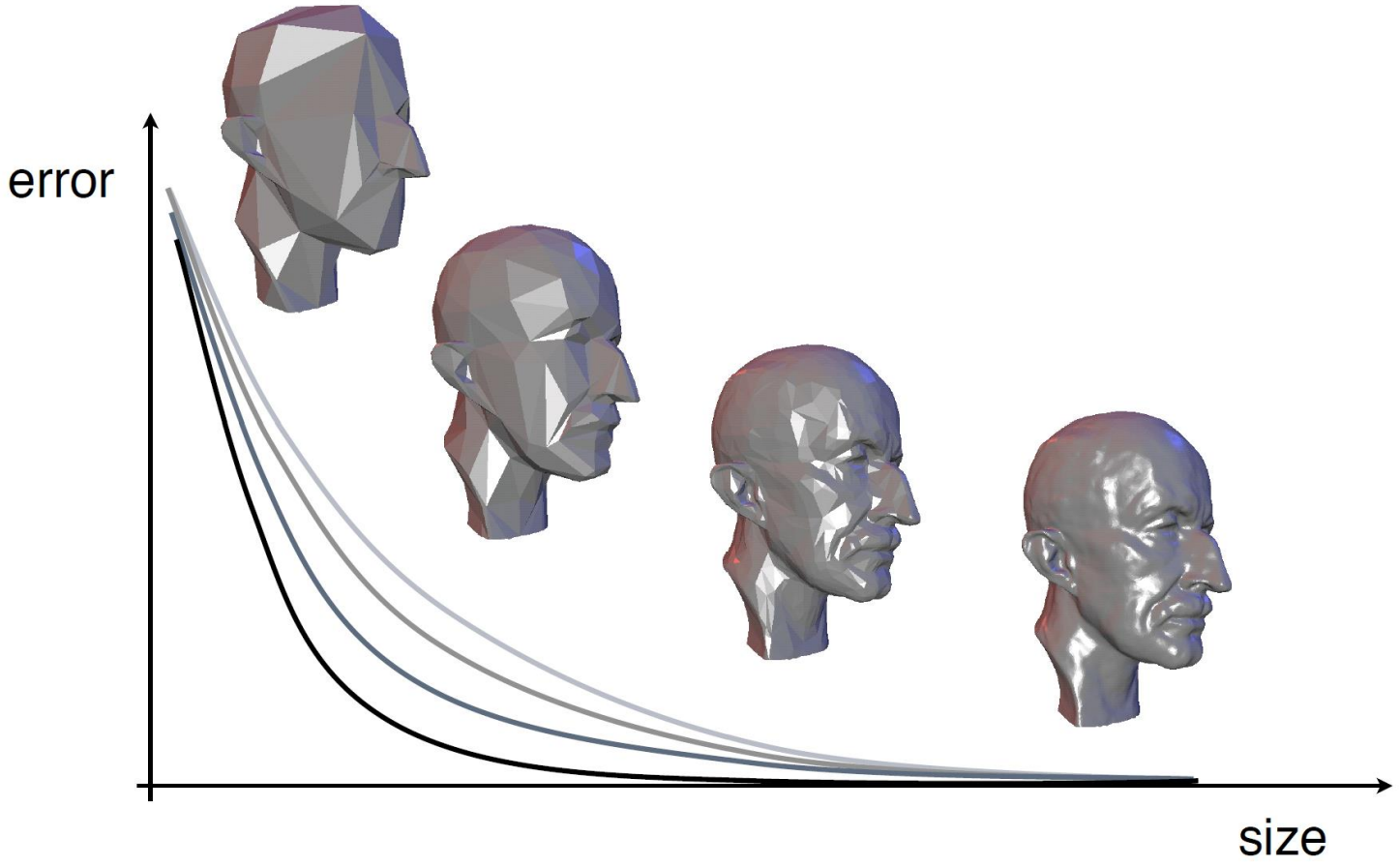
Geometric Design

Level-of-Detail Rendering: The Math

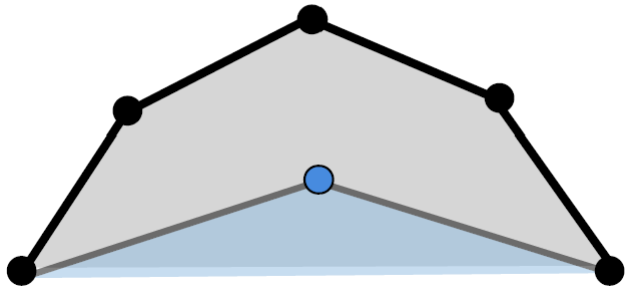
CS 415: Game Development

Professor Eric Shaffer

Mesh Simplification: Polygon Count vs Quality



Here ***error*** means some metric that measures how far away from the original surface the low poly surface is

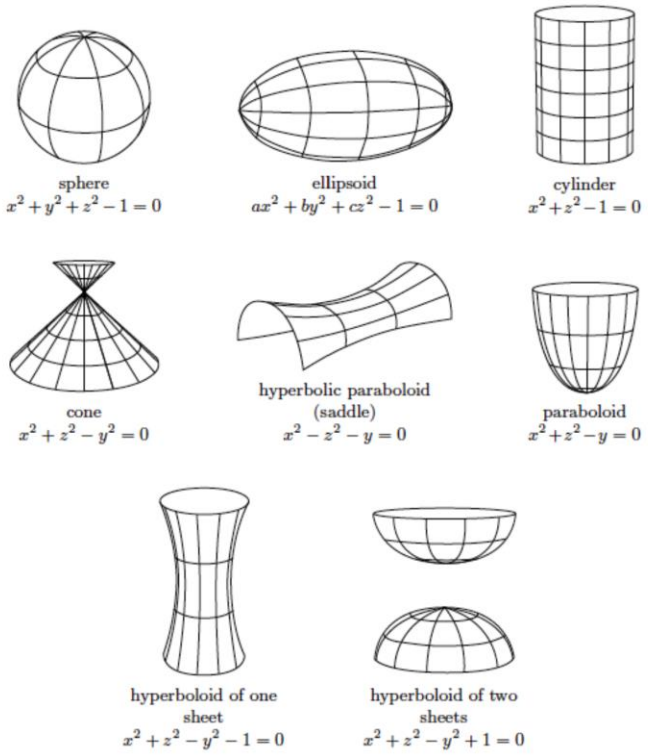


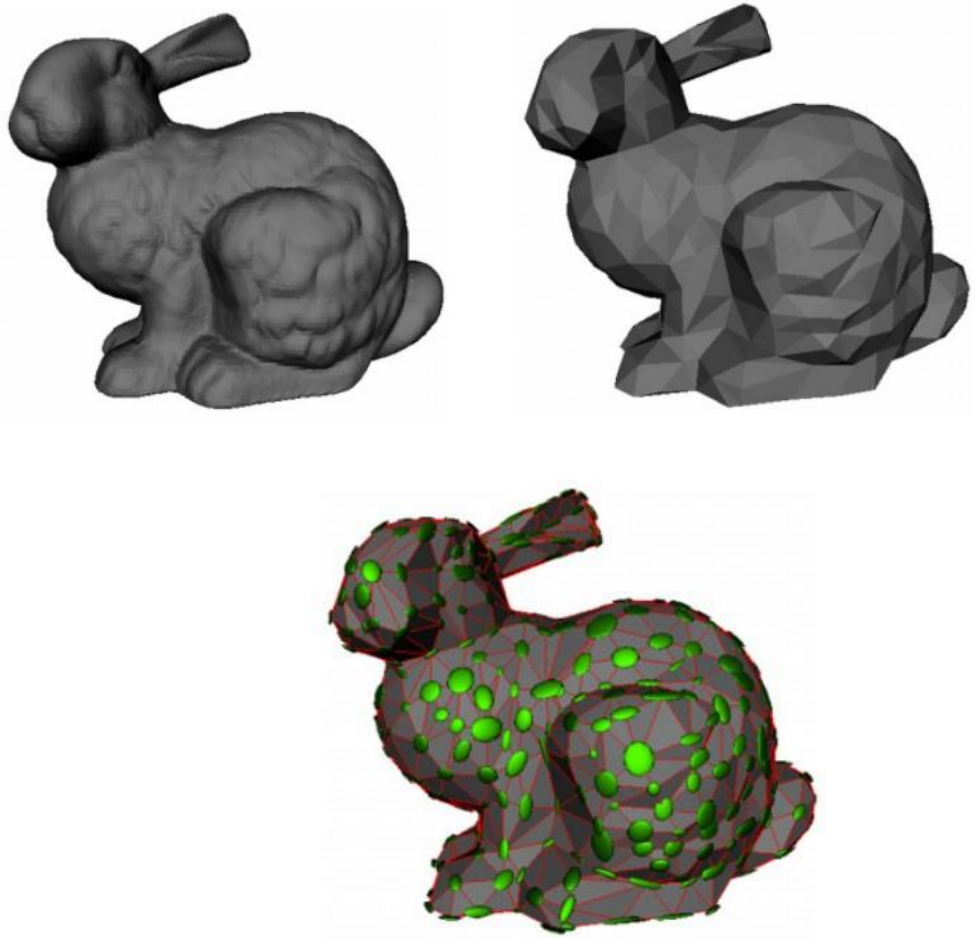
…measuring the difference between two surfaces is complicated and there are many different ways people attempt to do it

ILLINOIS

Quadric Error Metric: What is a Quadric?

Quadrics are surfaces that can be expressed as a second degree polynomial in x, y, and z.



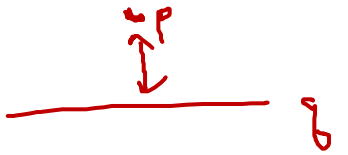


With QEM there is a function at each vertex that measures the error that occurs when moving that vertex to a new position.

The ellipsoids show a set of points around the vertex that all generate the same error.

ILLINOIS

Error Quadrics



- Squared distance to plane

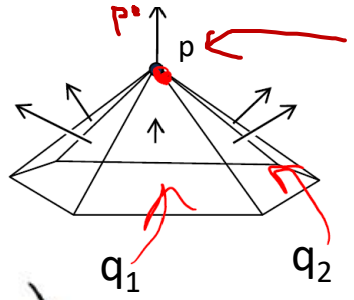
$$p = (x, y, z, 1)^T, \quad q = (a, b, c, d)^T$$

Using implicit form of a plane equation ax+by+cz+d=0

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

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

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

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

You can compute the sum of squared distances from p to N planes using a single 4x4 matrix

You simply sum up the N matrices Q_{q_i} component-wise and use it as shown here.

For a vertex v you can multiply out v^TQv and see it generates a quadric surface:

$$v^TQv = q_{11}x^2 + 2q_{12}xy + 2q_{13}xz + 2q_{14}x + q_{22}y^2 + 2q_{23}yz + 2q_{24} + q_{33}z^2 + 2q_{34}z + q_{44}$$

Vertex Placement

$$\mathbf{Q} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{b}^T & c \end{bmatrix}$$

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

Point that minimizes $Q(v)$ occurs when

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

The gradient of $Q(v)$ is

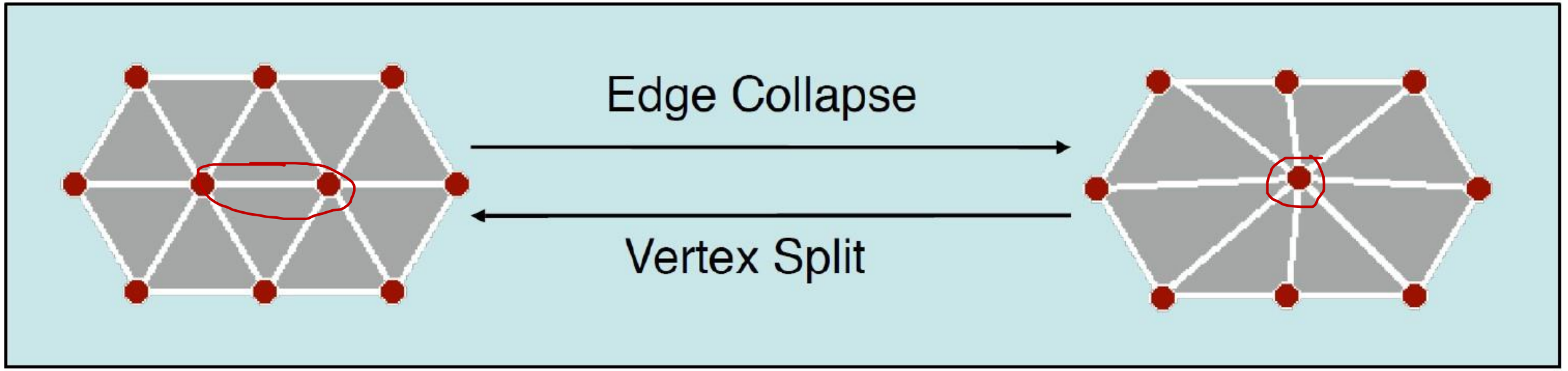
$$\nabla Q(\mathbf{v}) = 2\mathbf{A}\mathbf{v} + 2\mathbf{b}$$

Solving for $\nabla Q(\mathbf{v}) = 0$, we find that the optimal position is

$$\bar{\mathbf{v}} = -\mathbf{A}^{-1}\mathbf{b}$$

ILLINOIS

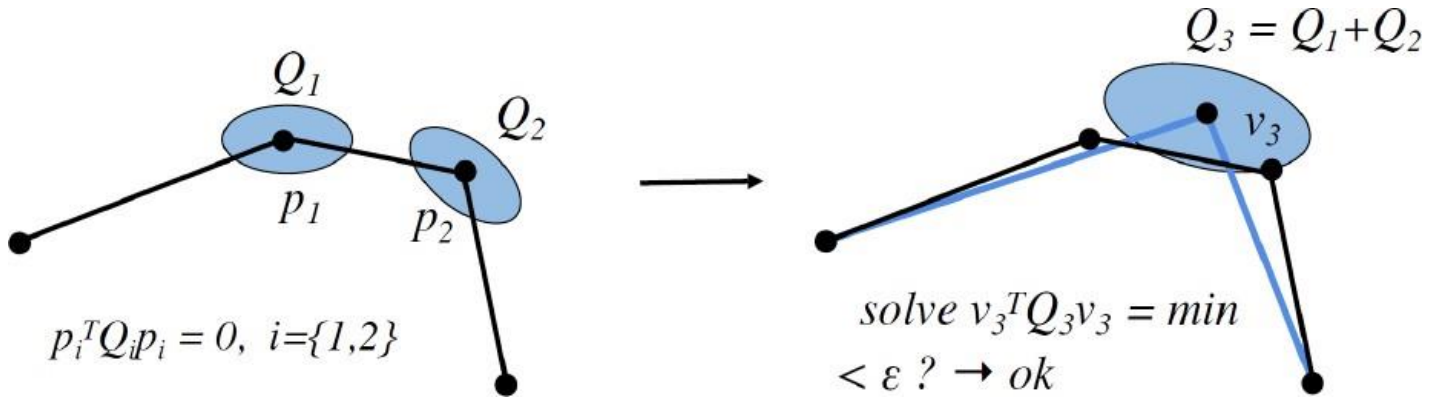
Incremental Simplification: Edge Collapse



- Merge two adjacent triangles
- Define new vertex position

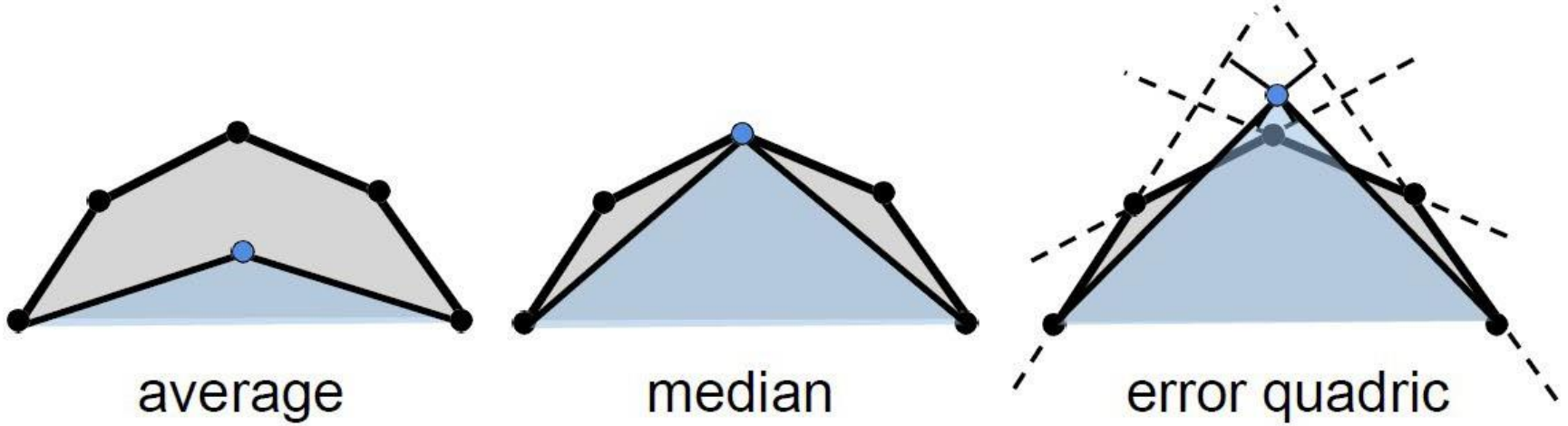
Incremental Simplification Algorithm

1. Compute Quadric for each vertex
2. Create a priority queue of all possible edge collapses $p_1 + p_2$
   1. For each edge collapse compute $Q_3 = Q_1 + Q_2$
   2. Compute new vertex v_3 such that $v_3^T Q_3 v_3 = min$
   3. Use the error $v_3^T Q_3 v_3$ as the key in priority queue
3. Choose collapse with least error…update quadrics and repeat

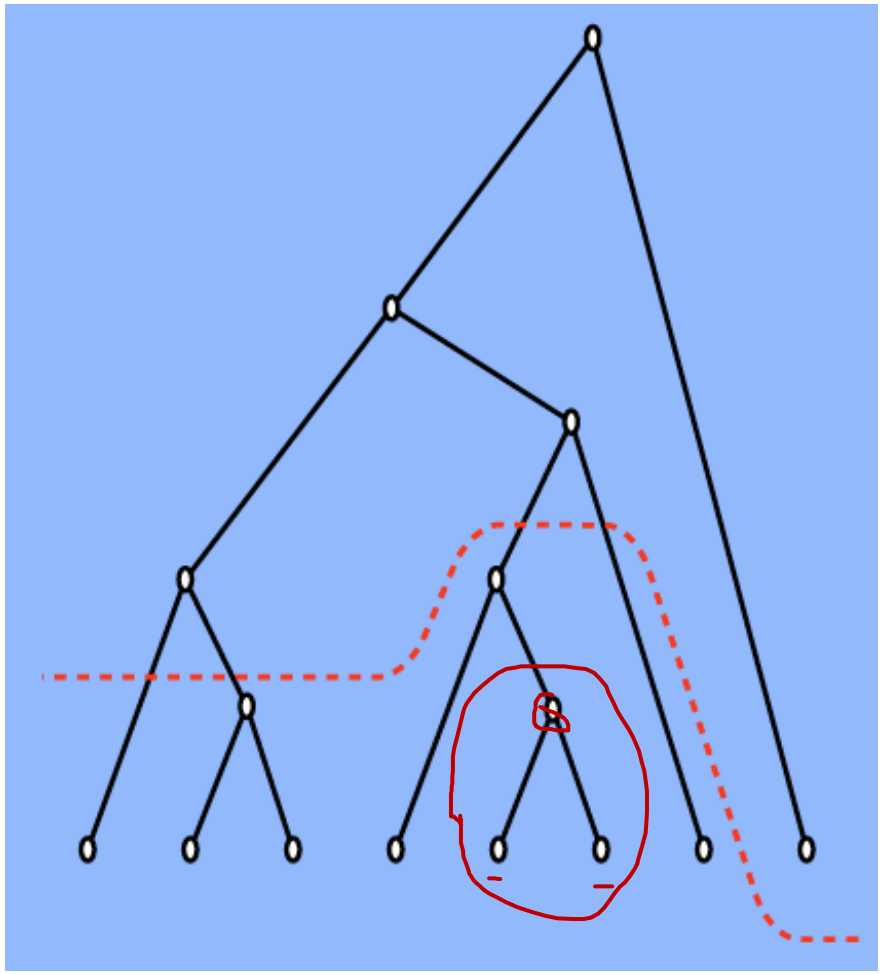


ILLINOIS

Comparison of Vertex Placements



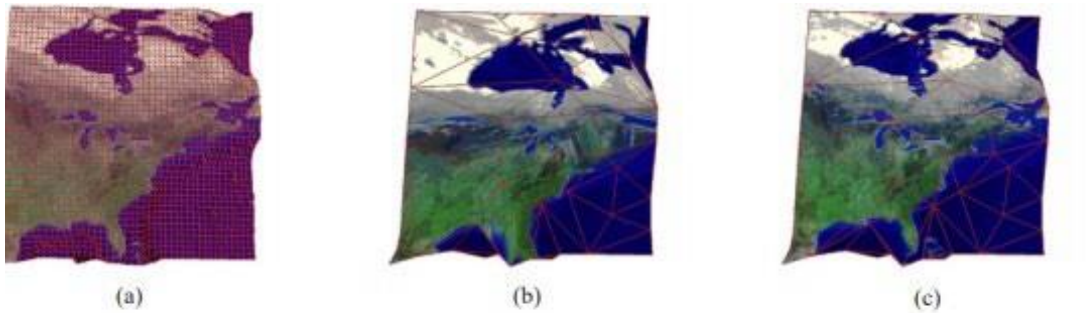
Continuous LOD using Vertex Hierarchies



- Each original vertex in mesh is a leaf in this diagram
- An edge contraction makes vertices siblings and creates a parent
- A cut through the tree representsset of contractions applied to the mesh
- Could do CLOD (sort of) by using screen space metric to determine cut
- …adjust cut each frame…

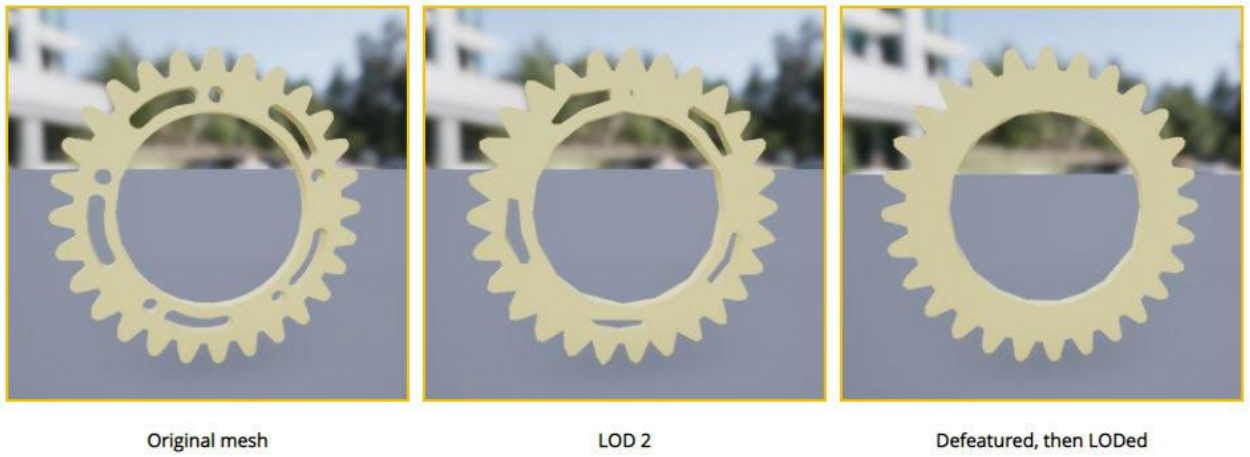
In Practice LOD Gets More Complicated

- Error metric should incorporate color and texture information
  - …don’t want to merge discontinuous parts of texture if possible



Geometry & texture: A 3,872 face model (a) reduced to 53 faces without (b) and with (c) updating texture coordinates.

- Features may need to be preserved or removed…requires artist input



ILLINOIS