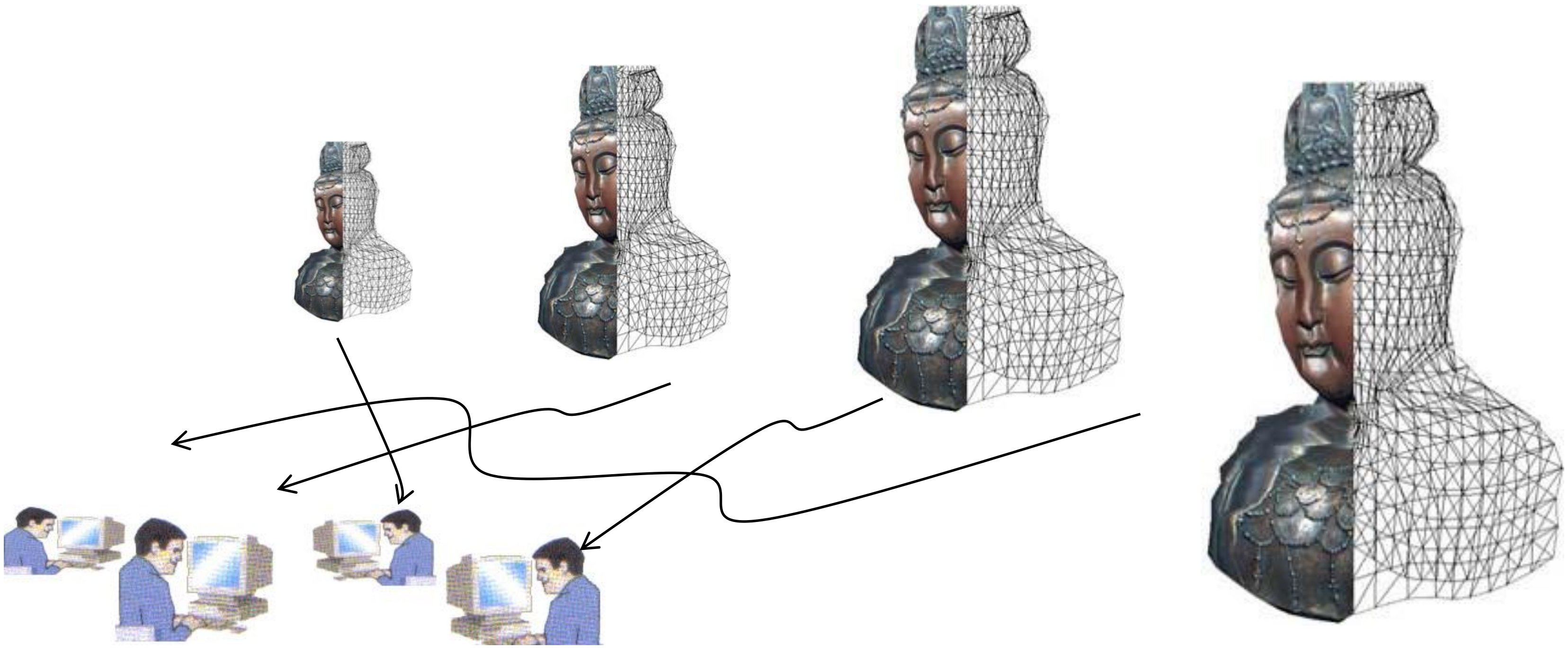
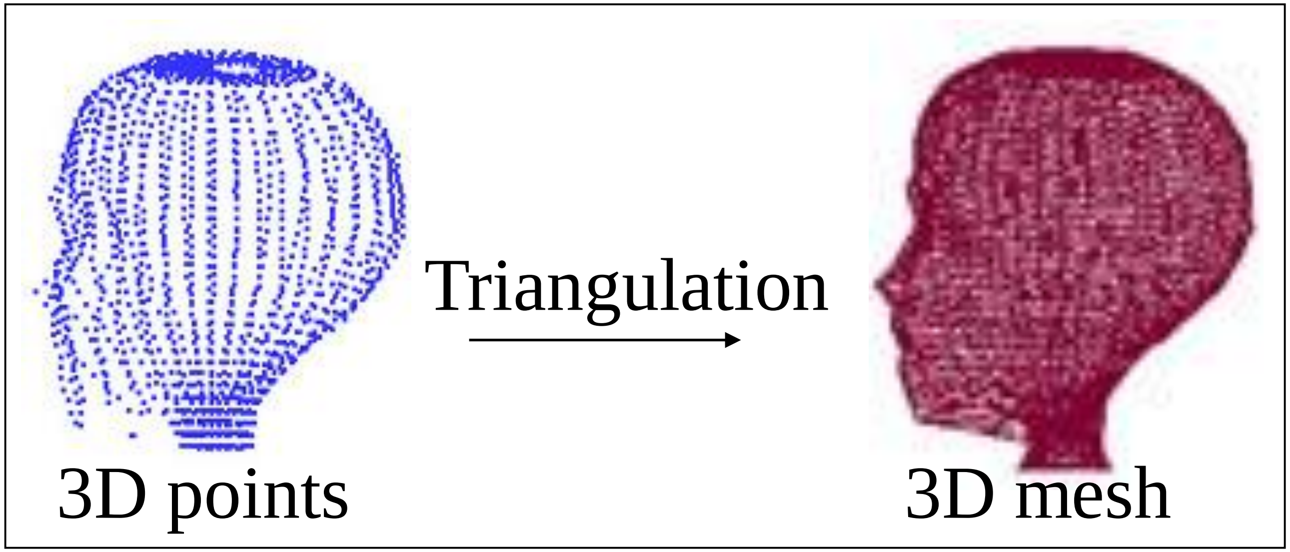
Overview of 3D Mesh Simplification

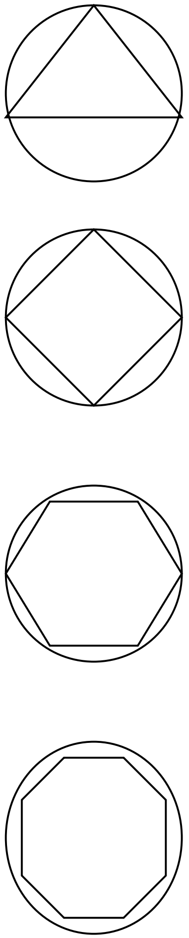
Notes prepared by: *Dr. Anup Basu & Dr. Irene Cheng*



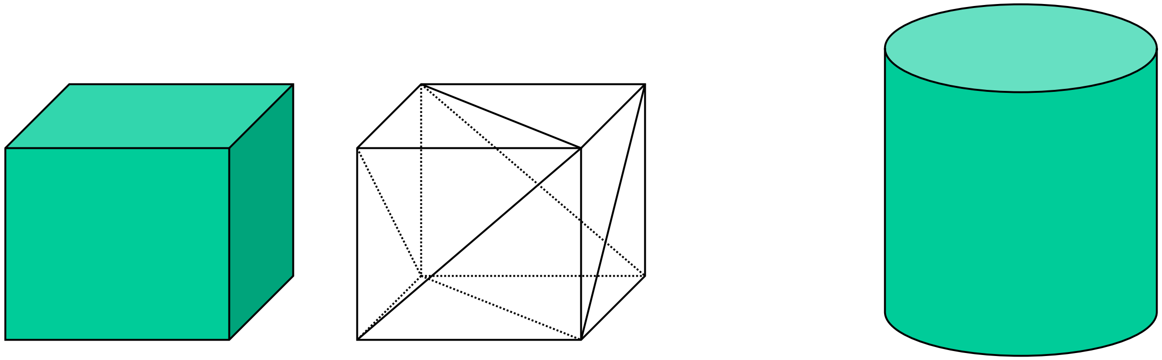
Department of Computing Science
University of Alberta

External Shape & 3D Geometry

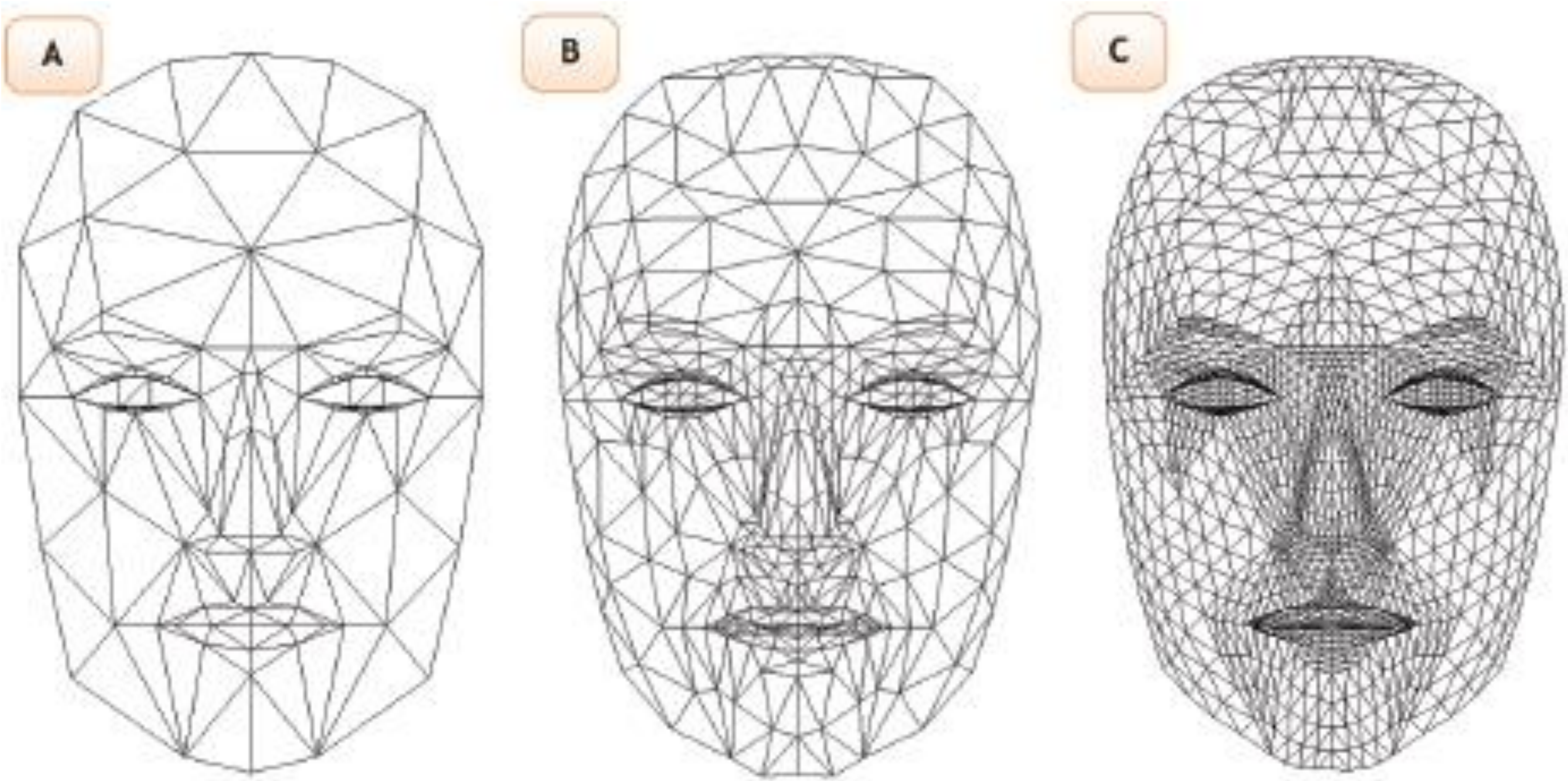




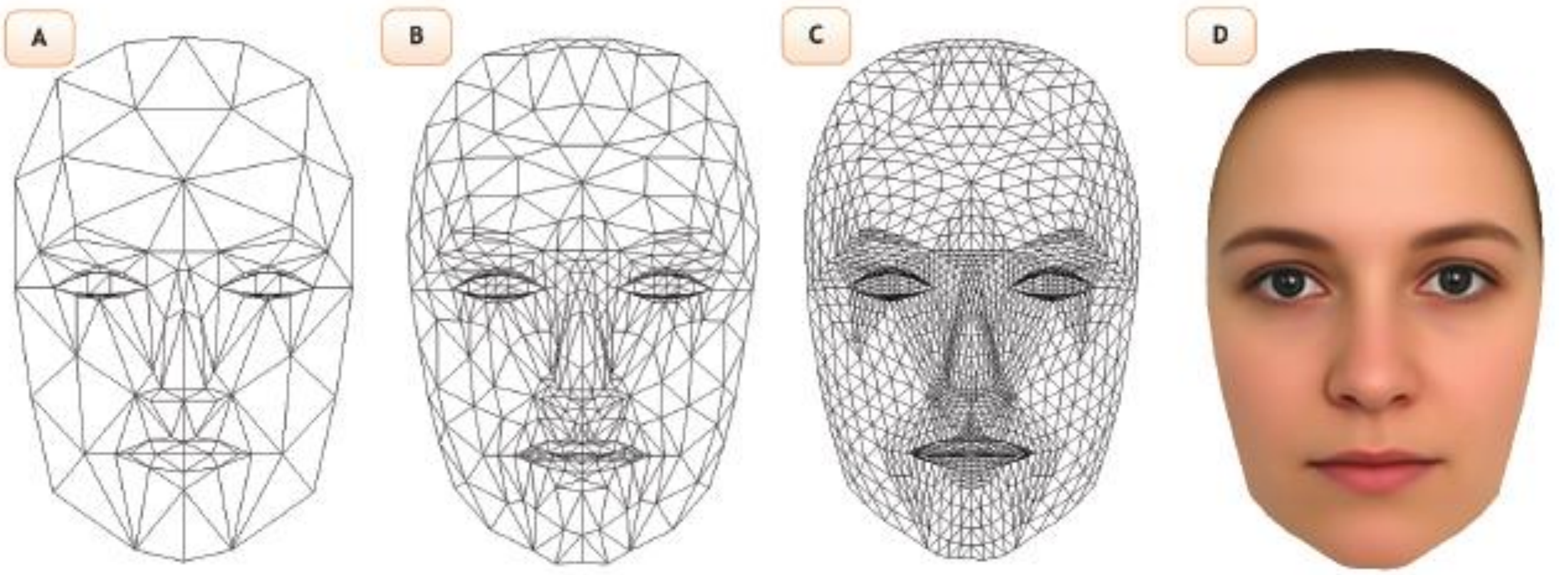
Feature point representation :



What level of detail do we need?



Depends on how the Mesh is Rendered or Texture Mapped



Tradeoff between Detail and Cost

- More detail on a mesh means higher Storage cost.

- More faces means more time to render: sometimes graphics cards specify how many triangles it can render per second.

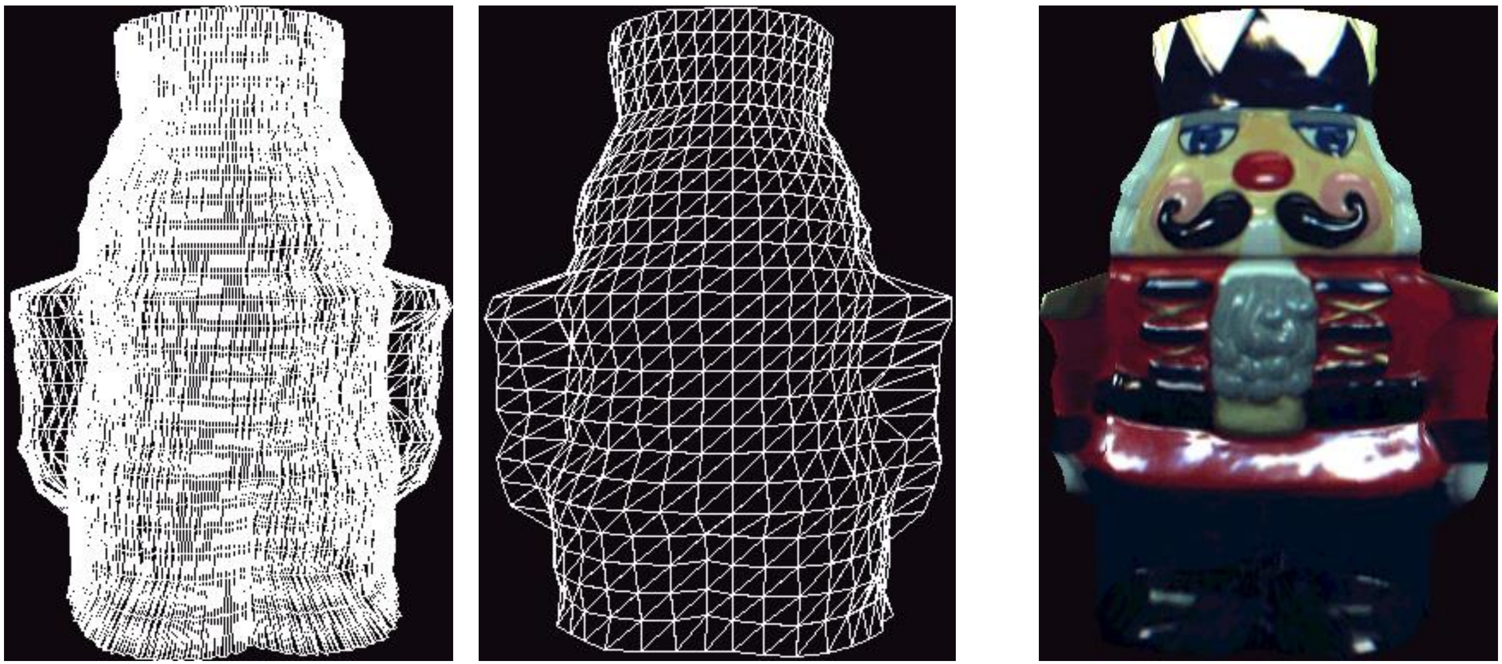
Simplification approaches

- Discrete LOD v.s. Continuous LOD

- View dependent v.s. View-independent
- Region of interest (Fovea)
- Perceptually based

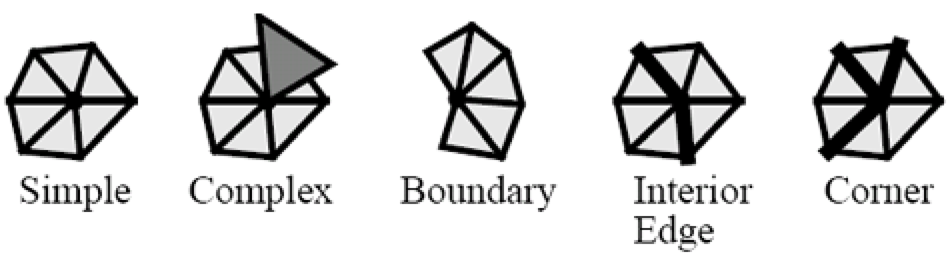
Feature point extraction &
Simplification techniques

Sub-sampling — regular mesh



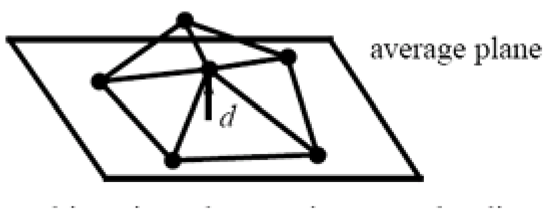
Decimation of triangle meshes (Schroeder et.al.1992)

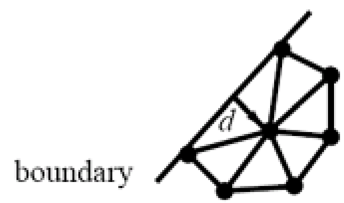
- Decimation criteria
  - Each vertex is assigned one of the following:
    - Simple (interior edge/(2 feature edges) and corner/...), complex, & boundary.
    - Complex vertex (non-manifold) is not deleted.



Decimation of triangle meshes (continued)

- Decimation criteria
  - If a vertex is within the specified distance *d* to the average plane (of the surrounding Δs), it may be deleted.

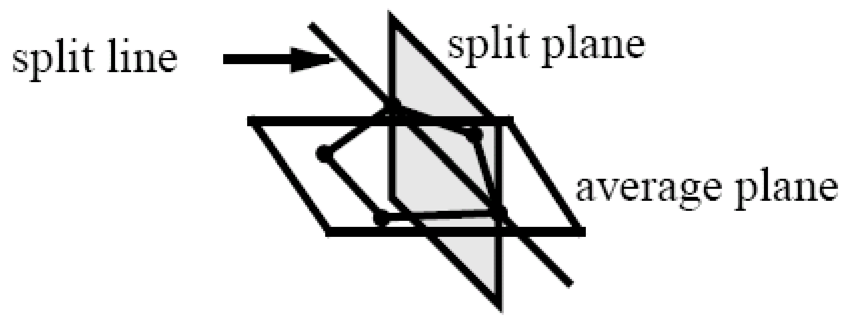




  - Boundary & interior edge vertices use the distance to edge criterion.
  - Relatively small Δs with large feature angles, contributing little to surface property, are removed.

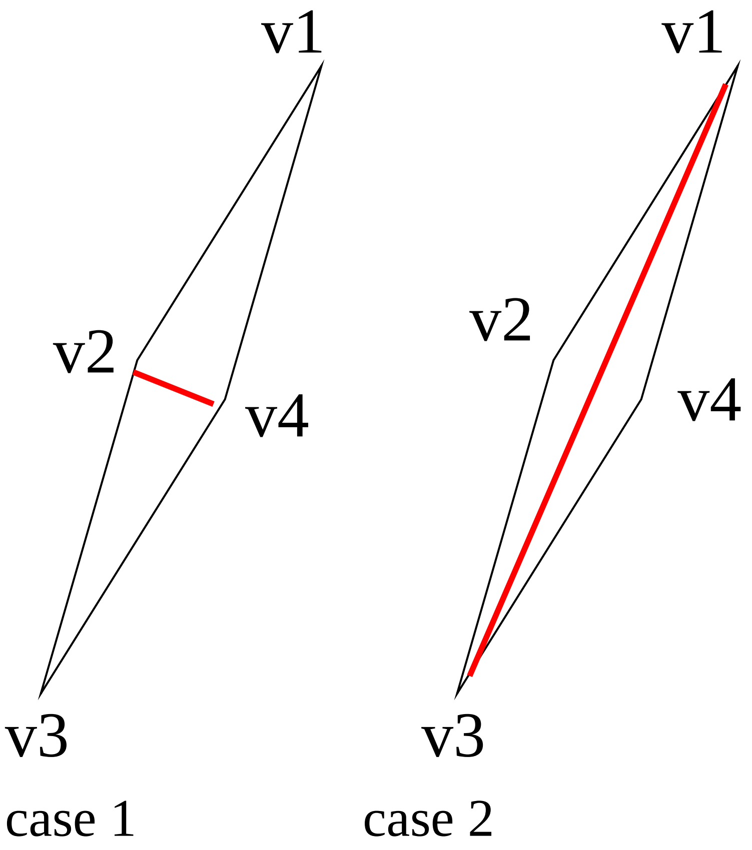
Decimation of triangle meshes (continued)

- Triangulation
  - Divide-and-conquer until only 3 vertices left.
  - The best splitting plane yields the max. aspect ratio.
    – The aspect ratio is the min. distance of the loop vertices to the split plane, divided by the length of the split line (constrained to > 0.1).



Decimation of triangle meshes (continued)

Aspect Ratio Example



Let d_i be the min. distance of v_i from the split plane.

Step1

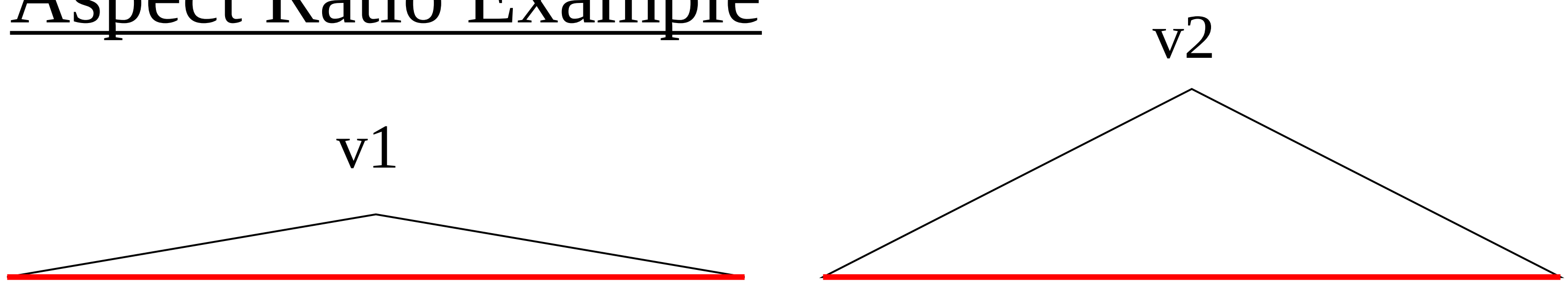
Compute case1=min{d1, d3} and case2=min{d2, d4}

Step2

Select max{case1, case2}

Decimation of triangle meshes (continued)

Aspect Ratio Example



Assuming the split lines have equal length. Since $d_2 > d_1$, split line2 will produce better aspect ratio.

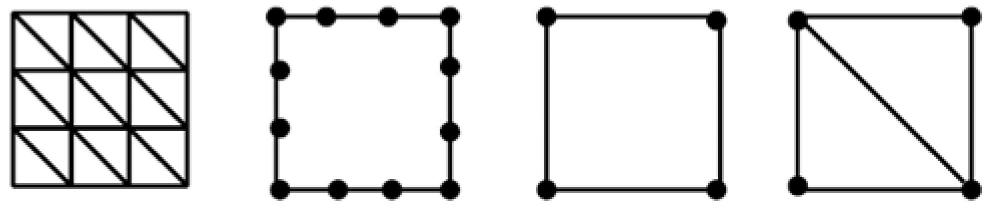
The goal is to generate more balanced triangles instead of sliver triangles.

Decimation of triangle meshes (continued)

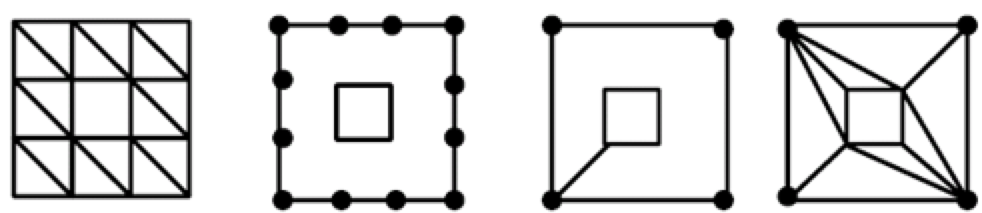
- Review
  - Use local operation on geometry to reduce the # of Δs.
  - Preserve the original topology.
  - Make multiple passes over all vertices.
  - A vertex & the associated Δs are deleted if the specified decimation criteria are met.
  - The resulting hole is patched by local triangulation.
  - Terminate when the required # of Δs is reduced.

Geometric optimization (Hinker et.al.1993)

•Merge coplanar and nearly coplanar Δs.

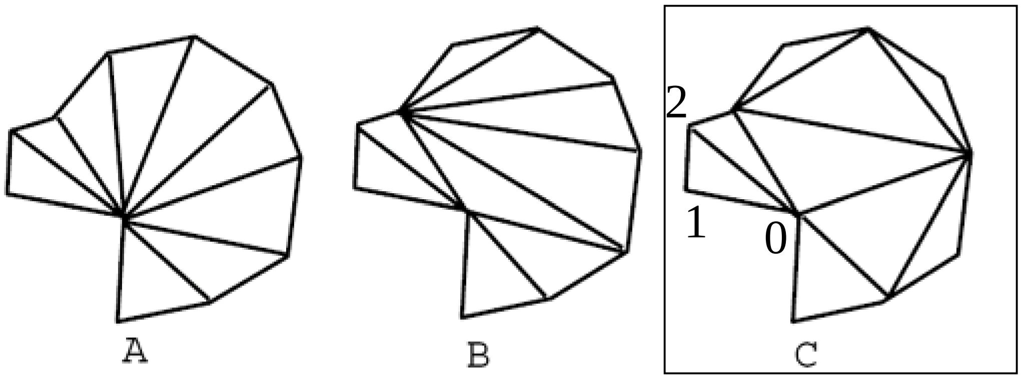


- Start from a representative normal, add an adjacent Δ if the inter-normal angle is within a specified ε.
- Replace the representative normal with the average normal.
- Preserve holes.



Geometric optimization (continued)

- Triangulation
  - Traverse the Δ 0-1-2.
  - Form segment 0-2 if it does not intersect any other segment & lies within the hole. Delete vertex 1.
  - Repeat the above for Δ 2-3-4. (Jump 1 on Starting Vertex)
  - If intersection occurs, the starting vertex is increased by one, i.e. 3-4-5.



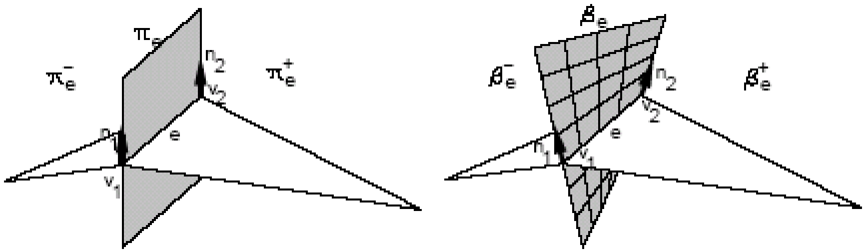
  - Can produce poorly proportioned Δs, but easy and fast to implement (c).

Simplification Envelopes (Cohen et.al. 1996)

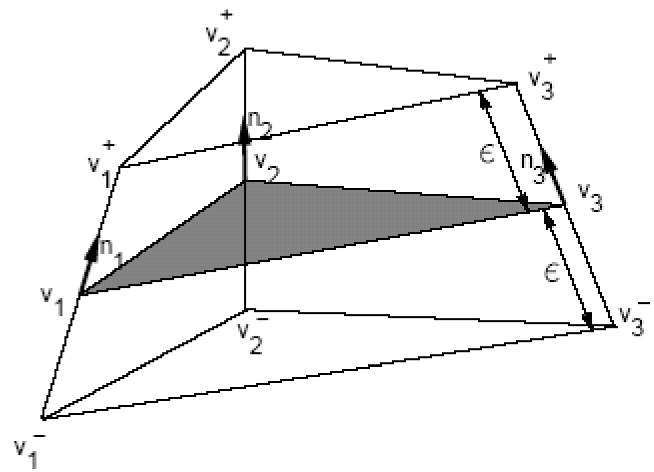
- Generate hierarchy of LOD.
- Guarantee an approximation is within ε (+/-) distance from the original model.
- Preserve genus.
- Prevent self-intersection.
- Preserve sharp features.
- Allow variation of approximation distance across different portions of a model.

Simplification Envelopes (continued)

- Envelope Computation



Edge Half-spaces



$\mathrm{Coord}(v_i^{+}) = \mathrm{Coord}(v_i) + \varepsilon\mathbf{n}(v_i)$ and $\mathbf{n}(v_i^{+}) = \mathbf{n}(v_i)$

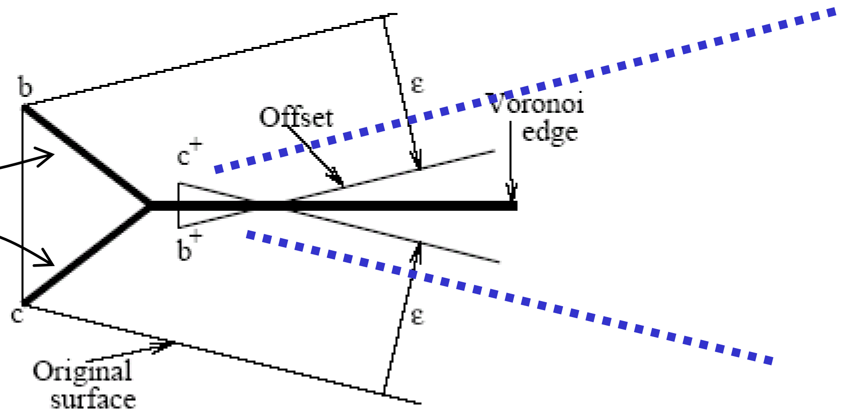
$-\varepsilon$ can be similarly defined in the opposite direction.

Simplification Envelopes (continued)

- Avoid self-intersection

normal of b

normal of c

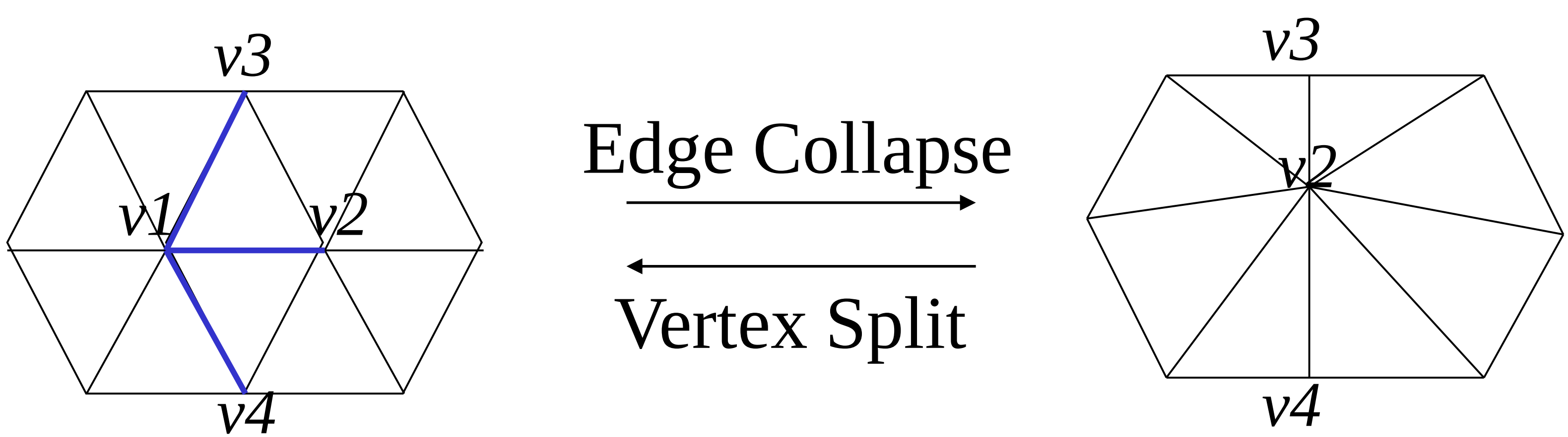


- Hole creation & hole filling.

Simplification Envelopes (continued)

Progressive meshes (PM)

- (Hoppe 1996)
  - Lossless CLOD.
  - Apply edge collapse (simplification) & vertex split (refinement).



Progressive meshes (continued)

- Energy Minimization
  - Estimate energy cost ΔE for each edge collapse transformation, and store in priority queue.
  - In each iteration, perform the transformation with lowest ΔE.
  - Recompute the priorities of edges in the neighborhood of this transformation.

Quadric Error Metric (QEM)

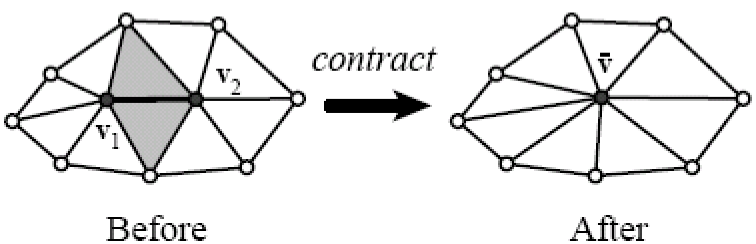
- (Garland et.al. 1997)
  - Use iterative contractions of vertex pairs.
  - Maintain surface error approximations using quadric matrices.
  - Able to join unconnected regions of models.
  - Select the set of valid pairs at initialization time, based on the assumption that, in a good approximation, points do not move far from their original positions.

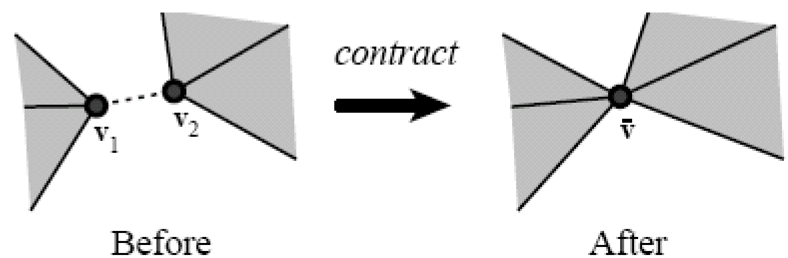
Quadric Error Metric (continued)

- Cost/error of contraction Δv for $(v1, v2) \rightarrow v$
  - Associate a cost with each vertex.
  - Compute an initial cost by accumulating the planes for the Δs which meet at that vertex. Note that the initial error estimate for each vertex is 0, because the vertex lies in the planes of all its incident Δs.
  - Define the error of the vertex w.r.t this set of planes as the sum of squared distances to the planes.

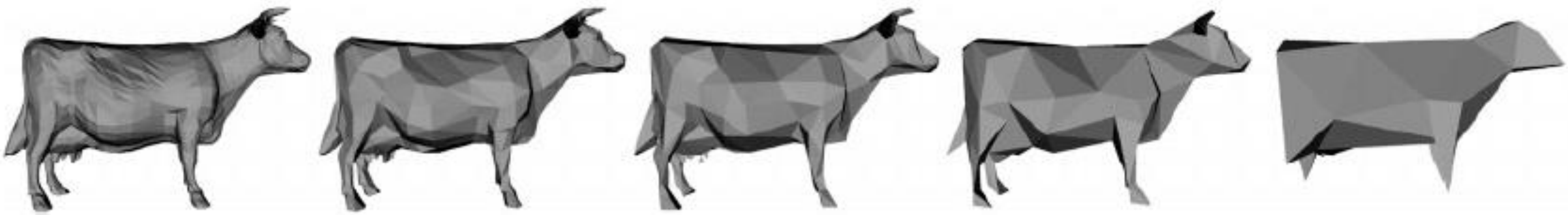
Quadric Error Metric (continued)

- Cost/Error of contraction Δv for $(v1, v2) \rightarrow v$
  - Select $v1$, $v2$ or $(v1 + v2)/2$ as position of v, depending on which one produces the lowest value of Δv.
  - $Qv = Q1 + Q2$.





Quadric Error Metric (continued)

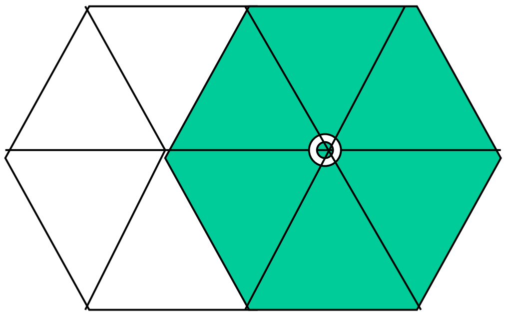


Early version

Vertex clustering (Low et al. 1997)

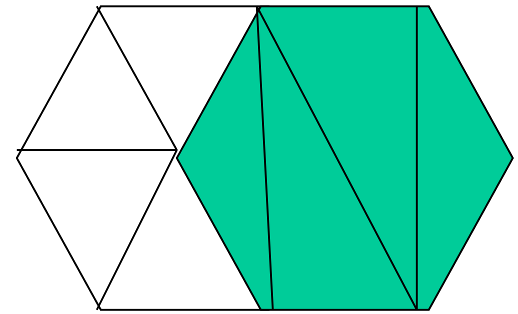
- Determine the closeness of the vertices.
- Vertices are grouped together based on their proximity.
- A new representative vertex is created to replace them.

Simplification & Refinement

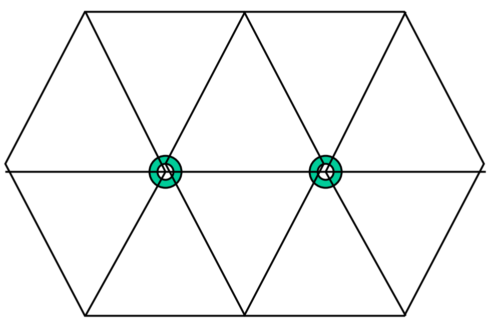


Vertex Removal

Vertex Insertion

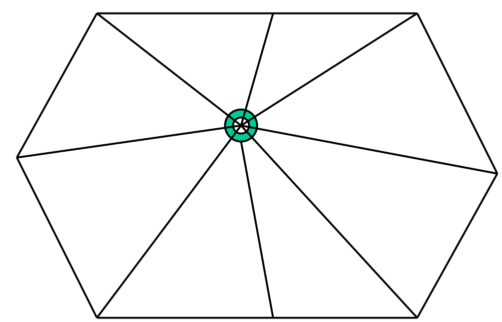


Subset



Edge Collapse

Vertex Split



Not a subset

Discussion

1. Vertex relocation
2. Continuous LOD
3. Prevent drastic simplification
4. Use priority queue
5. Preserve topology

A. Simplification envelope
B. Progressive meshes
C. Quadric error
D. Vertex clustering
F. Geometric optimization

- How to evaluate the performance of these techniques ?
- Can we assess quality solely based on geometric computation ?