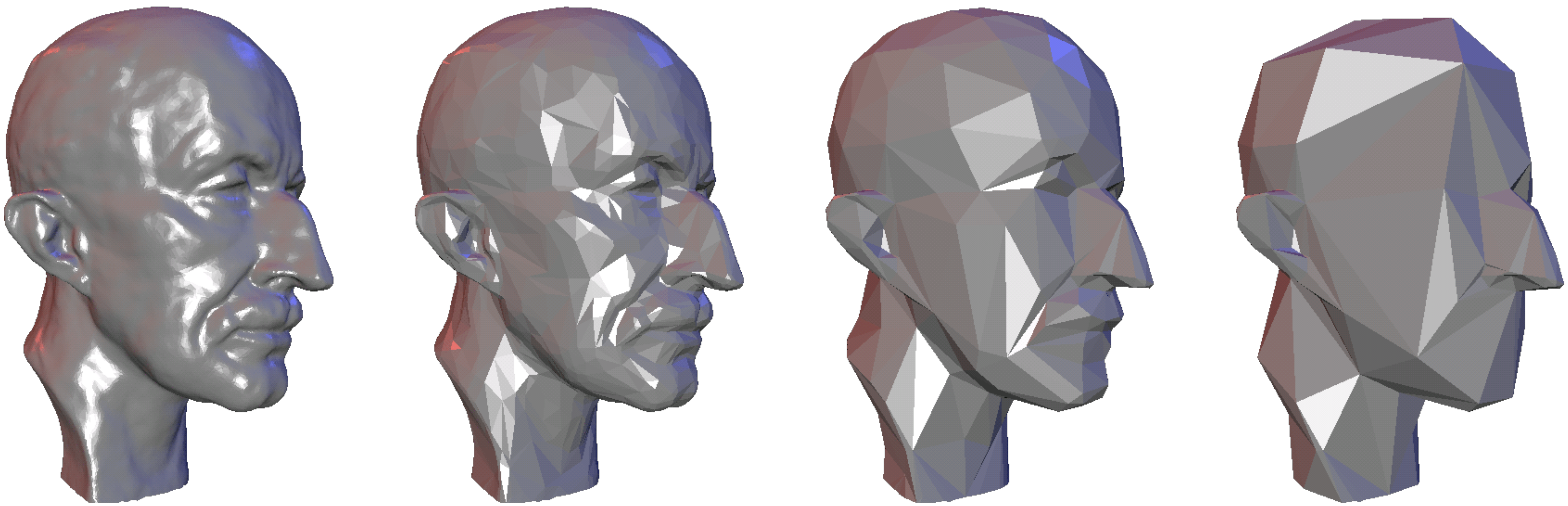
Applications

- Overtessellation: E.g. iso-surface extraction



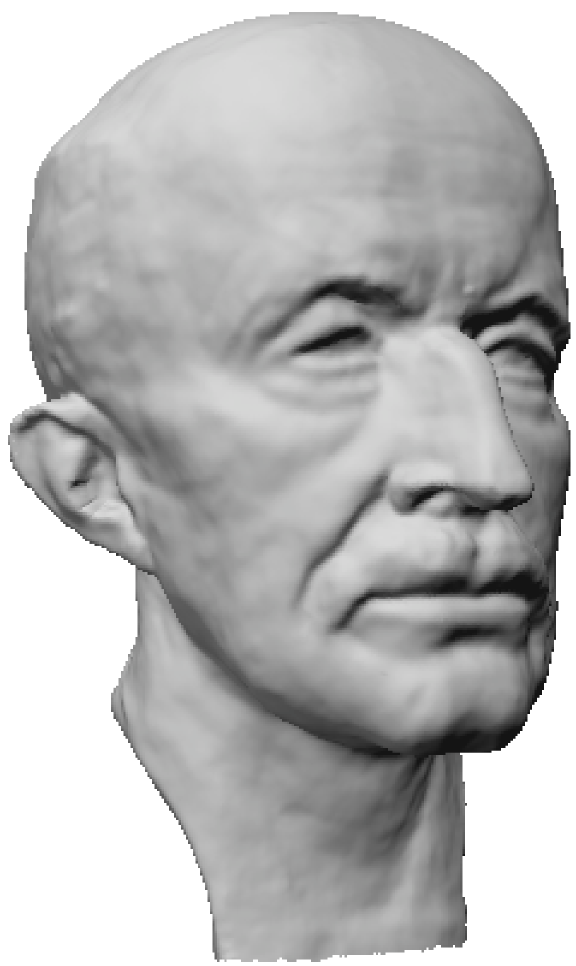
Applications

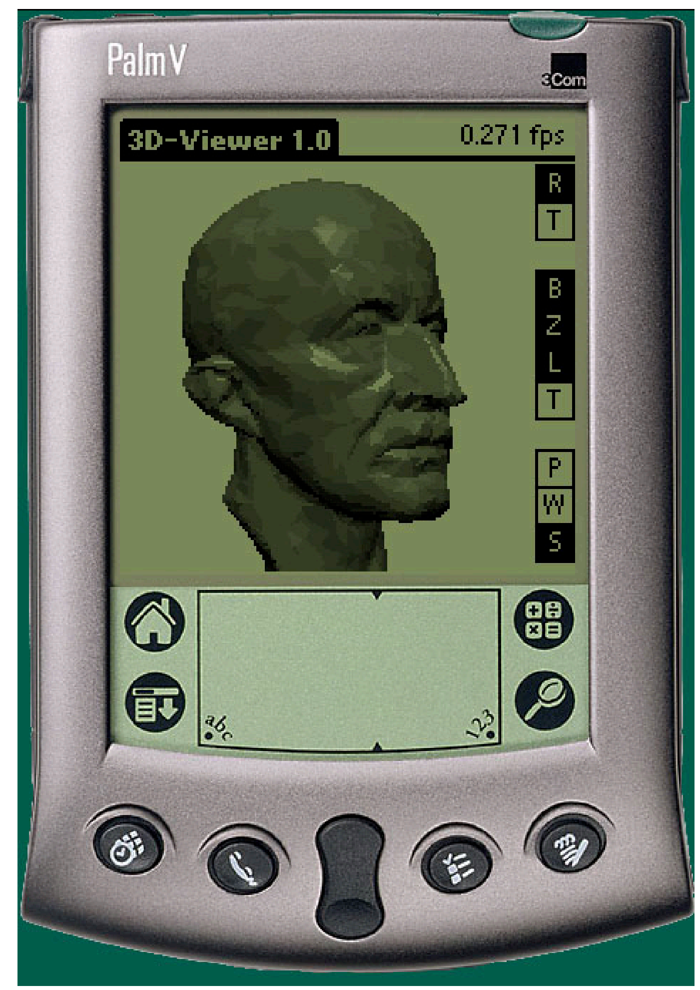
- Multi-resolution hierarchies for
  - efficient geometry processing
  - level-of-detail (LOD) rendering



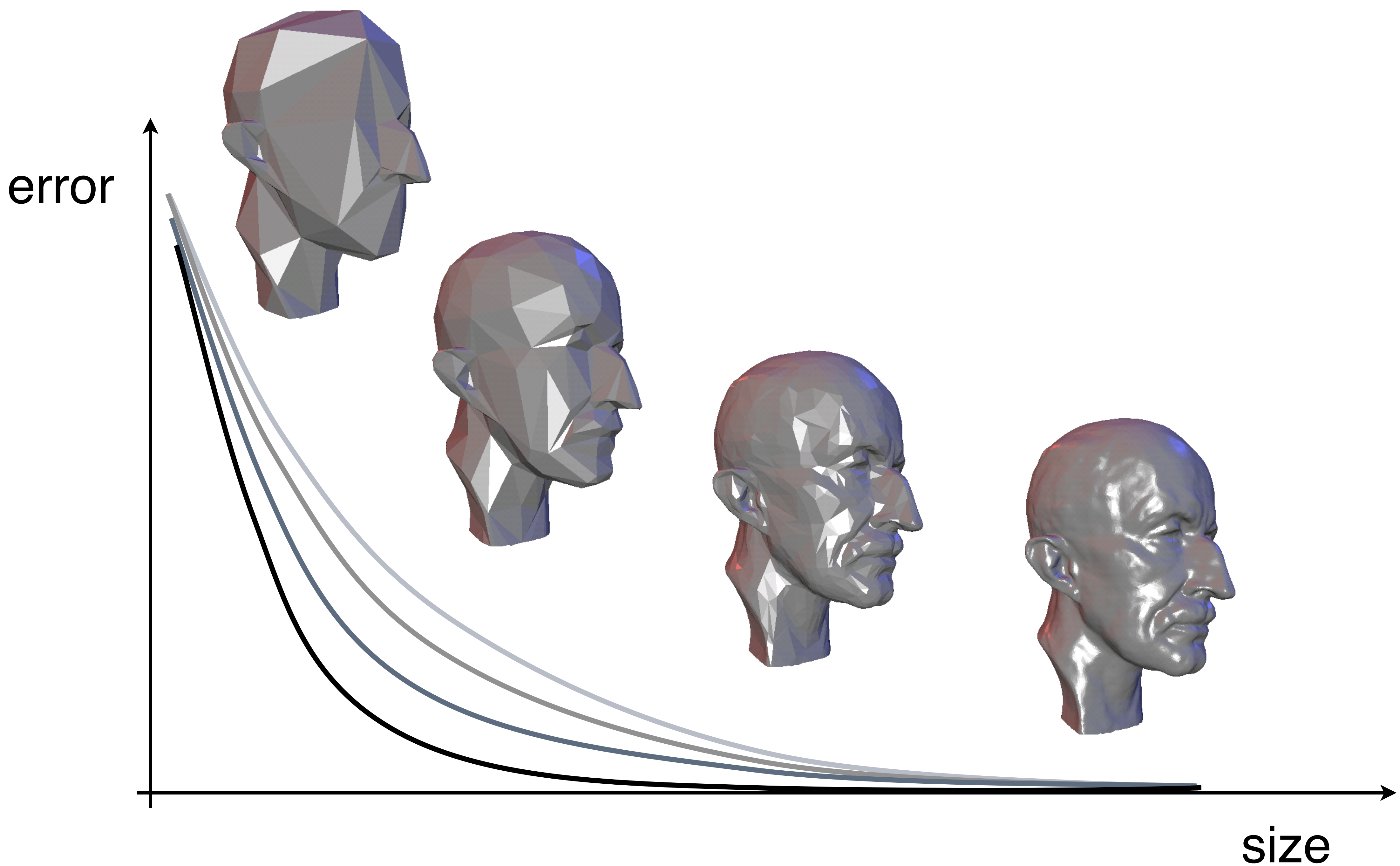
Applications

- Adaptation to hardware capabilities





Size-Quality Tradeoff



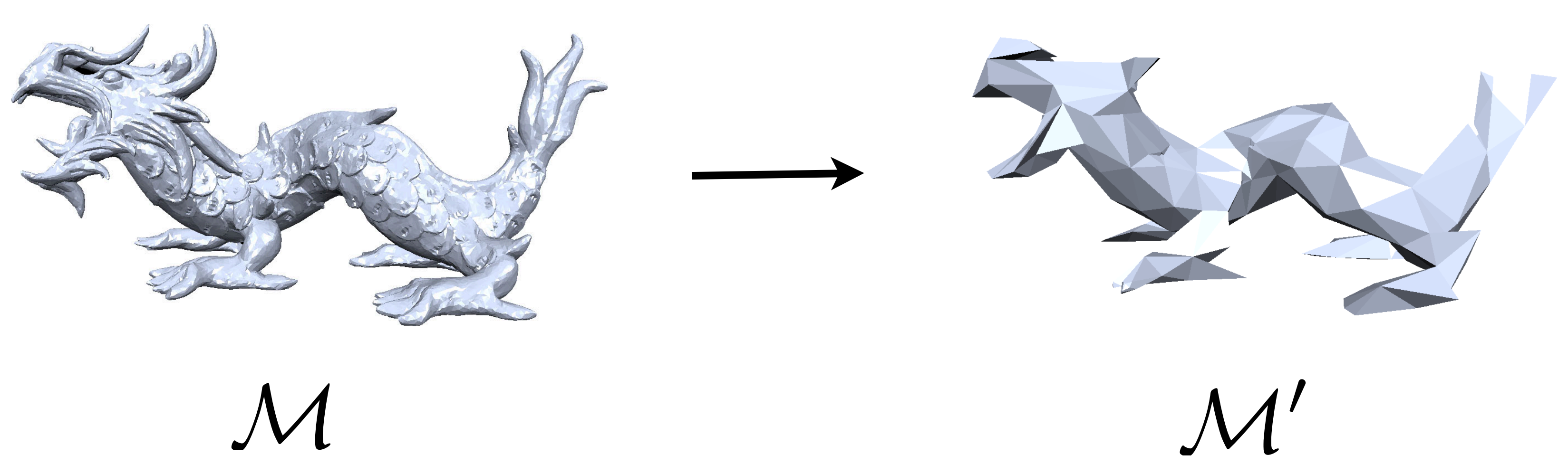
Outline

- Applications
- Problem Statement
- Mesh Decimation Methods
  - Vertex Clustering
  - Iterative Decimation
  - Extensions
  - Remeshing
  - Variational Shape Approximation

Problem Statement

- Given: $\mathcal{M} = (\mathcal{V}, \mathcal{F})$
- Find: $\mathcal{M}' = (\mathcal{V}', \mathcal{F}')$ such that

1. $|\mathcal{V}'| = n < |\mathcal{V}|$ and $\|\mathcal{M} - \mathcal{M}'\|$ is minimal, or
2. $\|\mathcal{M} - \mathcal{M}'\| < \epsilon$ and $|\mathcal{V}'|$ is minimal



Problem Statement

- Given: $\mathcal{M} = (\mathcal{V}, \mathcal{F})$
- Find: $\mathcal{M}' = (\mathcal{V}', \mathcal{F}')$ such that

1. $|\mathcal{V}'| = n < |\mathcal{V}|$ and $\|\mathcal{M} - \mathcal{M}'\|$ is minimal, or
2. $\|\mathcal{M} - \mathcal{M}'\| < \epsilon$ and $|\mathcal{V}'|$ is minimal

hard!

→ look for sub-optimal solution

Problem Statement

- Given: $\mathcal{M} = (\mathcal{V}, \mathcal{F})$
- Find: $\mathcal{M}' = (\mathcal{V}', \mathcal{F}')$ such that

  1. $|\mathcal{V}'| = n < |\mathcal{V}|$ and $\|\mathcal{M} - \mathcal{M}'\|$ is minimal, or
  2. $\|\mathcal{M} - \mathcal{M}'\| < \epsilon$ and $|\mathcal{V}'|$ is minimal

- Respect additional fairness criteria
  - normal deviation, triangle shape, scalar attributes, etc.

Outline

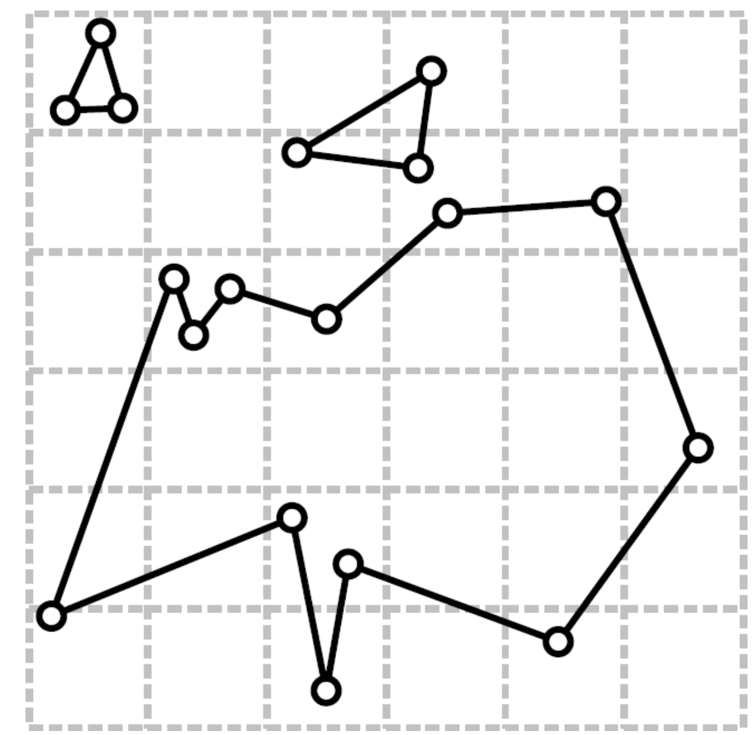
- Applications
- Problem Statement
- Mesh Decimation Methods
  - **Vertex Clustering**
  - Iterative Decimation
  - Extensions

Vertex Clustering

- Cluster Generation
- Computing a representative
- Mesh generation
- Topology changes

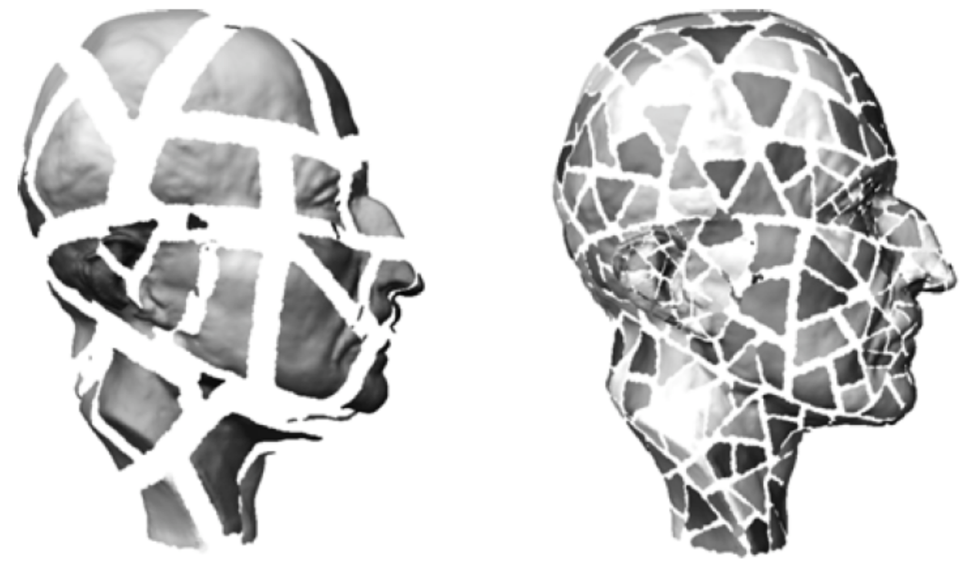
Vertex Clustering

- Cluster Generation
  - Uniform 3D grid
  - Map vertices to cluster cells
- Computing a representative
- Mesh generation
- Topology changes



Vertex Clustering

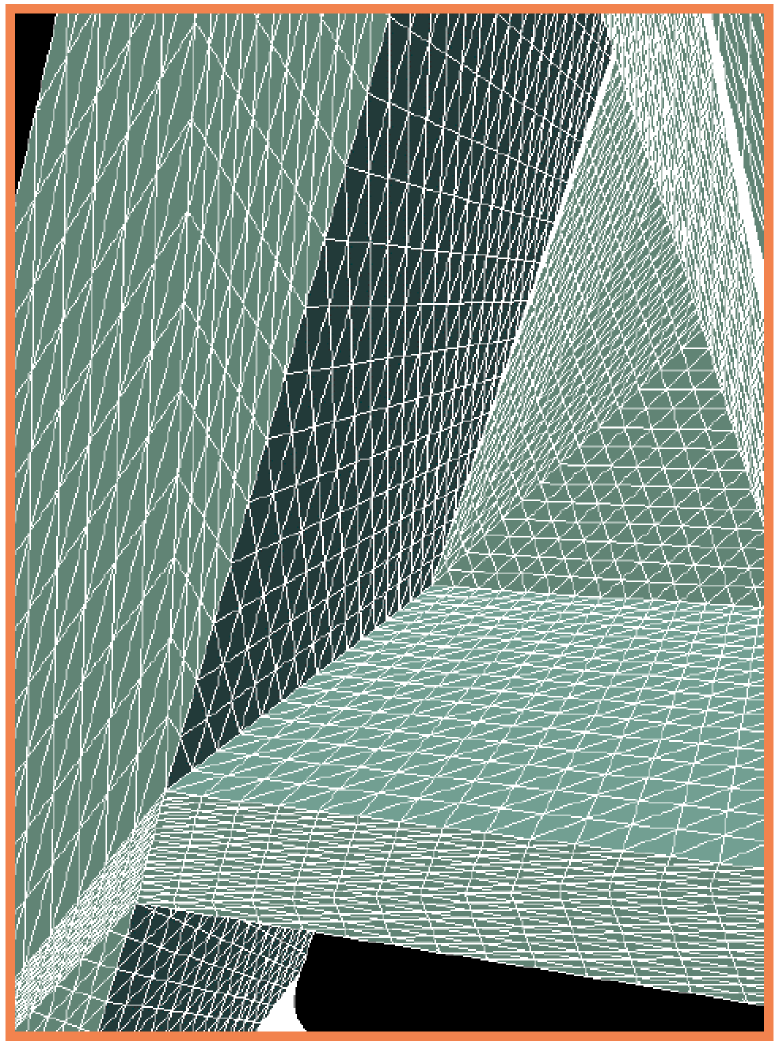
- Cluster Generation
  - Hierarchical approach
  - Top-down or bottom-up
- Computing a representative
- Mesh generation
- Topology changes

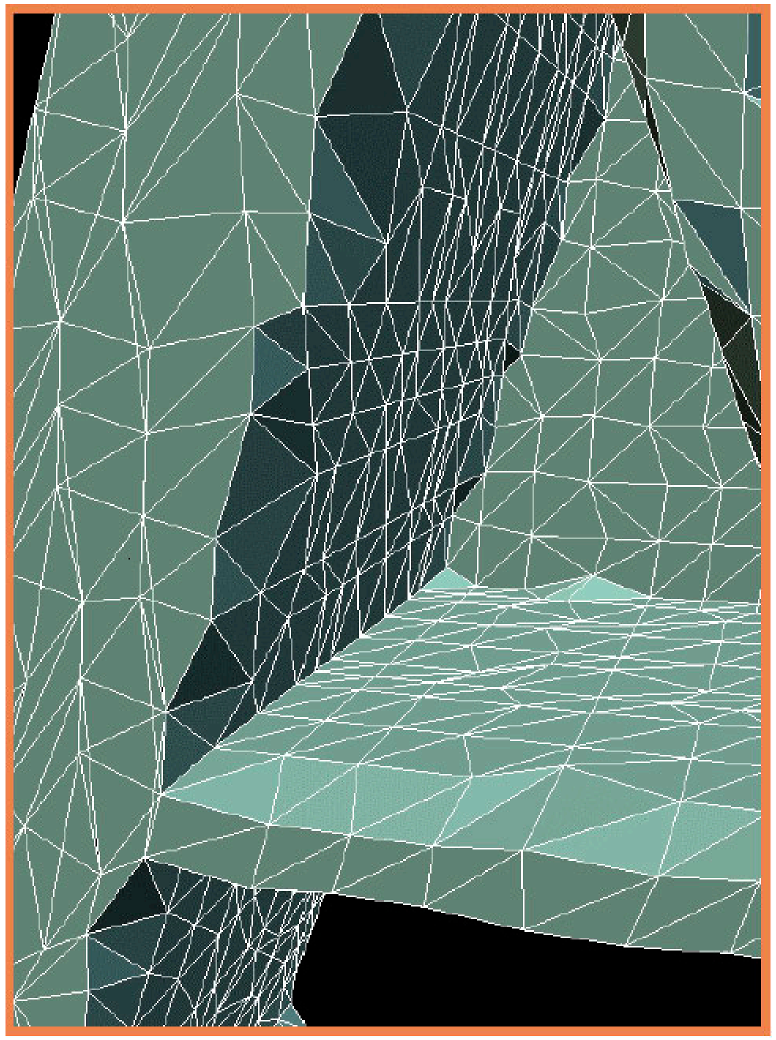


Vertex Clustering

- Cluster Generation
- Computing a representative
  - Average/median vertex position
  - Error quadrics
- Mesh generation
- Topology changes

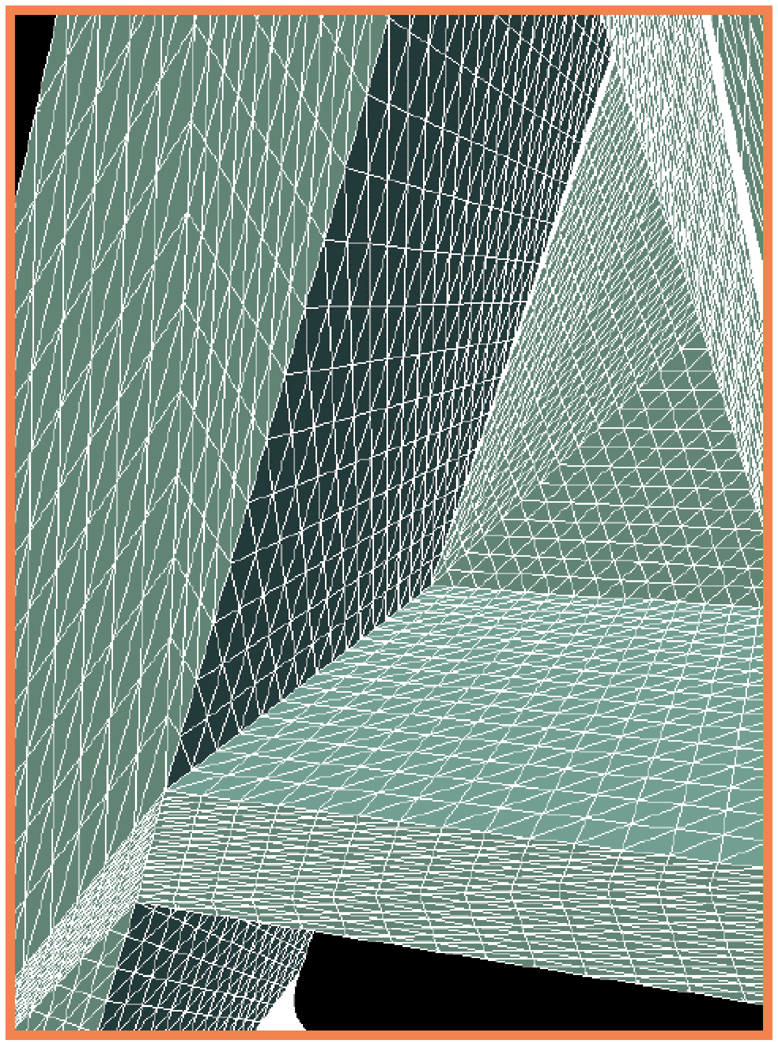
Computing a Representative

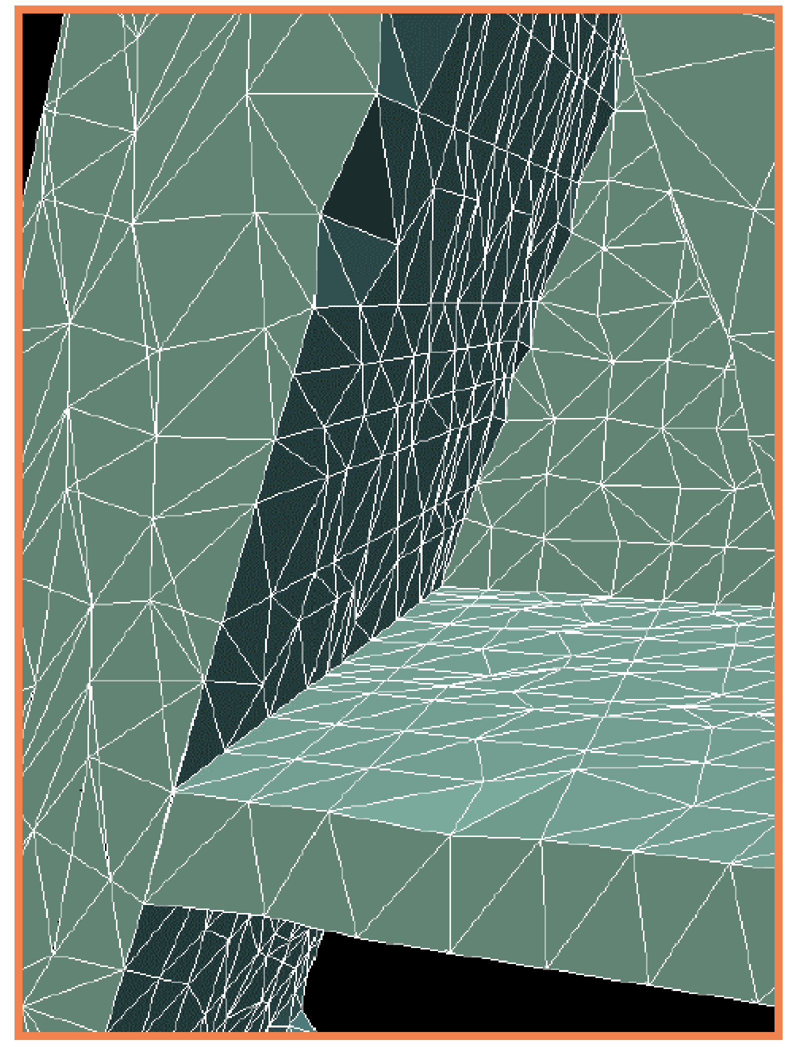




Average vertex position → Low-pass filter

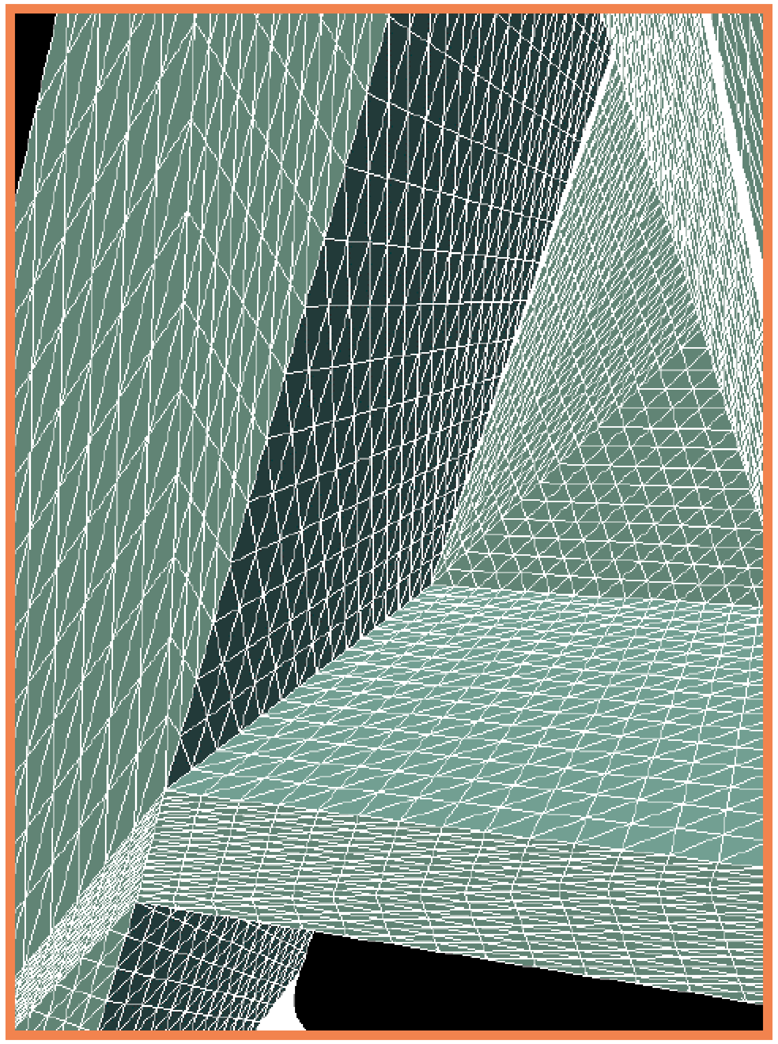
Computing a Representative

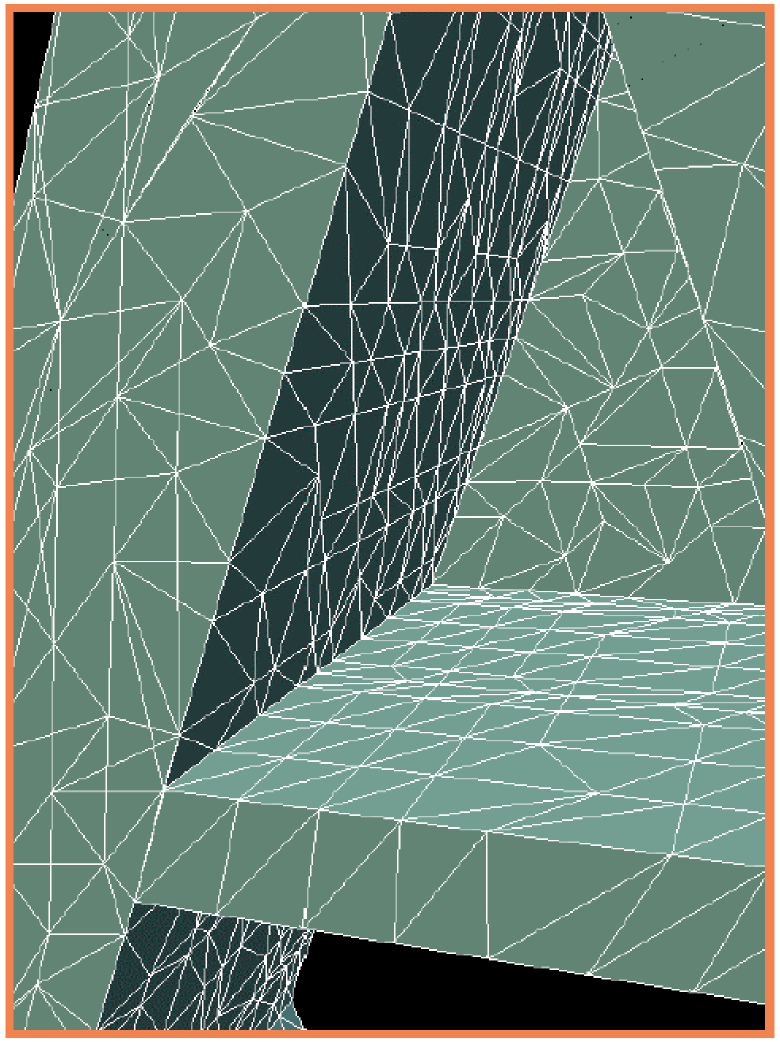




Median vertex position → Sub-sampling

Computing a Representative





Error quadrics

Error Quadrics

- Squared distance to plane

$$p = (x, y, z, 1)^T, \quad q = (a, b, c, d)^T$$

$$dist(q,p)^2 = (q^T p)^2 = p^T (qq^T) p =: p^T Q_q p$$

$$Q_q = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & b^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix}$$

Error Quadrics

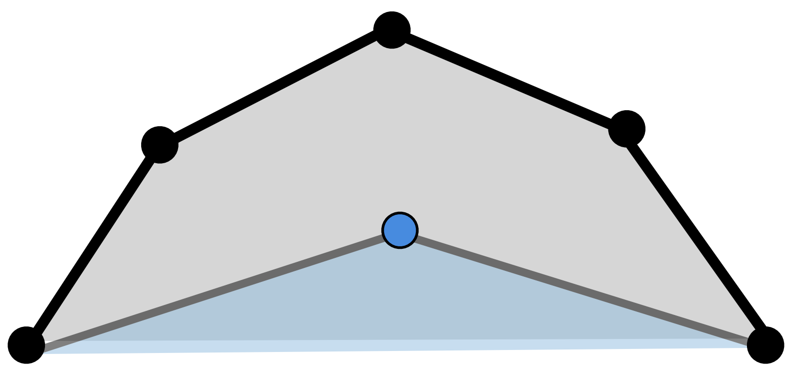
- Sum distances to vertex' planes

$$\sum_i dist(q_i, p)^2 = \sum_i p^T Q_{q_i} p = p^T \left(\sum_i Q_{q_i} \right) p =: p^T Q_p p$$

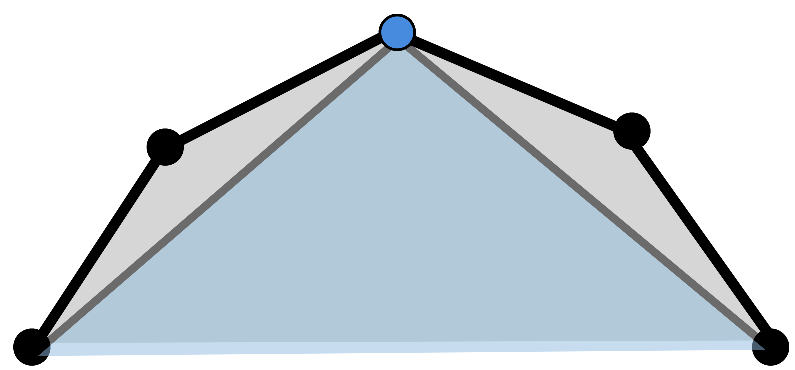
- Point that minimizes the error

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

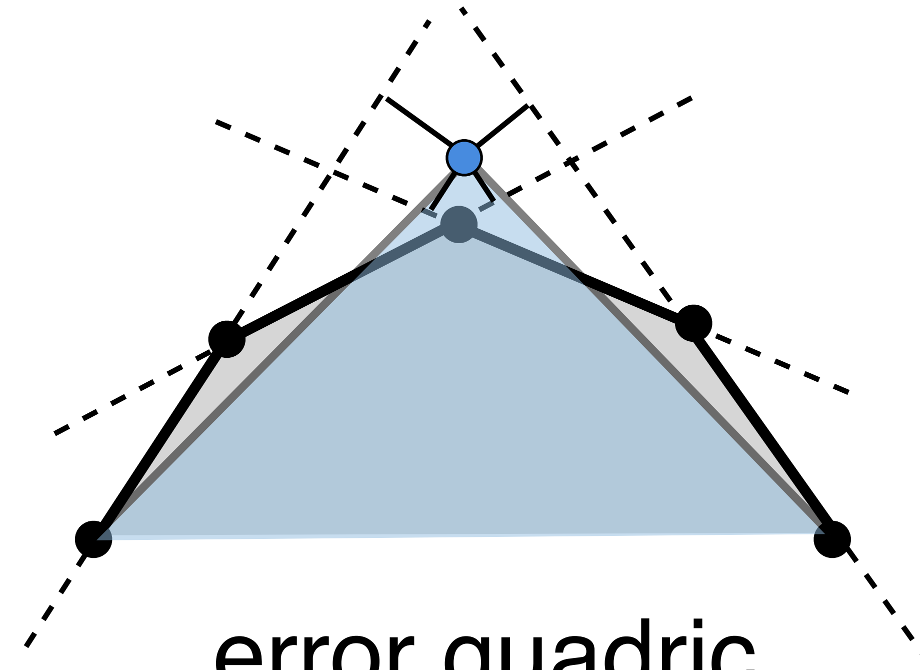
Comparison



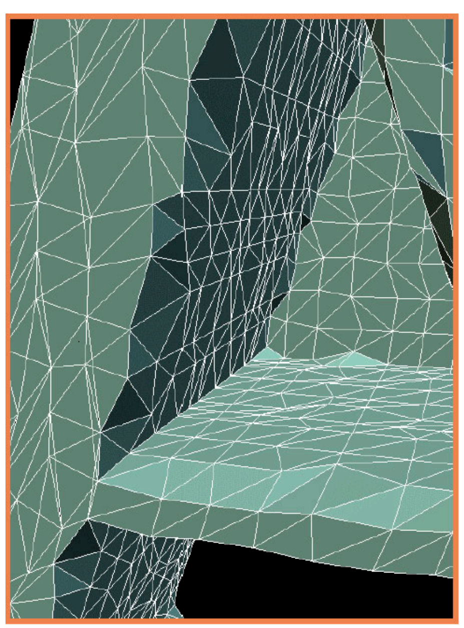
average

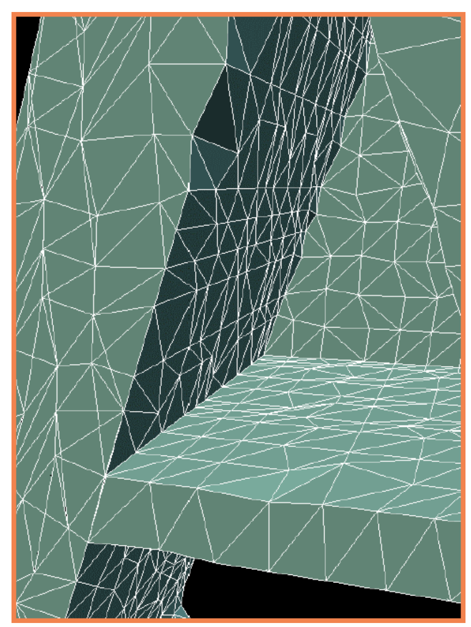


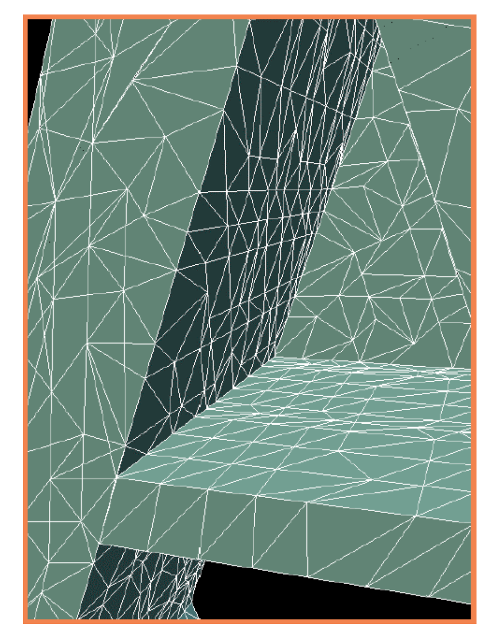
median



error quadric





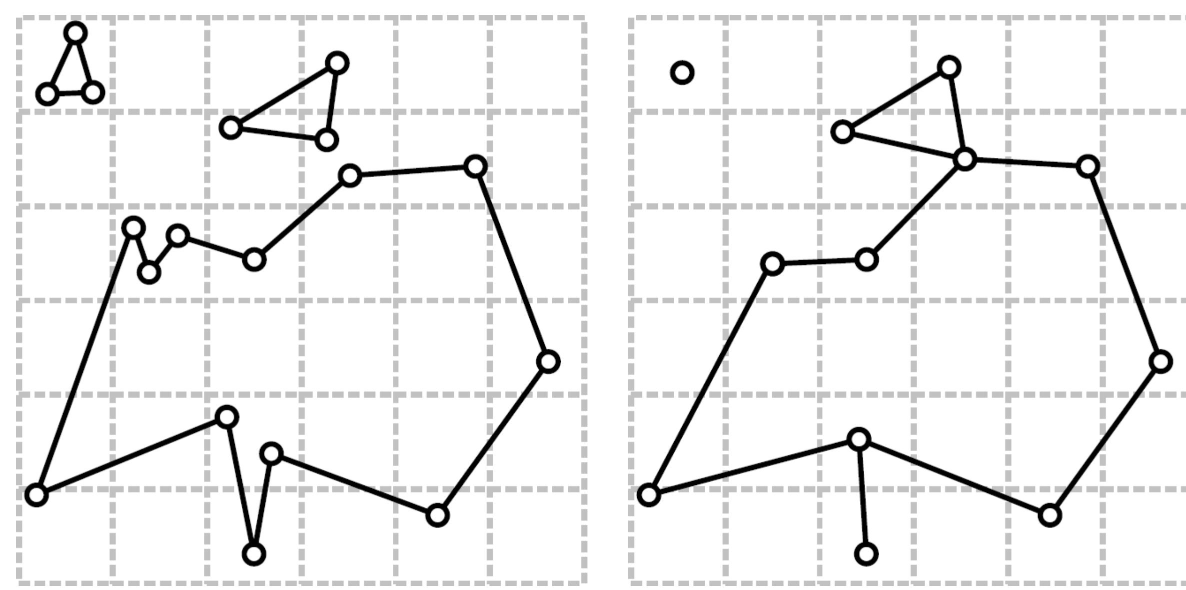


Vertex Clustering

- Cluster Generation
- Computing a representative
- Mesh generation
  - Clusters $p \Leftrightarrow \{p_0,\ldots,p_n\}$, $q \Leftrightarrow \{q_0,\ldots,q_m\}$
  - Connect (p,q) if there was an edge (p_i,q_j)
- Topology changes

Vertex Clustering

- Cluster Generation
- Computing a representative
- Mesh generation
- Topology changes
  - If different sheets pass through one cell
  - Not manifold



Outline

- Applications
- Problem Statement
- Mesh Decimation Methods
  - Vertex Clustering
  - **Iterative Decimation**
  - Extensions

Incremental Decimation

- General Setup
- Decimation operators
- Error metrics
- Fairness criteria
- Topology changes

General Setup

```
Repeat:
  pick mesh region
  apply decimation operator
Until no further reduction possible
```

Greedy Optimization

```
For each region
  evaluate quality after decimation
  enqueue(quality, region)

Repeat:
  pick best mesh region
  apply decimation operator
  update queue
Until no further reduction possible
```

Global Error Control

```
For each region
  evaluate quality after decimation
  enqeue(quality, region)

Repeat:
  pick best mesh region
  if error < ϵ
     apply decimation operator
     update queue
Until no further reduction possible
```

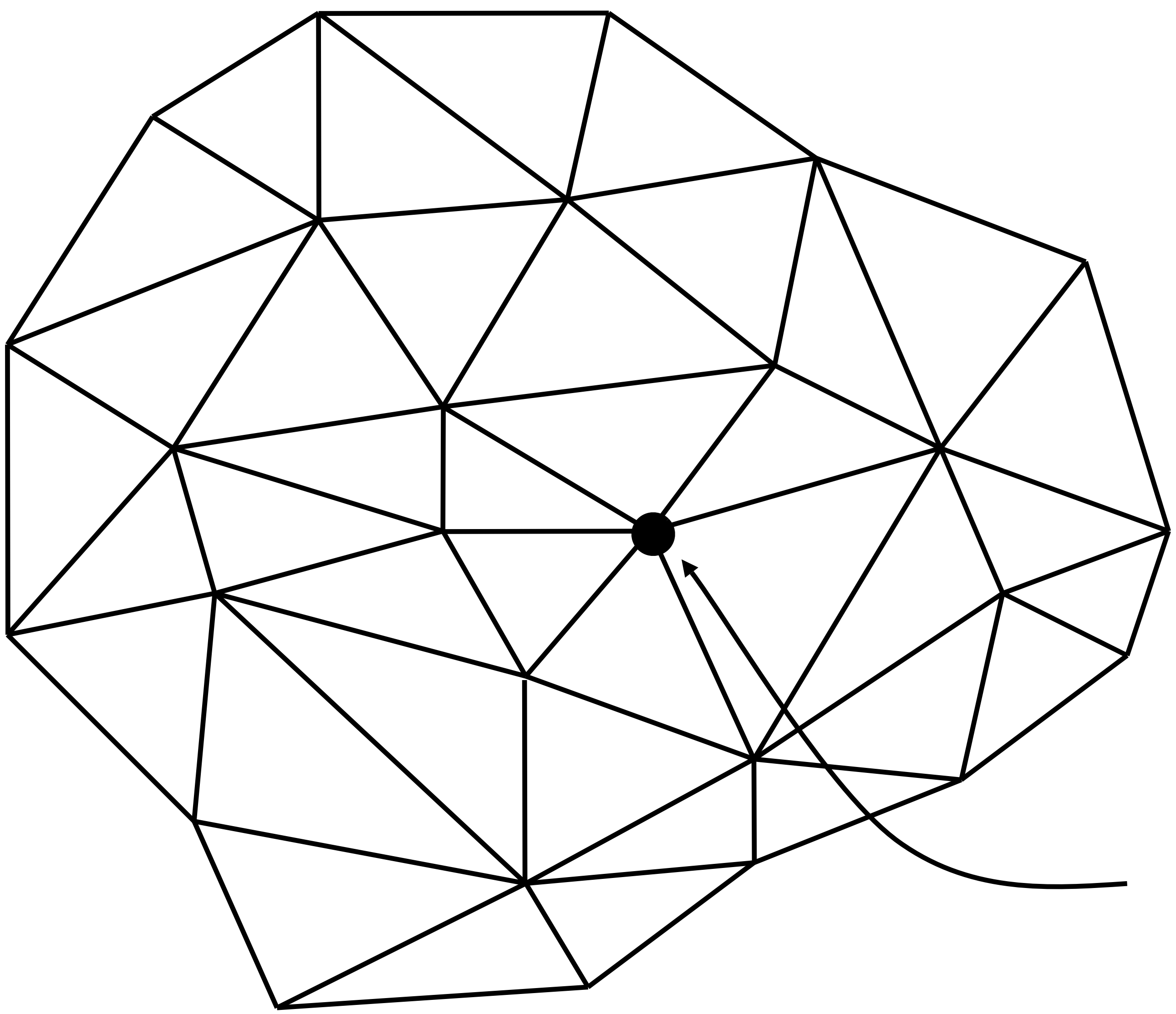
Incremental Decimation

- General Setup
- Decimation operators
- Error metrics
- Fairness criteria
- Topology changes

Decimation Operators

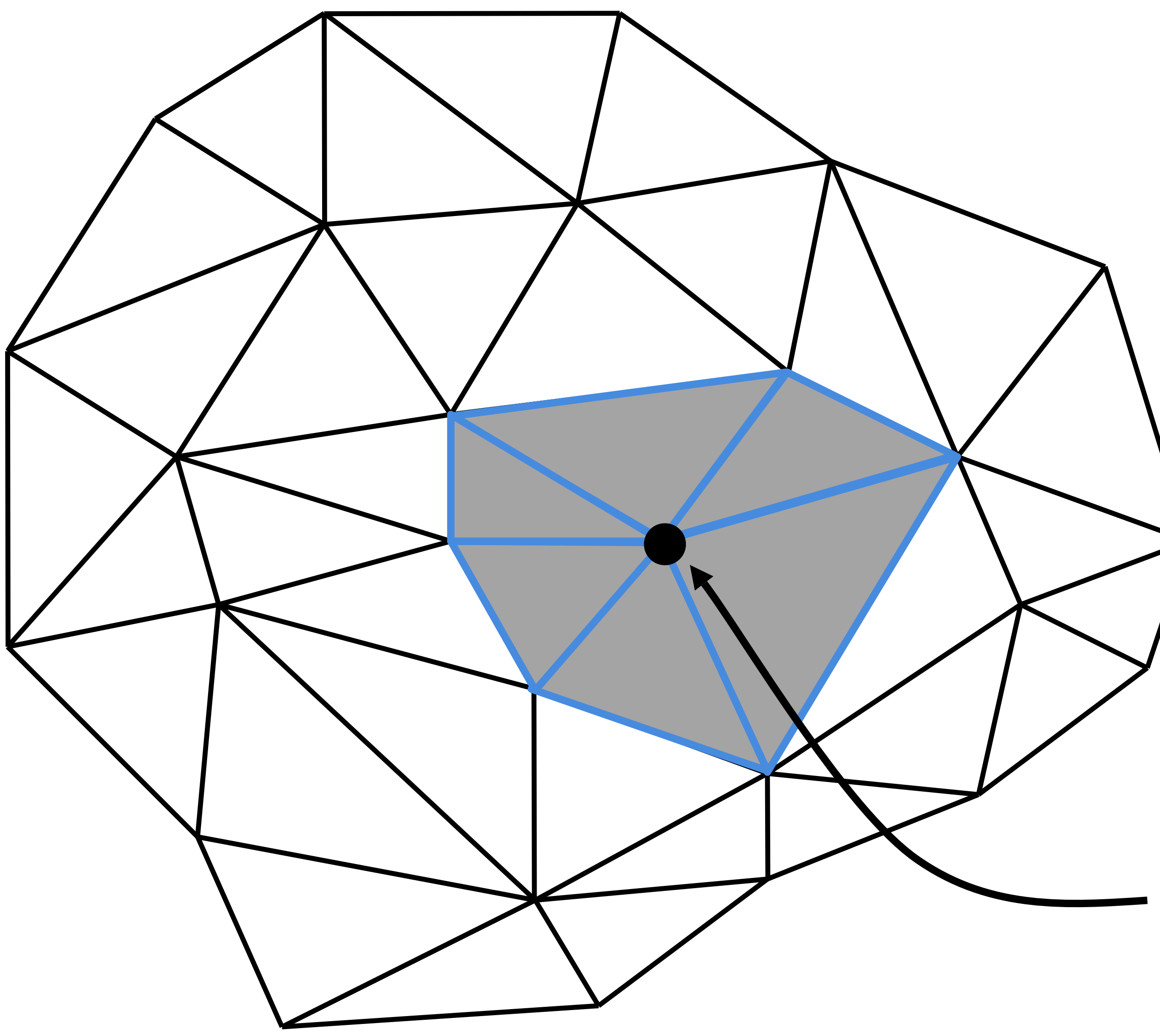
- What is a "region" ?
- What are the DOF for re-triangulation?
- Classification
  - Topology-changing vs. topology-preserving
  - Subsampling vs. filtering
  - Inverse operation → progressive meshes

Vertex Removal



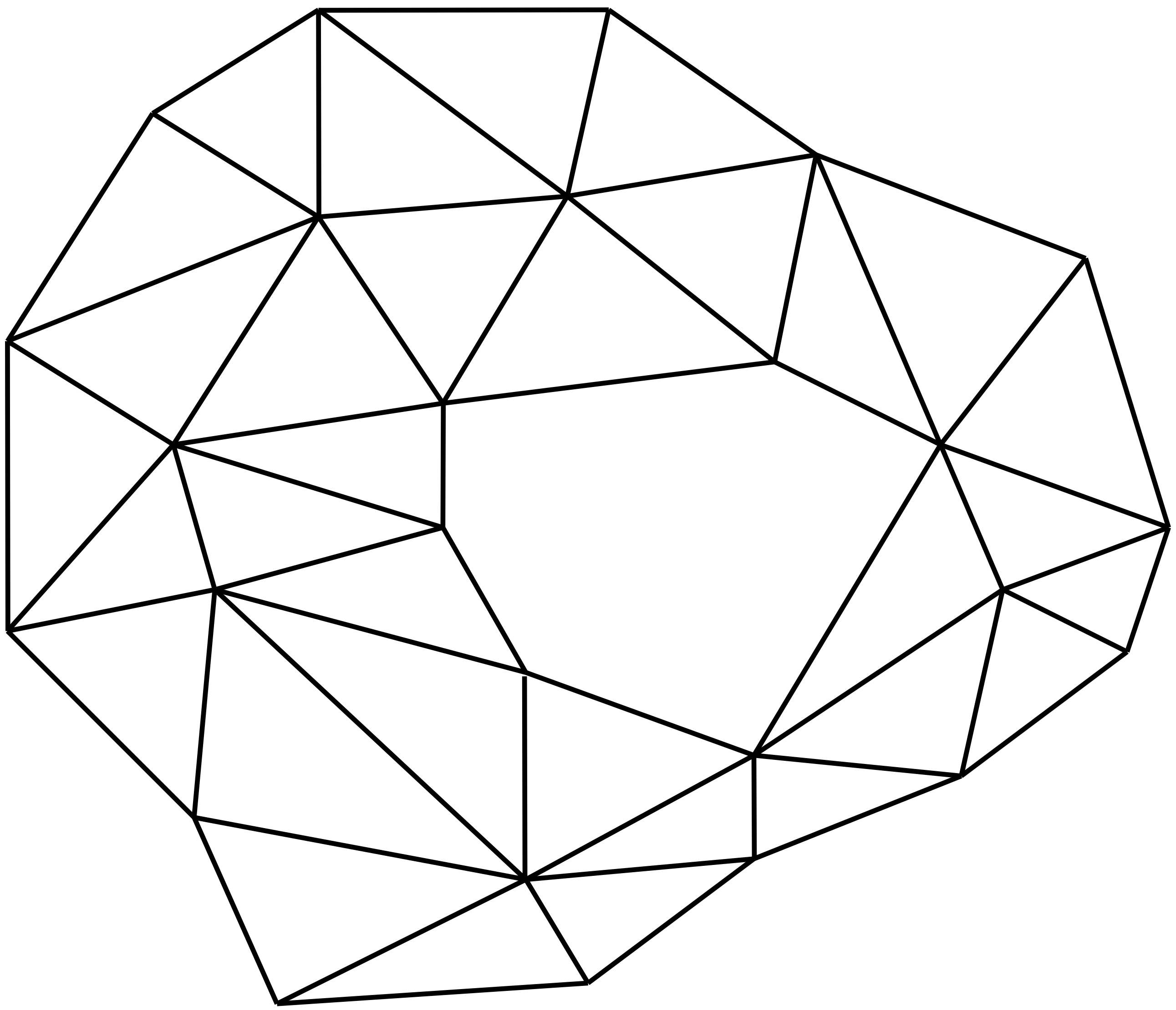
Select a vertex to be eliminated

Vertex Removal



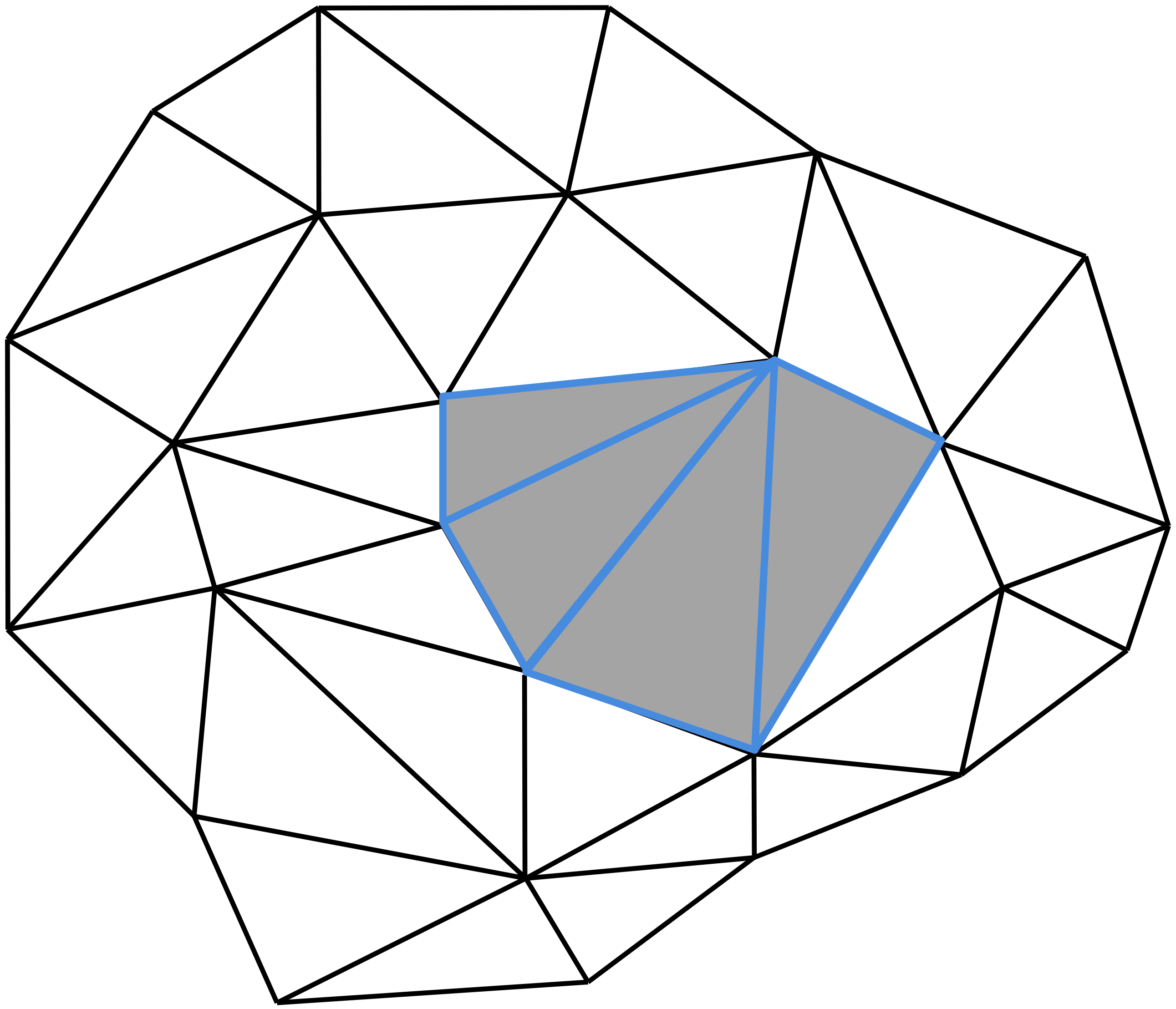
Select all triangles sharing this vertex

Vertex Removal



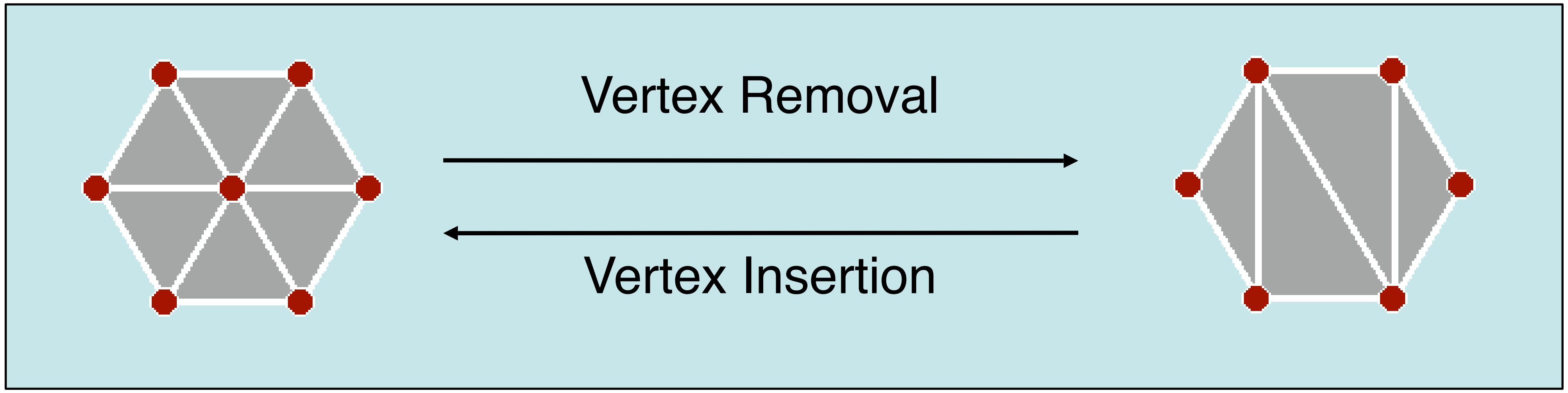
Remove the
selected triangles,
creating the hole

Vertex Removal



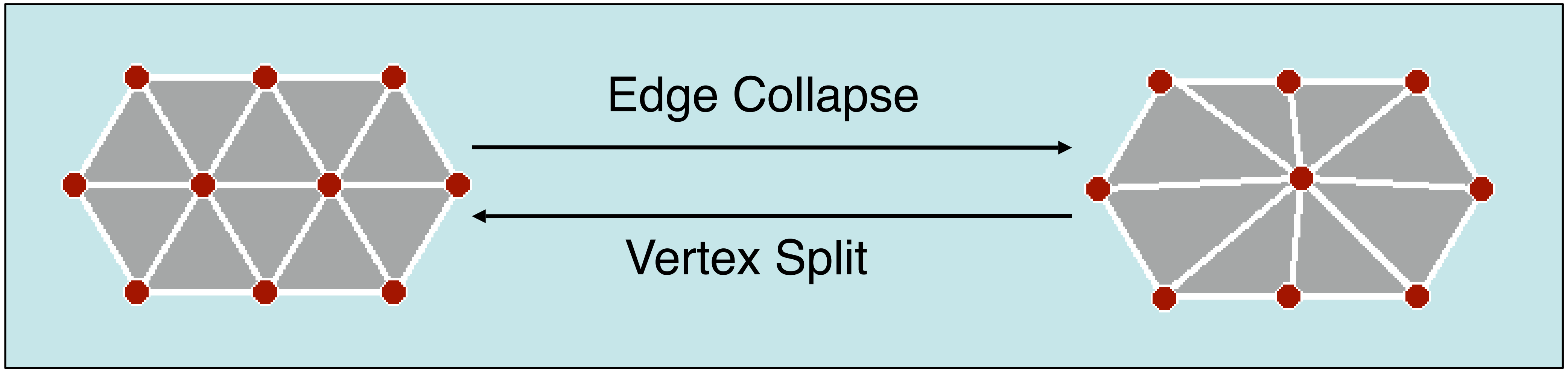
Fill the hole
with triangles

Decimation Operators



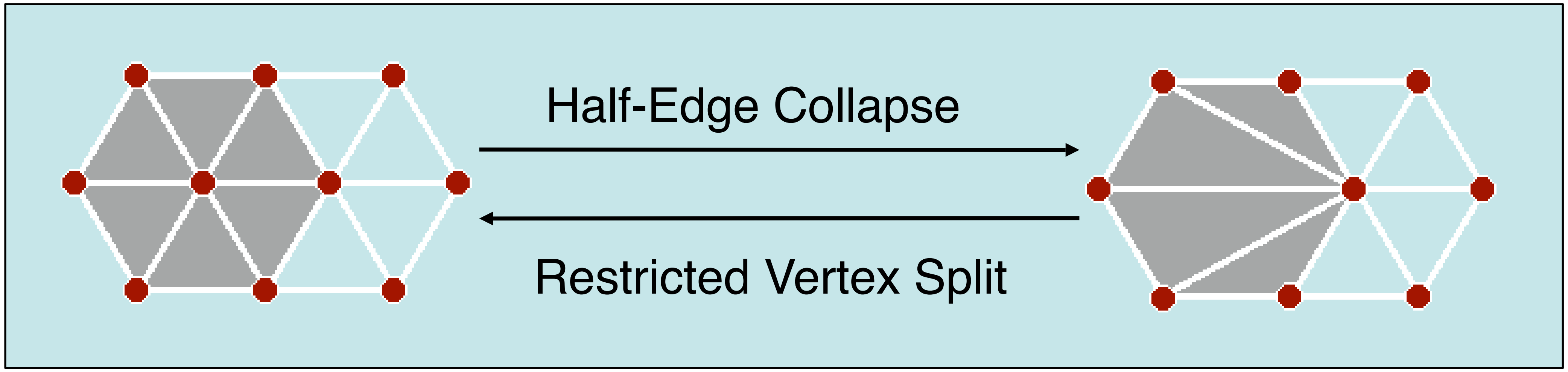
- Remove vertex
- Re-triangulate hole
  - Combinatorial DOFs
  - Sub-sampling

Decimation Operators



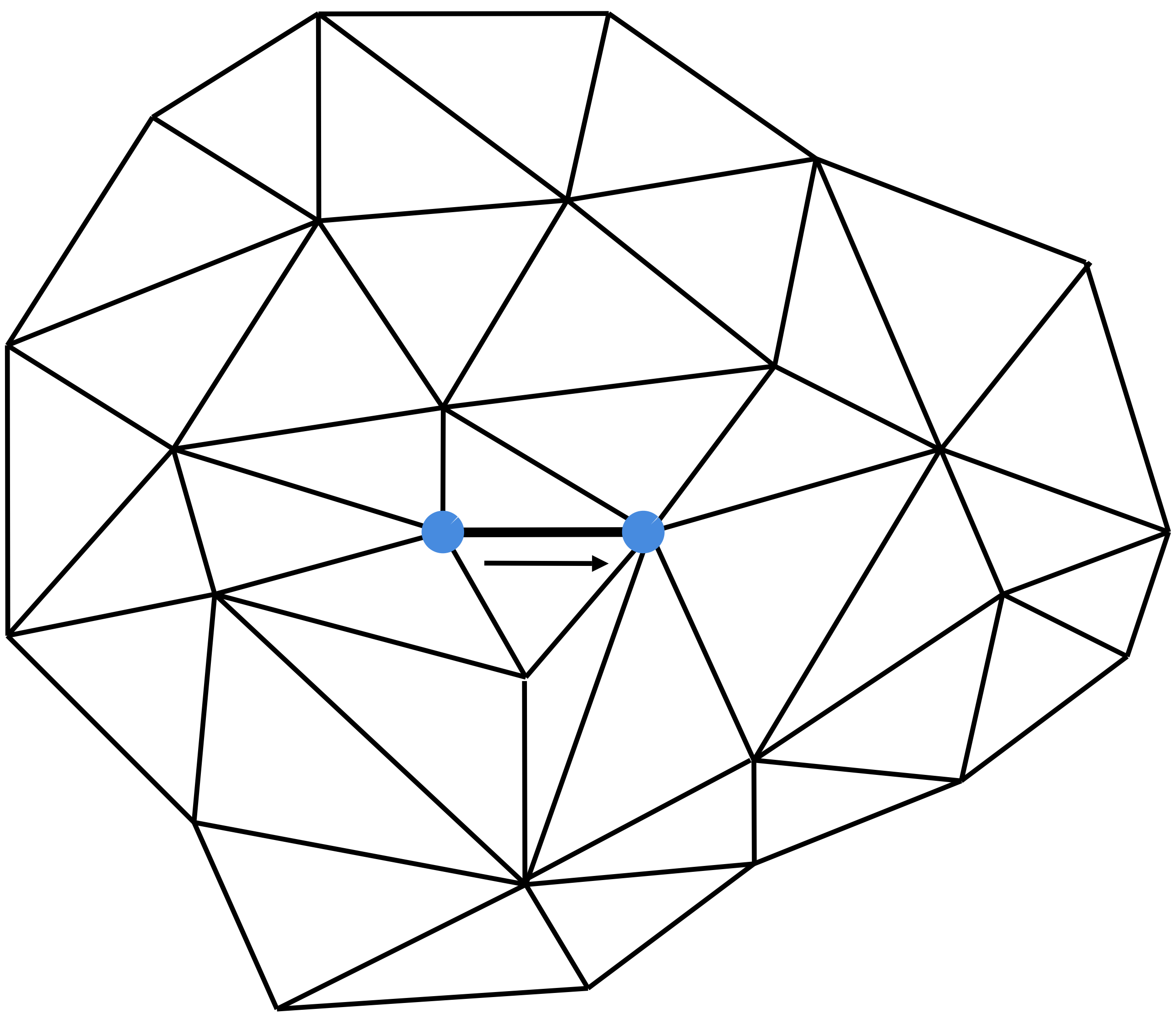
- Merge two adjacent triangles
- Define new vertex position
  - Continuous DOF
  - Filtering

Decimation Operators

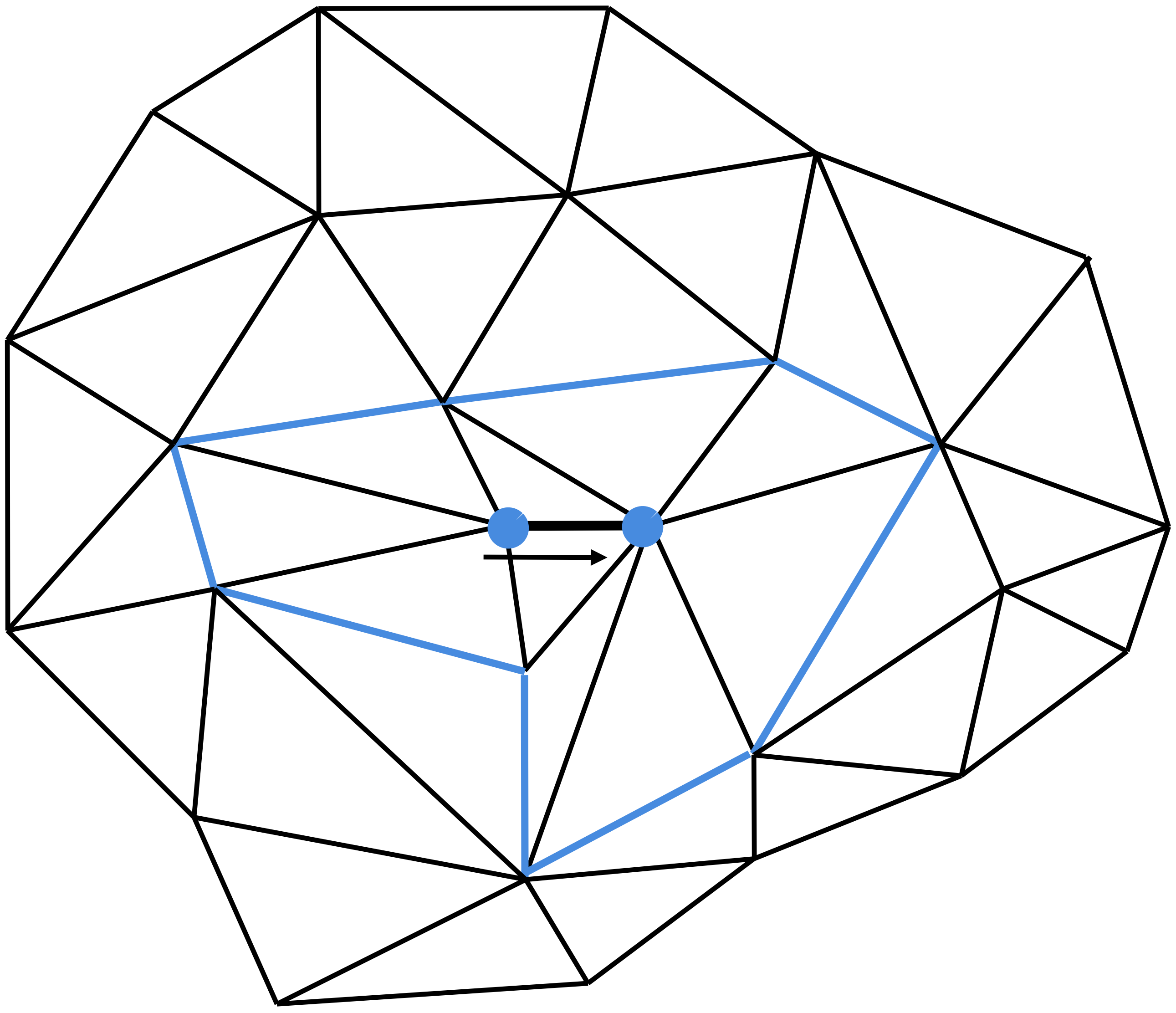


- Collapse edge into one end point
  - Special vertex removal
  - Special edge collapse
- No DOFs
  - One operator per half-edge
  - Sub-sampling!

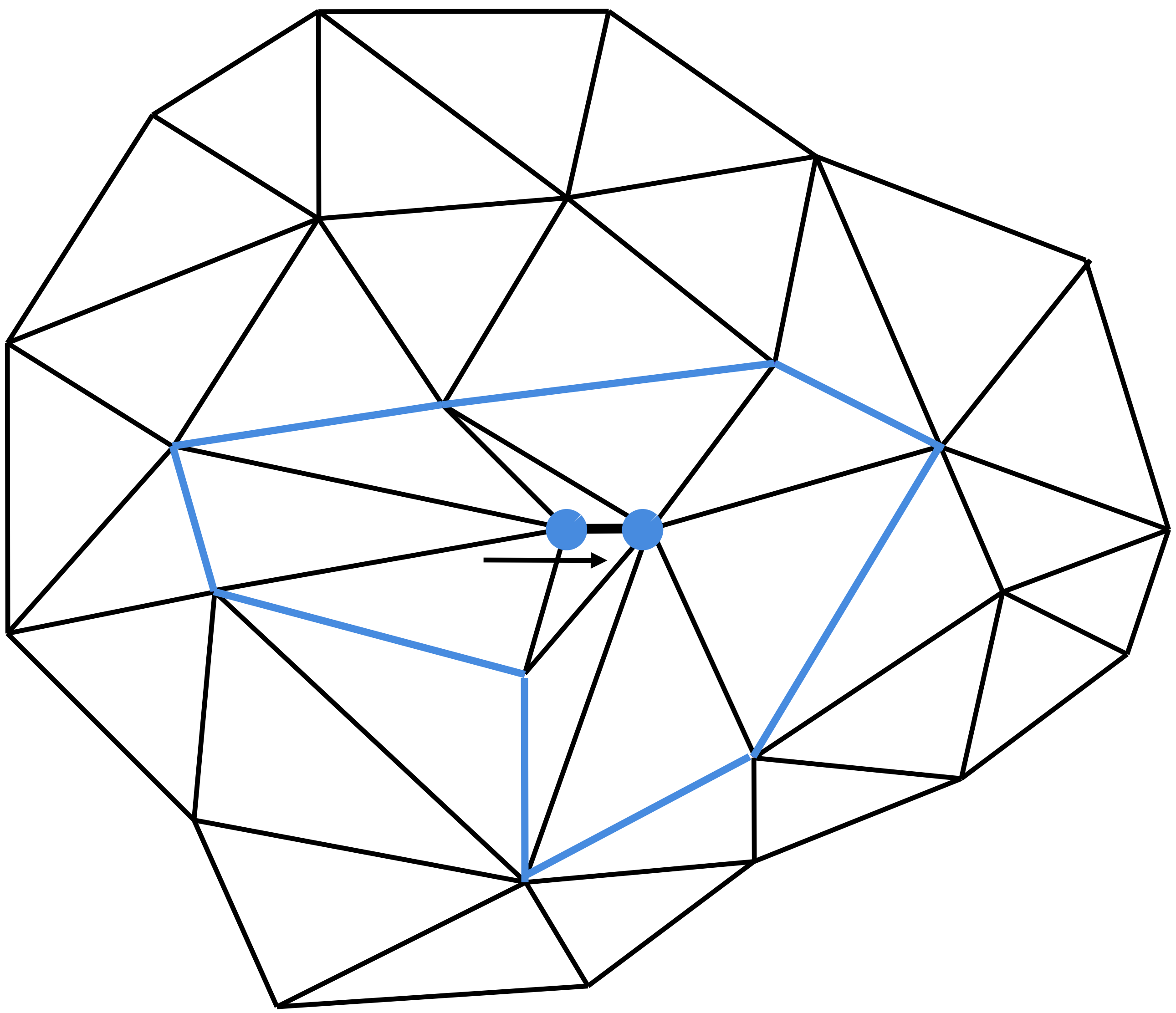
Edge Collapse



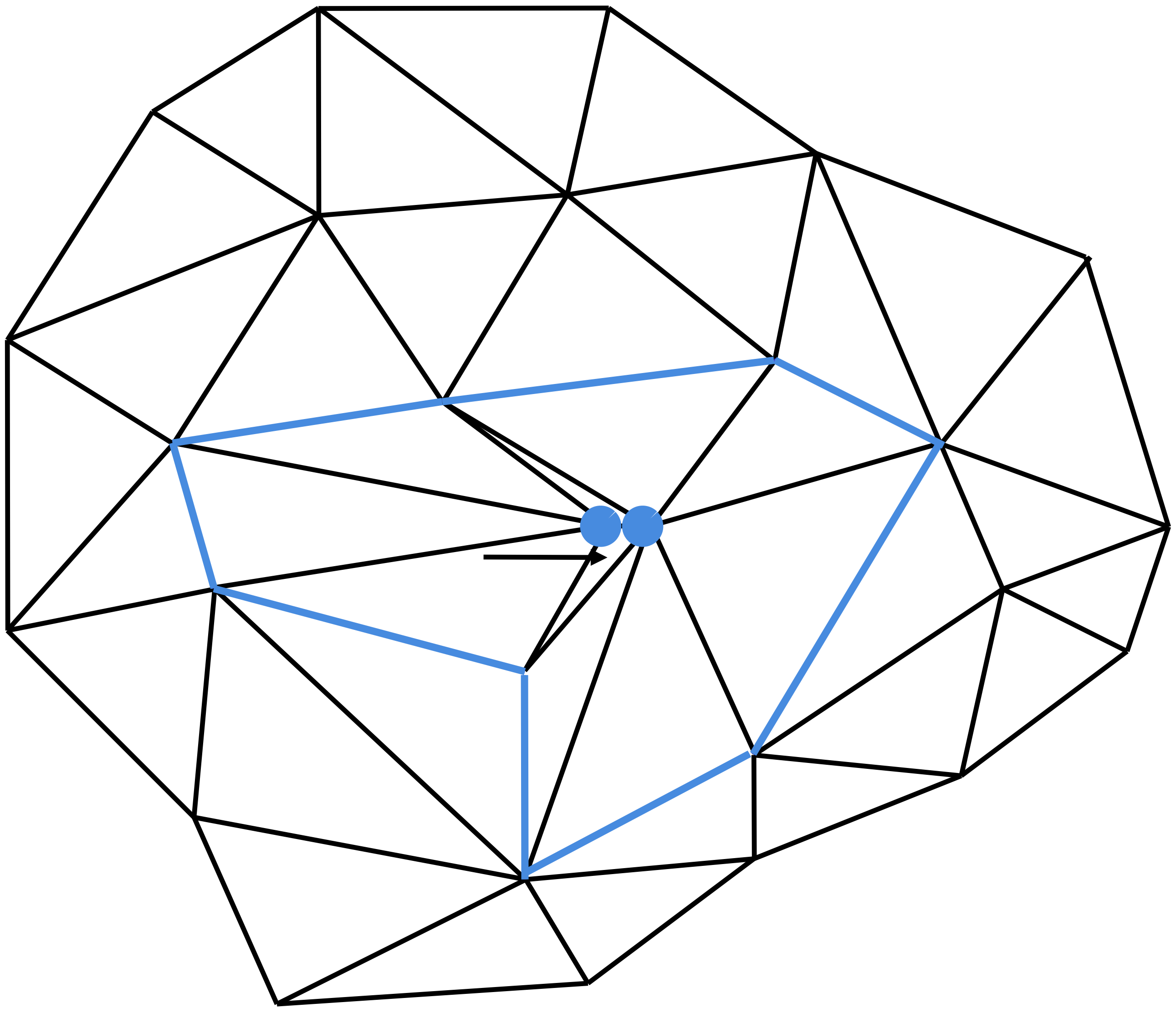
Edge Collapse



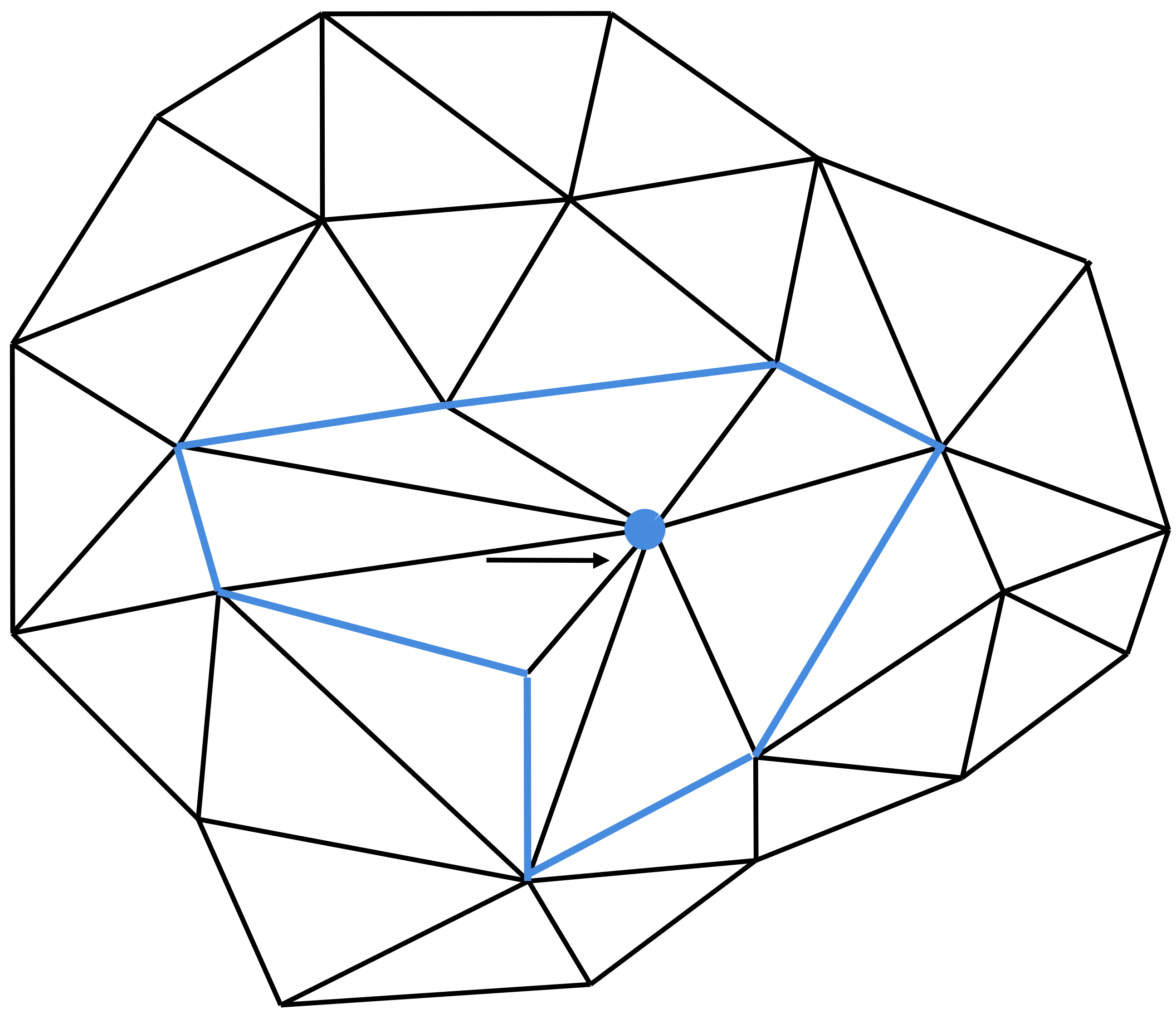
Edge Collapse



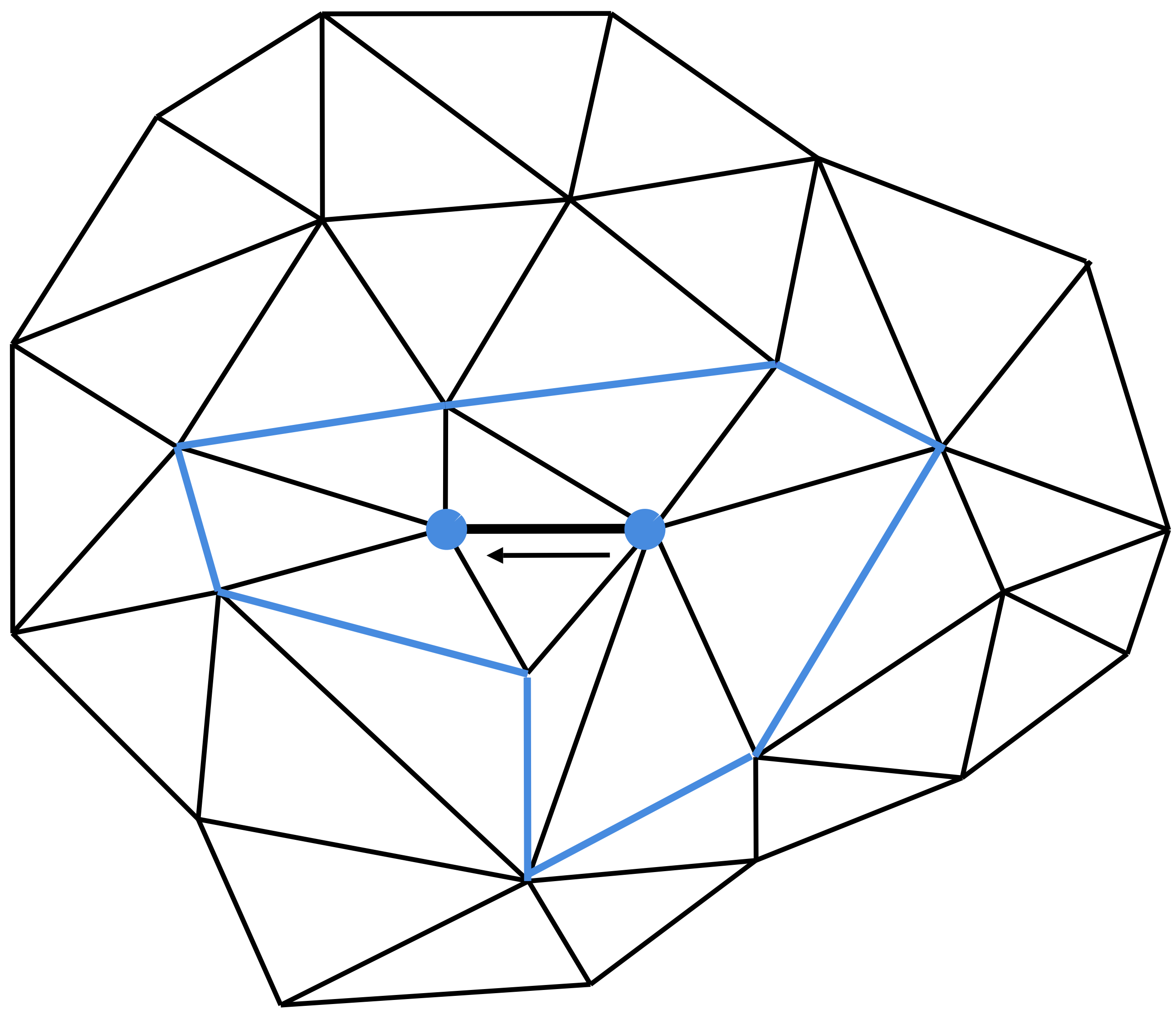
Edge Collapse



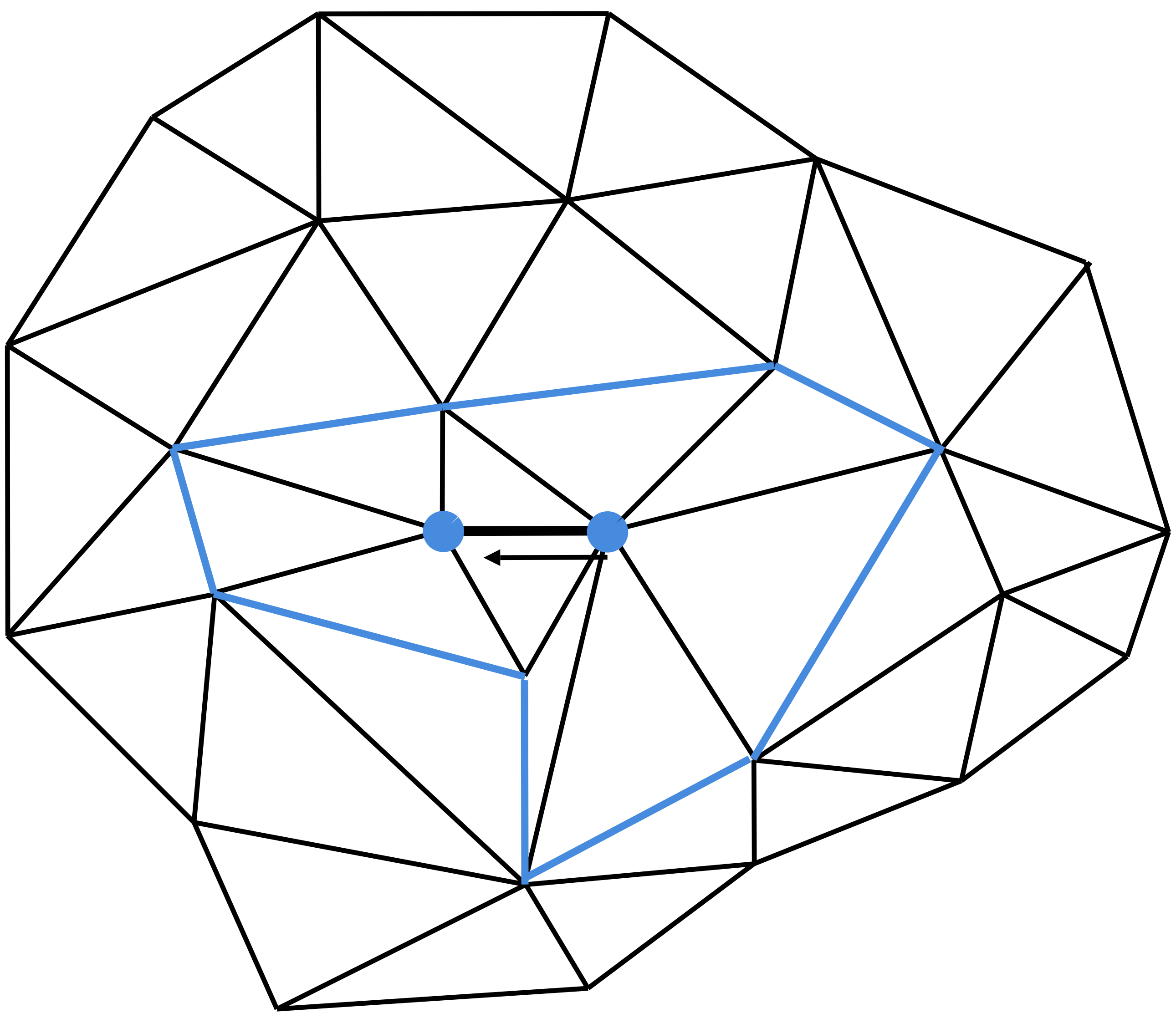
Edge Collapse



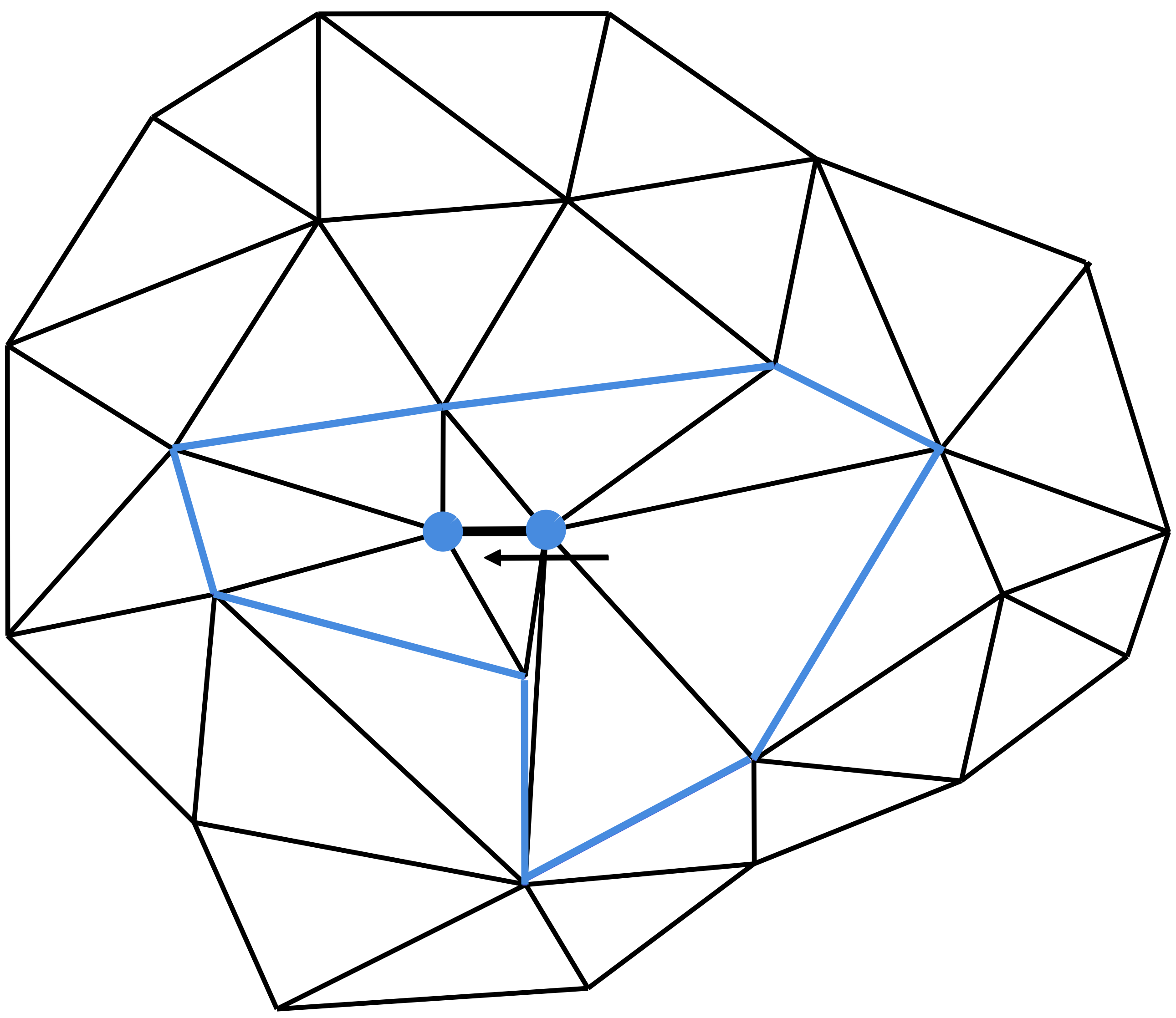
Edge Collapse



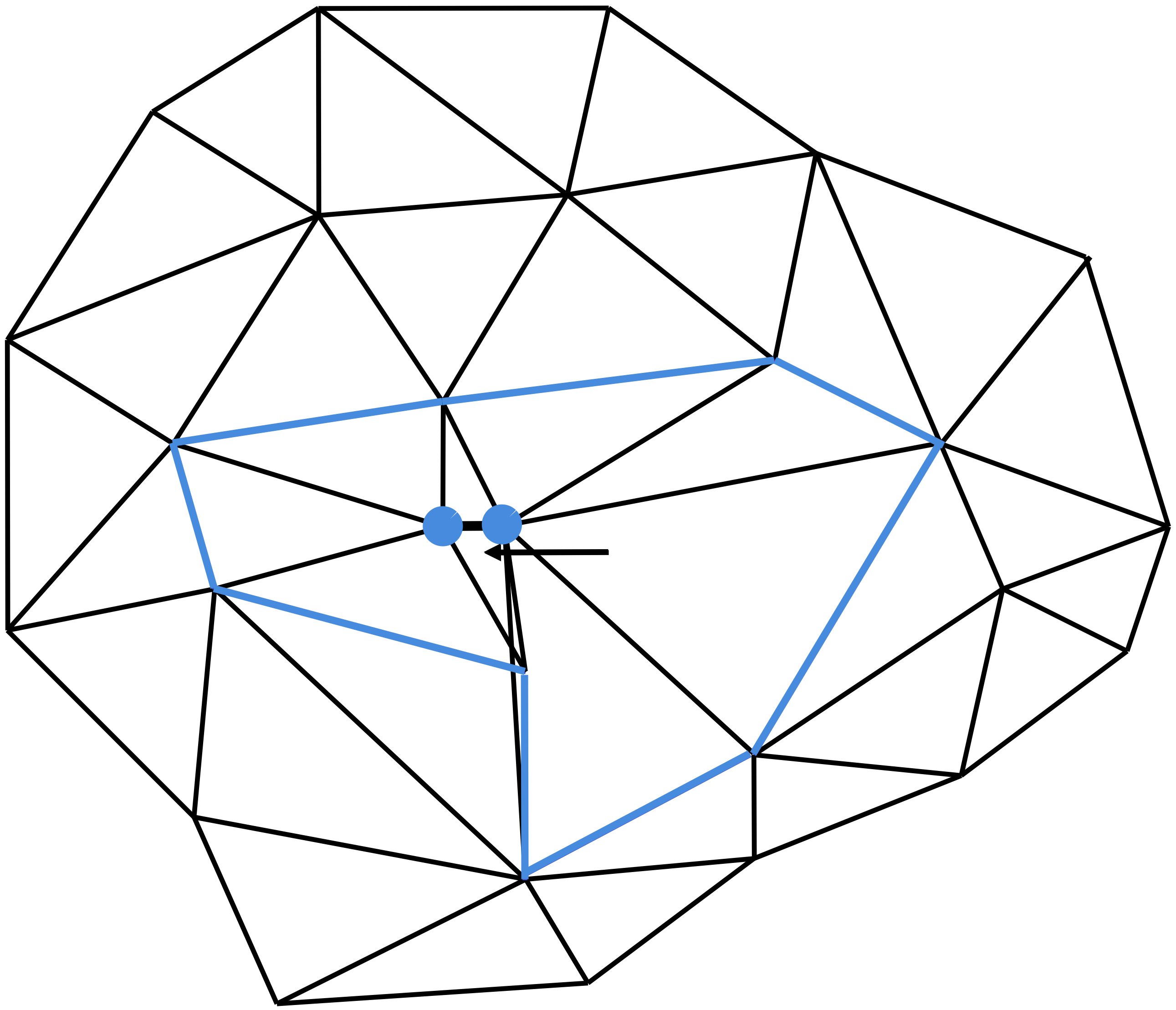
Edge Collapse



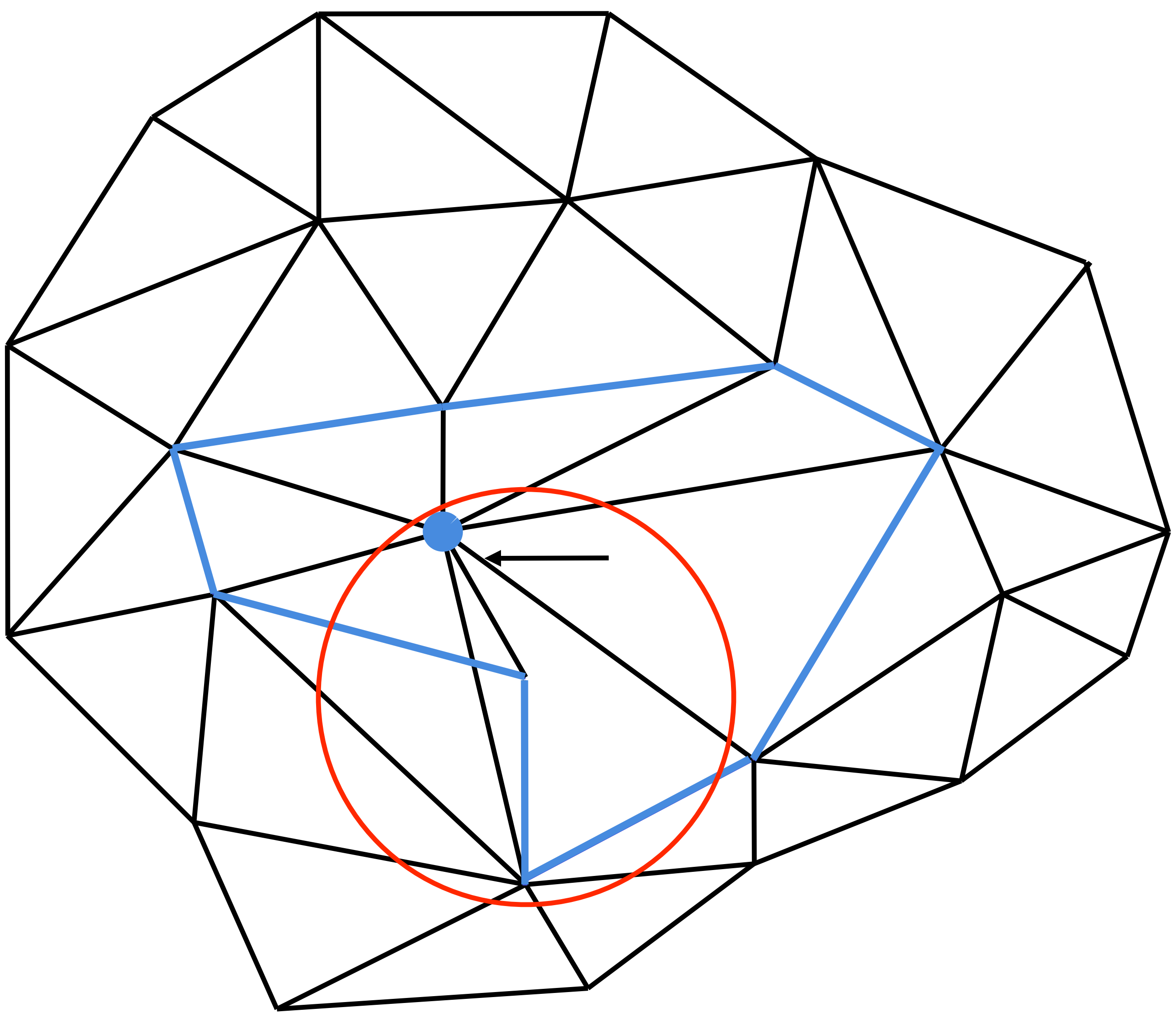
Edge Collapse



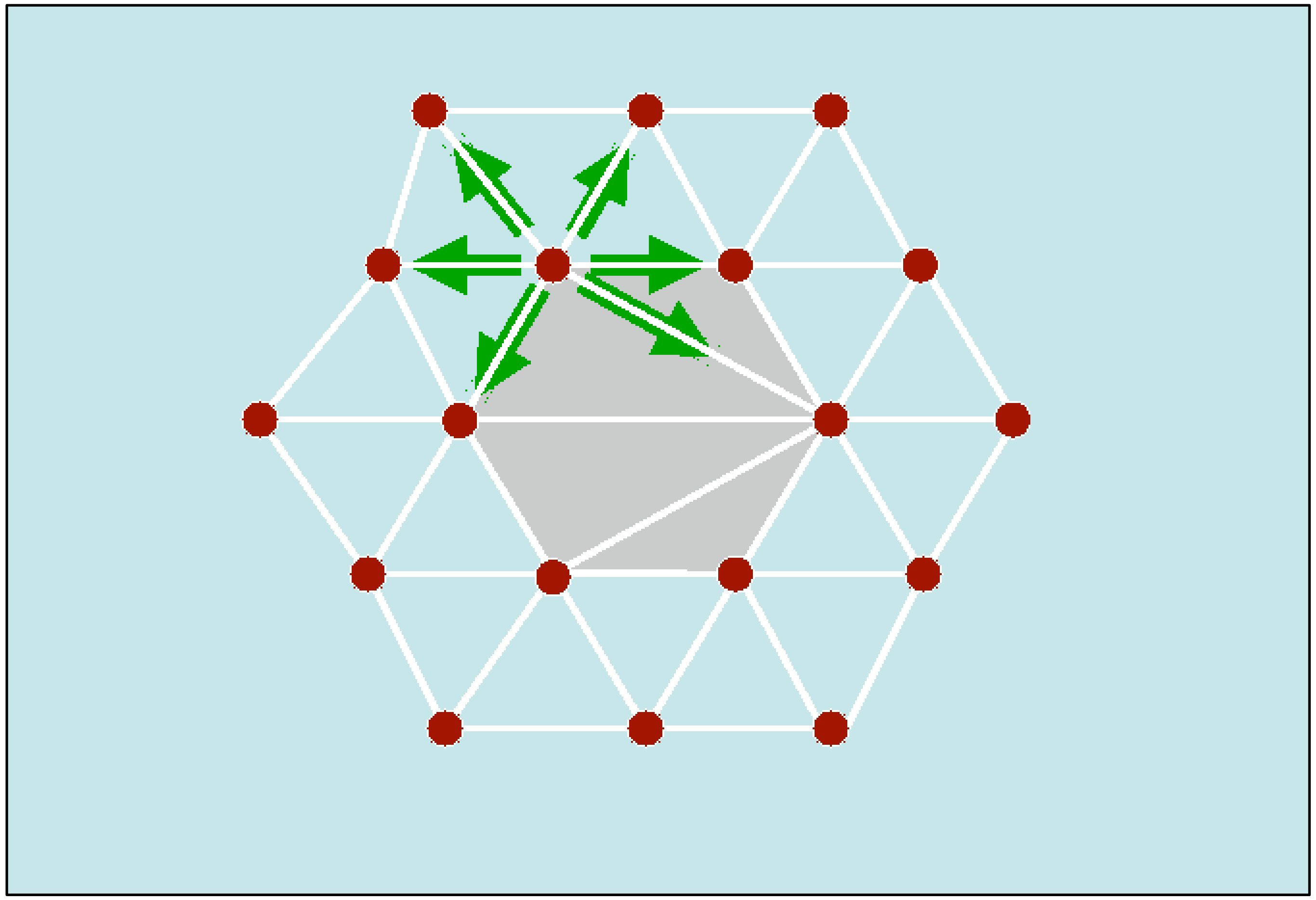
Edge Collapse



Edge Collapse



Priority Queue Updating

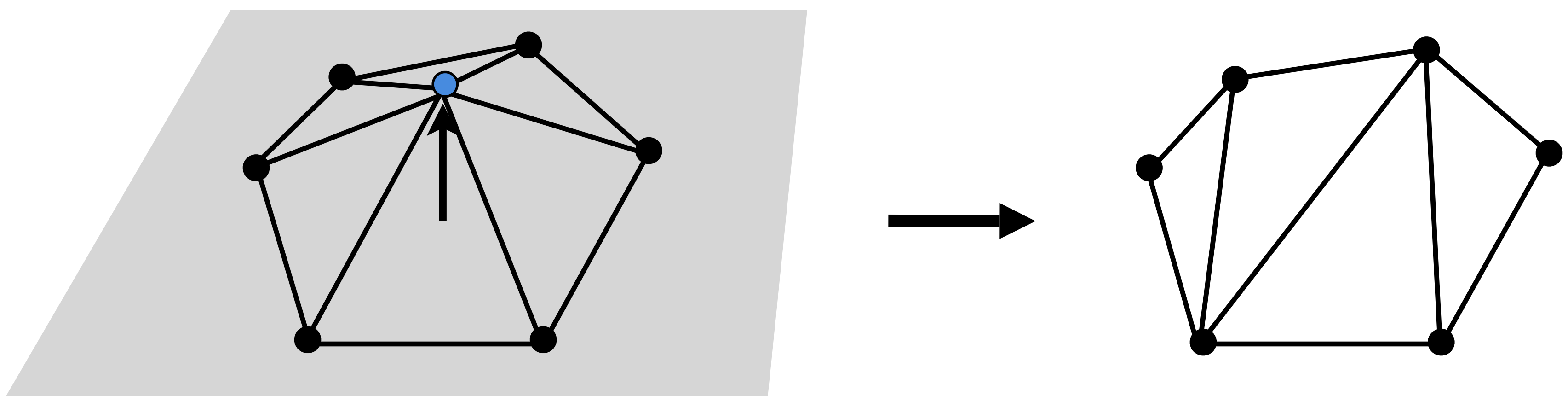


Incremental Decimation

- General Setup
- Decimation operators
- Error metrics
- Fairness criteria
- Topology changes

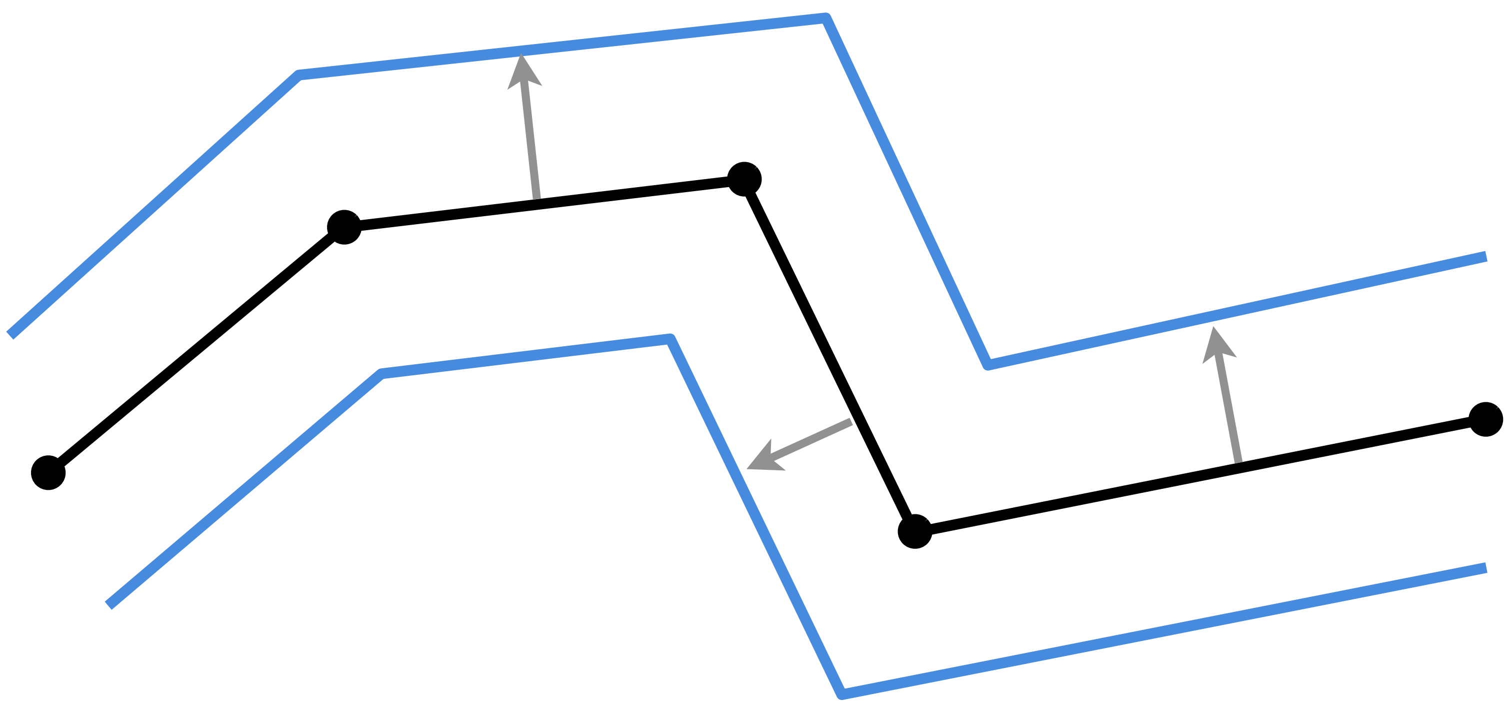
Local Error Metrics

- Local distance to mesh [Schroeder et al. 92]
  - Compute average plane
  - No comparison to *original* geometry



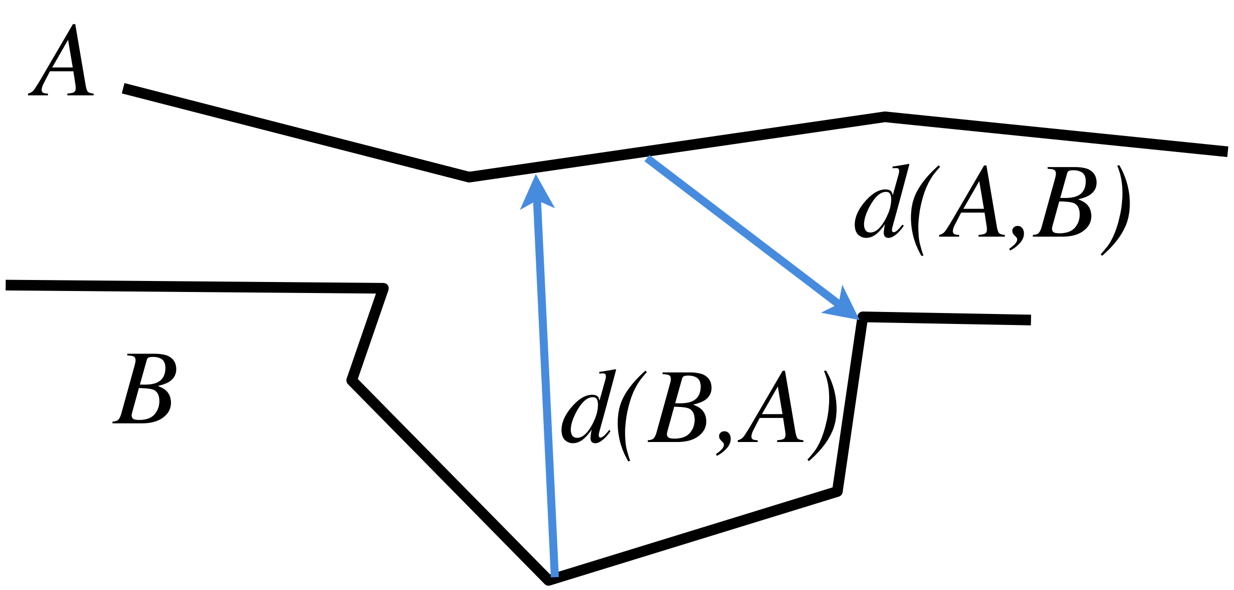
Global Error Metrics

- Simplification envelopes [Cohen et al. 96]
  - Compute (non-intersecting) offset surfaces
  - Simplification guarantees to stay within bounds



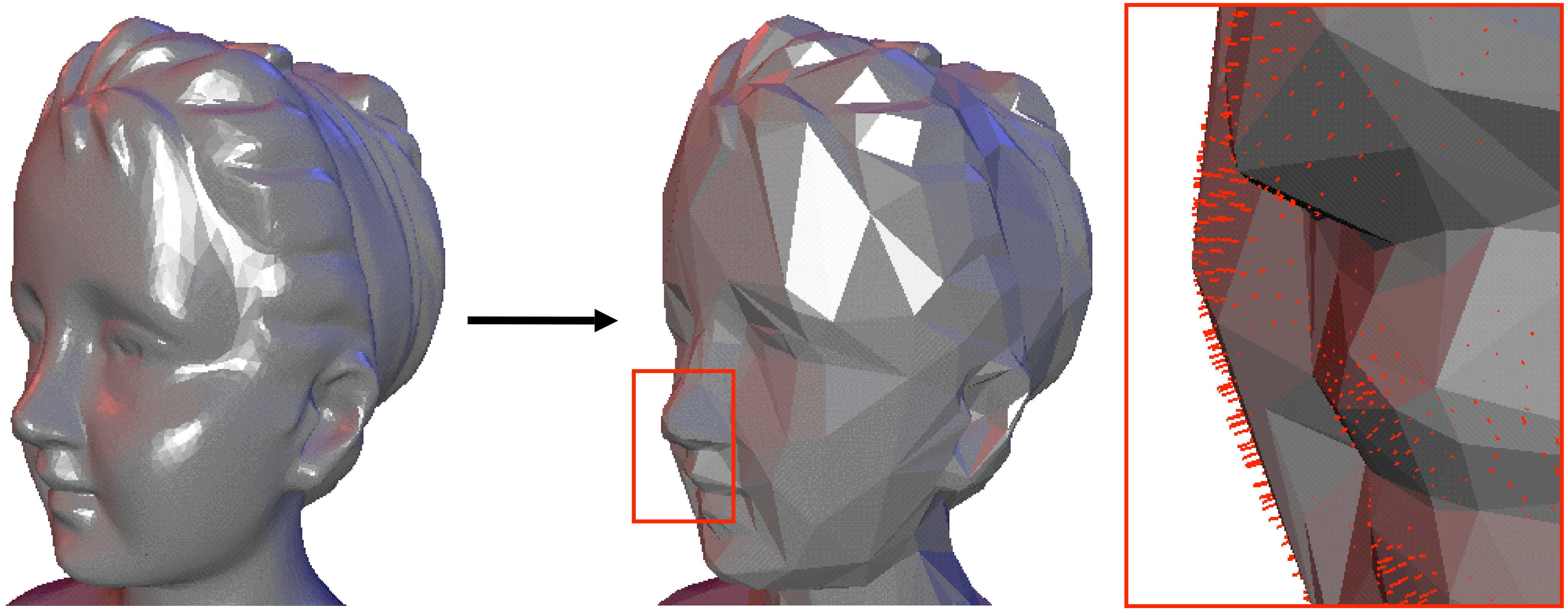
Global Error Metrics

- (Two-sided) Hausdorff distance: Maximum distance between two shapes
  - In general $d(A,B) \neq d(B,A)$
  - Computationally involved



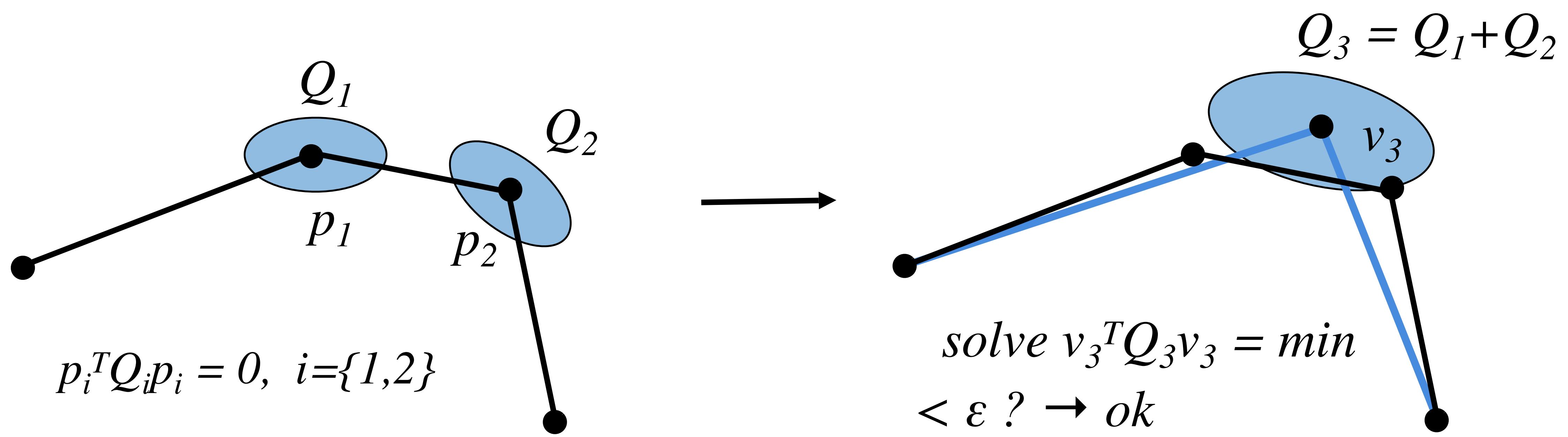
Global Error Metrics

- Scan data: One-sided Hausdorff distance sufficient
  - From original vertices to current surface



Global Error Metrics

- Error quadrics [Garland, Heckbert 97]
  - Squared distance to planes at vertex
  - No bound on true error



Complexity

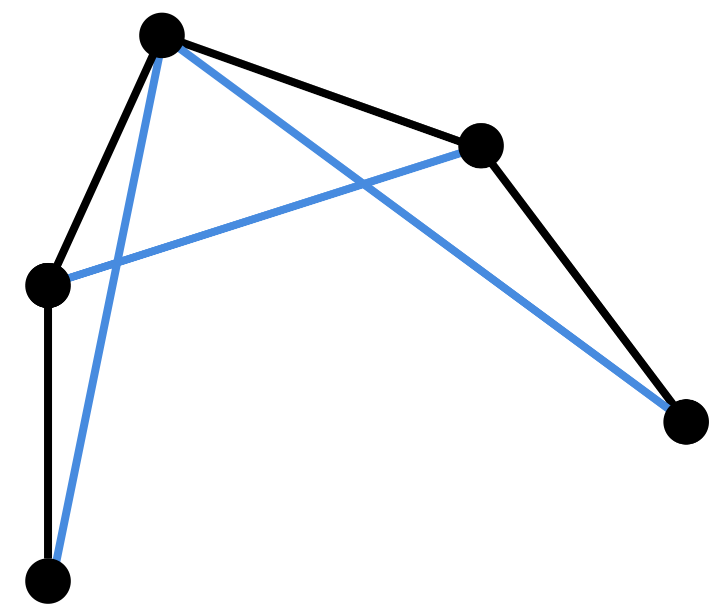
- N = number of vertices
- Priority queue for half-edges
  - 6 N * log (6 N)
- Error control
  - Local $O(1) \Rightarrow$ global $O(N)$
  - Local $O(N) \Rightarrow$ global $O(N^2)$

Incremental Decimation

- General Setup
- Decimation operators
- Error metrics
- Fairness criteria
- Topology changes

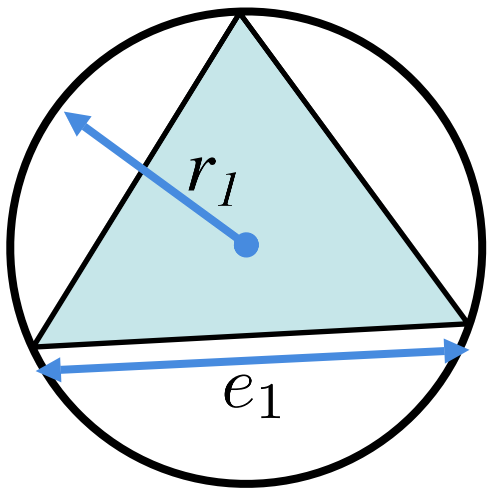
Fairness Criteria

- Rate quality of decimation operation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...

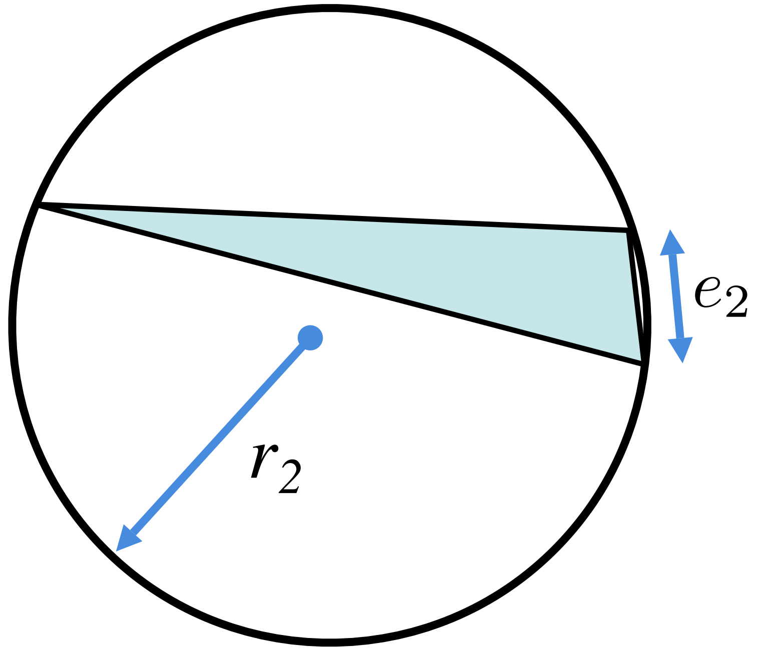


Fairness Criteria

- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...

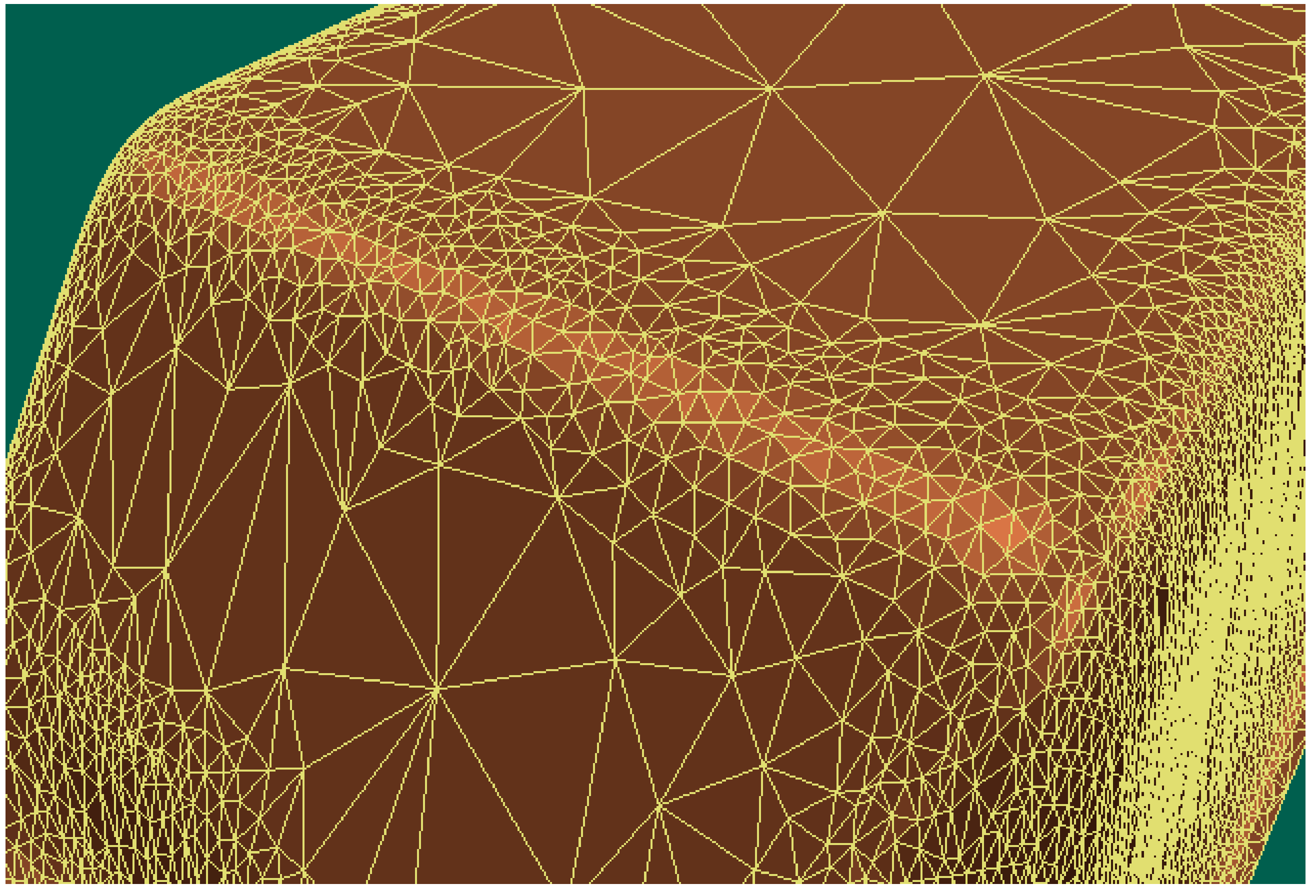


$$\frac{r_1}{e_1} < \frac{r_2}{e_2}$$



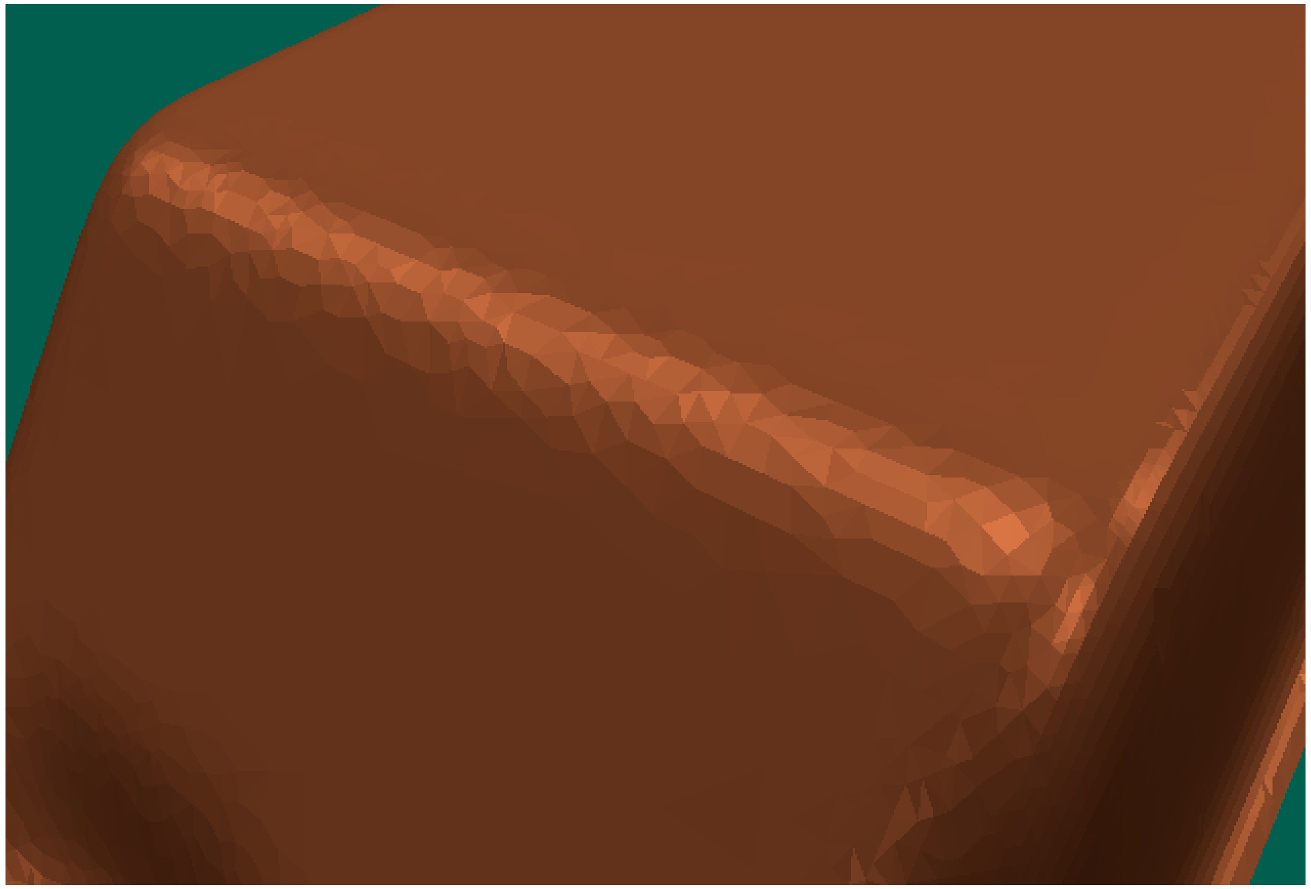
Fairness Criteria

- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...



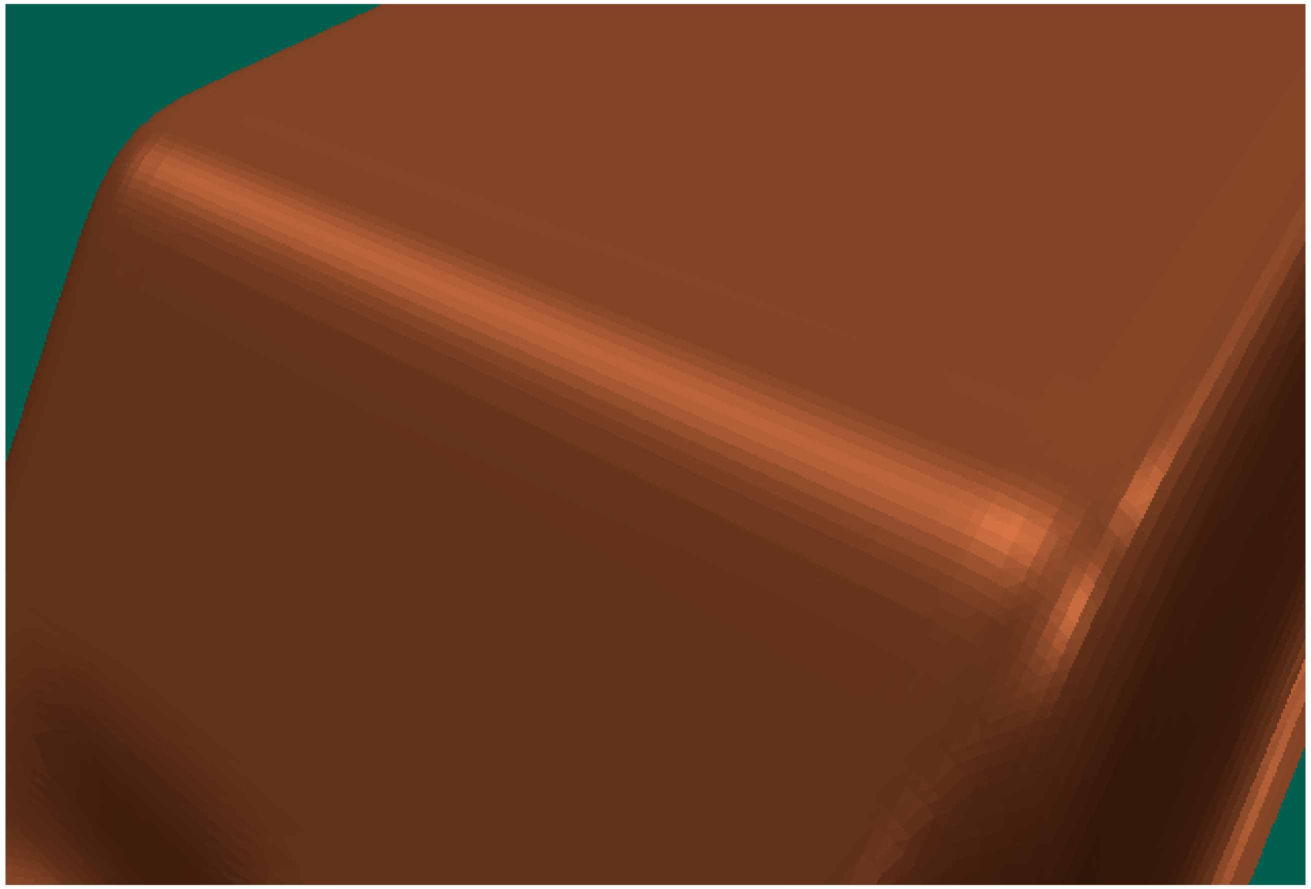
Fairness Criteria

- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...



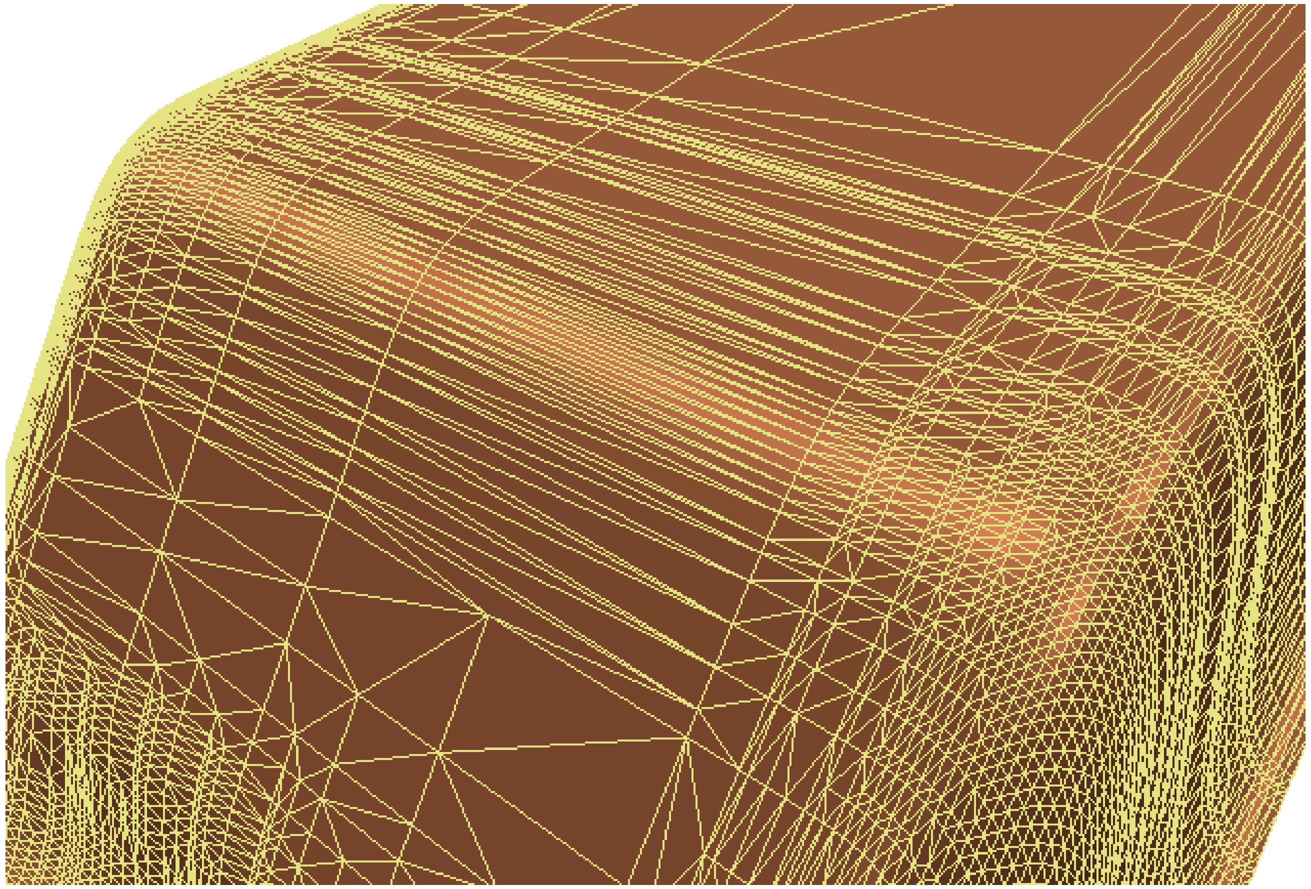
Fairness Criteria

- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...



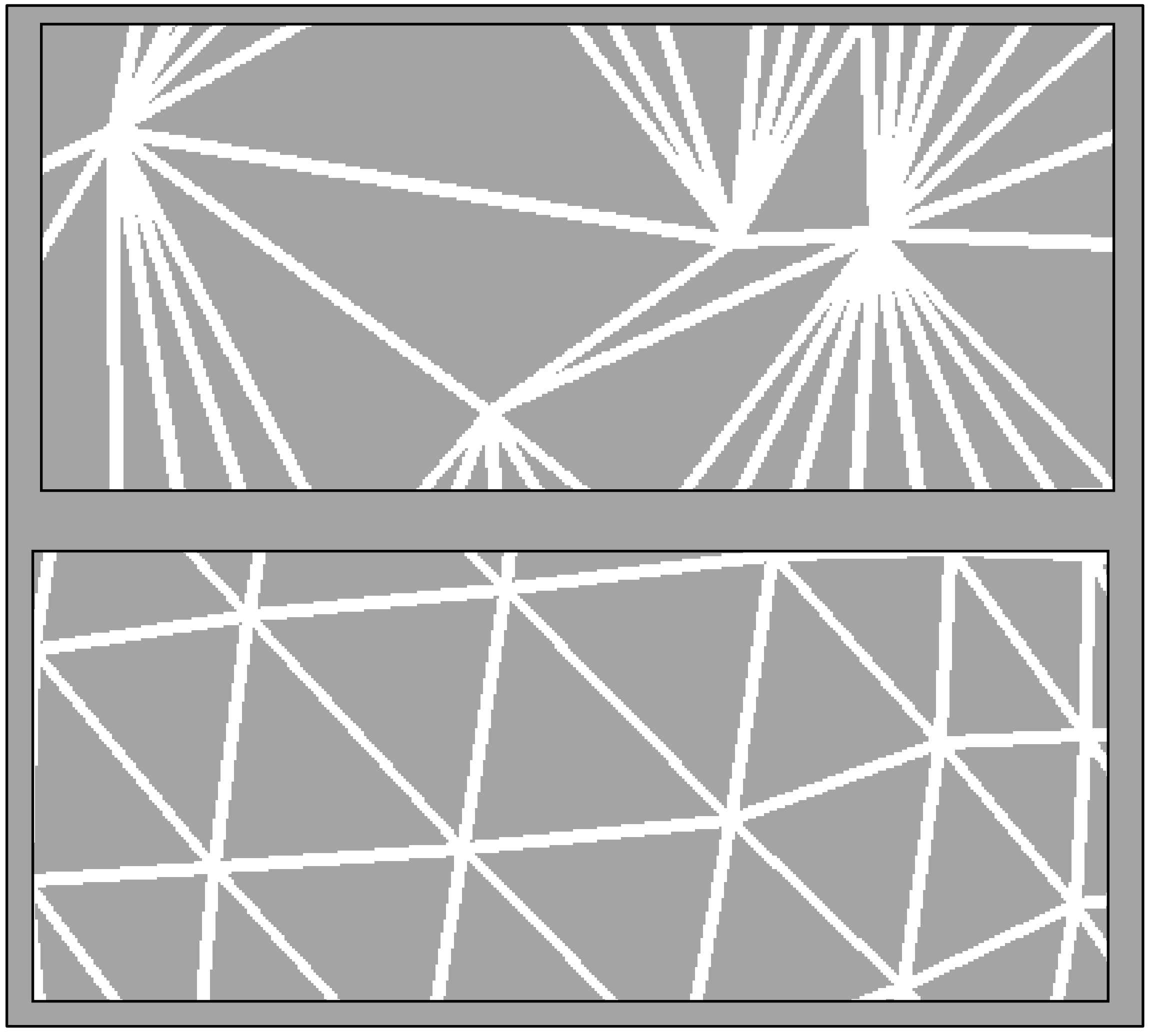
Fairness Criteria

- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...



Fairness Criteria

- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valence balance
  - Color differences
  - ...



Fairness Criteria

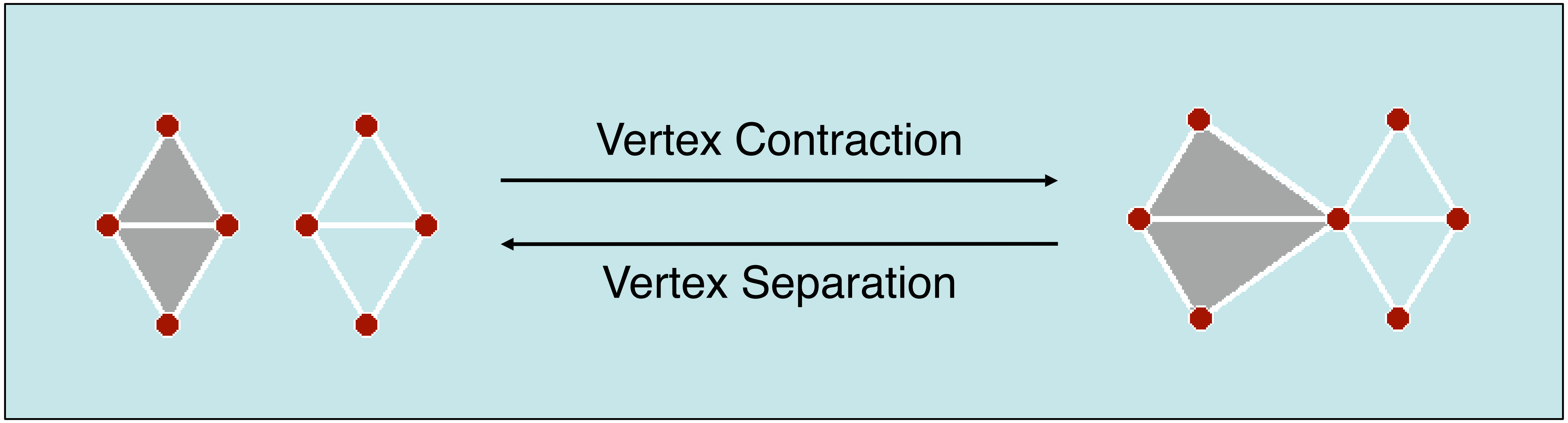
- Rate quality after decimation
  - Approximation error
  - Triangle shape
  - Dihedral angles
  - Valance balance
  - Color differences
  - ...

Incremental Decimation

- General Setup
- Decimation operators
- Error metrics
- Fairness criteria
- Topology changes

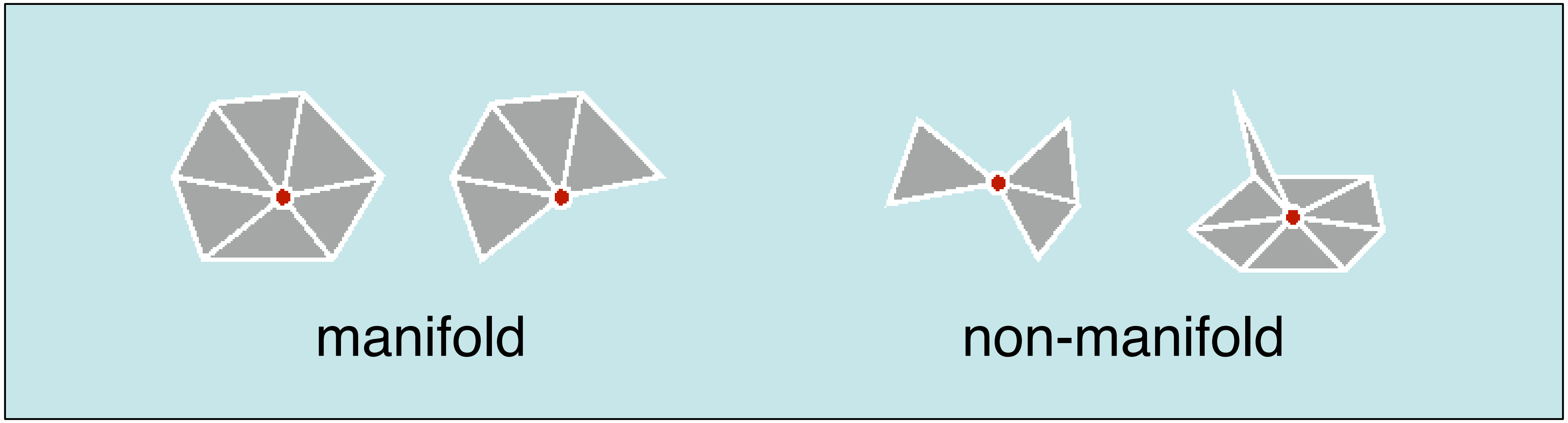
Topology Changes ?

- Merge vertices across non-edges
  - Changes mesh topology
  - Need *spatial neighborhood* information
  - Generates *non-manifold* meshes



Topology Changes ?

- Merge vertices across non-edges
  - Changes mesh topology
  - Need *spatial neighborhood* information
  - Generates *non-manifold* meshes



Comparison

- Vertex clustering
  - fast, but difficult to control simplified mesh
  - topology changes, non-manifold meshes
  - global error bound, but often not close to optimum
- Iterative decimation with quadric error metrics
  - good trade-off between mesh quality and speed
  - explicit control over mesh topology
  - restricting normal deviation improves mesh quality

Outline

- Applications
- Problem Statement
- Mesh Decimation Methods
  - Vertex Clustering
  - Iterative Decimation
  - **Extensions**

Out-of-core Decimation

- Handle very large data sets that do not fit into main memory
- Key: Avoid random access to mesh data structure during simplification
- Examples
  - Garland, Shaffer: *A Multiphase Approach to Efficient Surface Simplification*, IEEE Visualization 2002
  - Wu, Kobbelt: *A Stream Algorithm for the Decimation of Massive Meshes*, Graphics Interface 2003

Multiphase Simplification

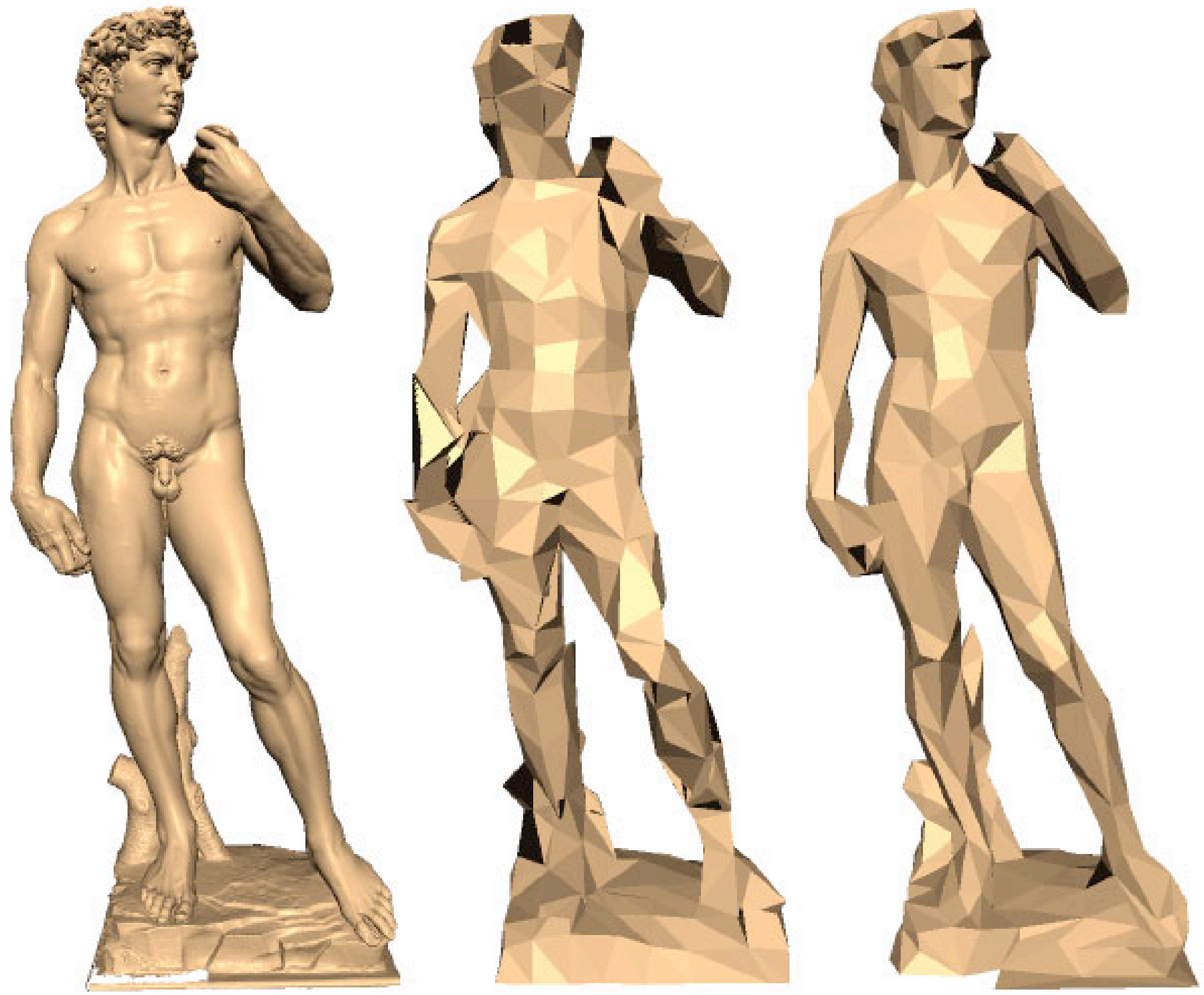
1. Phase: Out-of-core clustering
    - compute accumulated error quadrics and vertex representative for each cell of uniform voxel grid
2. Phase: In-core iterative simplification
    - compute fundamental quadrics
    - iteratively contract edge of smallest cost

Multiphase Simplification

1. Phase: Out-of-core clustering
   - compute accumulated error quadrics and vertex representative for each cell of uniform voxel grid
2. Phase: In-core iterative simplification
   - ~~compute fundamental quadrics~~
   - use accumulated quadrics from clustering phase
   - iteratively contract edge of smallest cost

→ achieves a coupling between the two phases

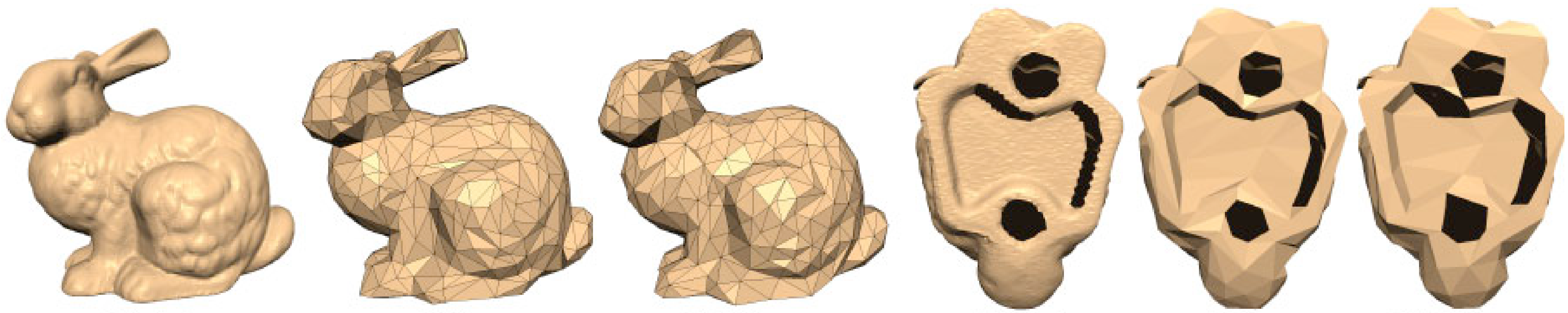
Multiphase Simplification



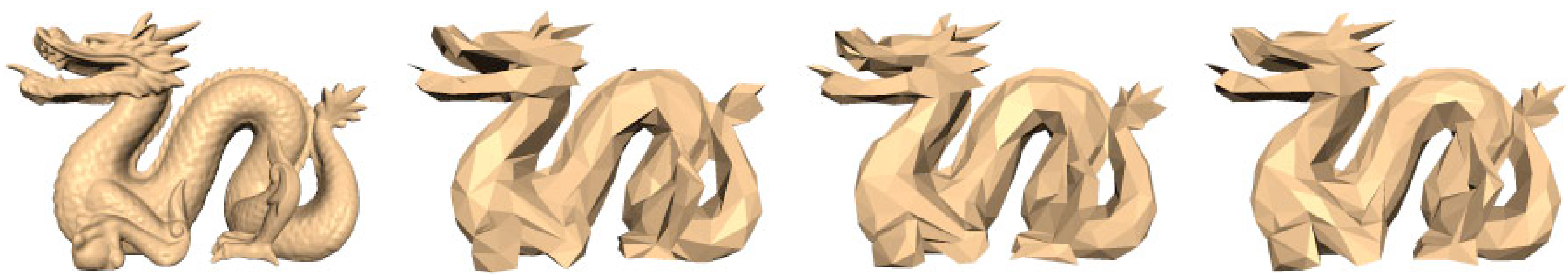
Original (8 million triangles) | Uniform clustering (1157 triangles) | Multiphase (1000 triangles)

Garland, Shaffer: *A Multiphase Approach to Efficient Surface Simplification*, IEEE Visualization 2002

Multiphase Simplification



Original QSlim Multiphase Original QSlim Multiphase

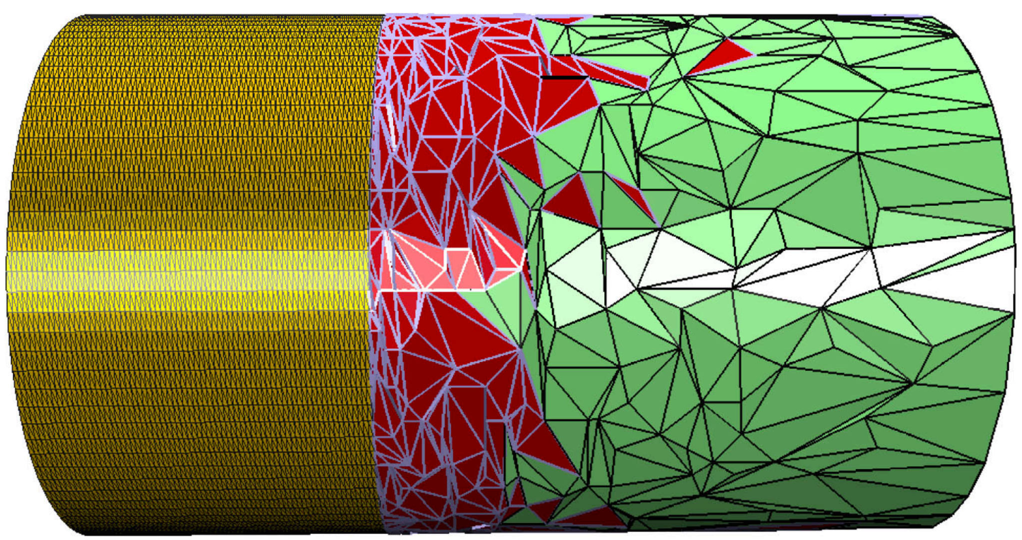


(a) Original model (b) Separate passes (c) Coupled passes (d) QSlim

Garland, Shaffer: *A Multiphase Approach to Efficient Surface Simplification*, IEEE Visualization 2002

Out-of-core Decimation

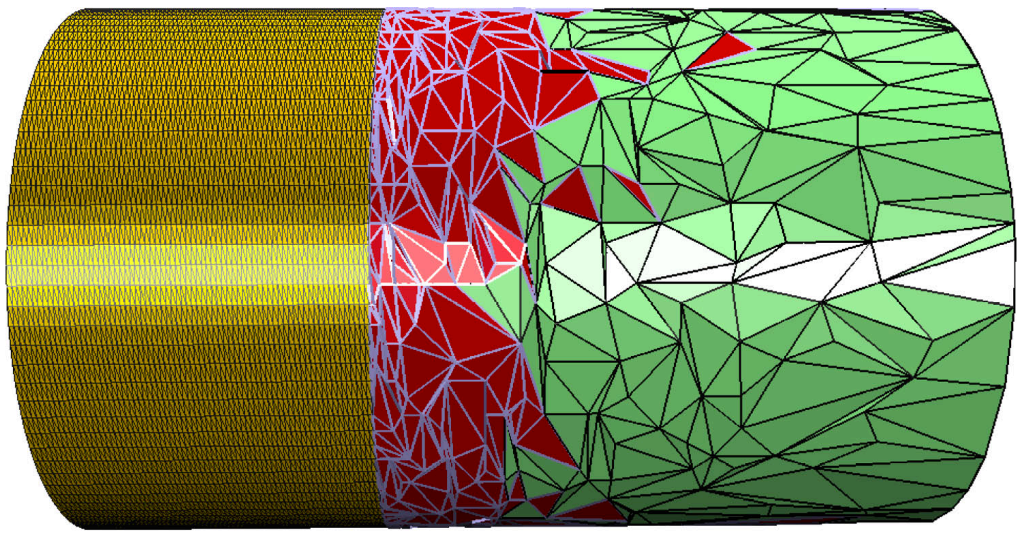
- Streaming approach based on edge collapse operations using QEM
- Pre-sorted input stream allows fixed-sized active working set independent of input and output model complexity



Wu, Kobbelt: *A Stream Algorithm for the Decimation of Massive Meshes*, Graphics Interface 2003

Out-of-core Decimation

- Randomized multiple choice optimization avoids global heap data structure
- Special treatment for boundaries required



Wu, Kobbelt: *A Stream Algorithm for the Decimation of Massive Meshes*, Graphics Interface 2003