Lecture 12:

More Geometric Processing

Computer Graphics

CMU 15-462/15-662, Spring 2016

What if we want fewer triangles?

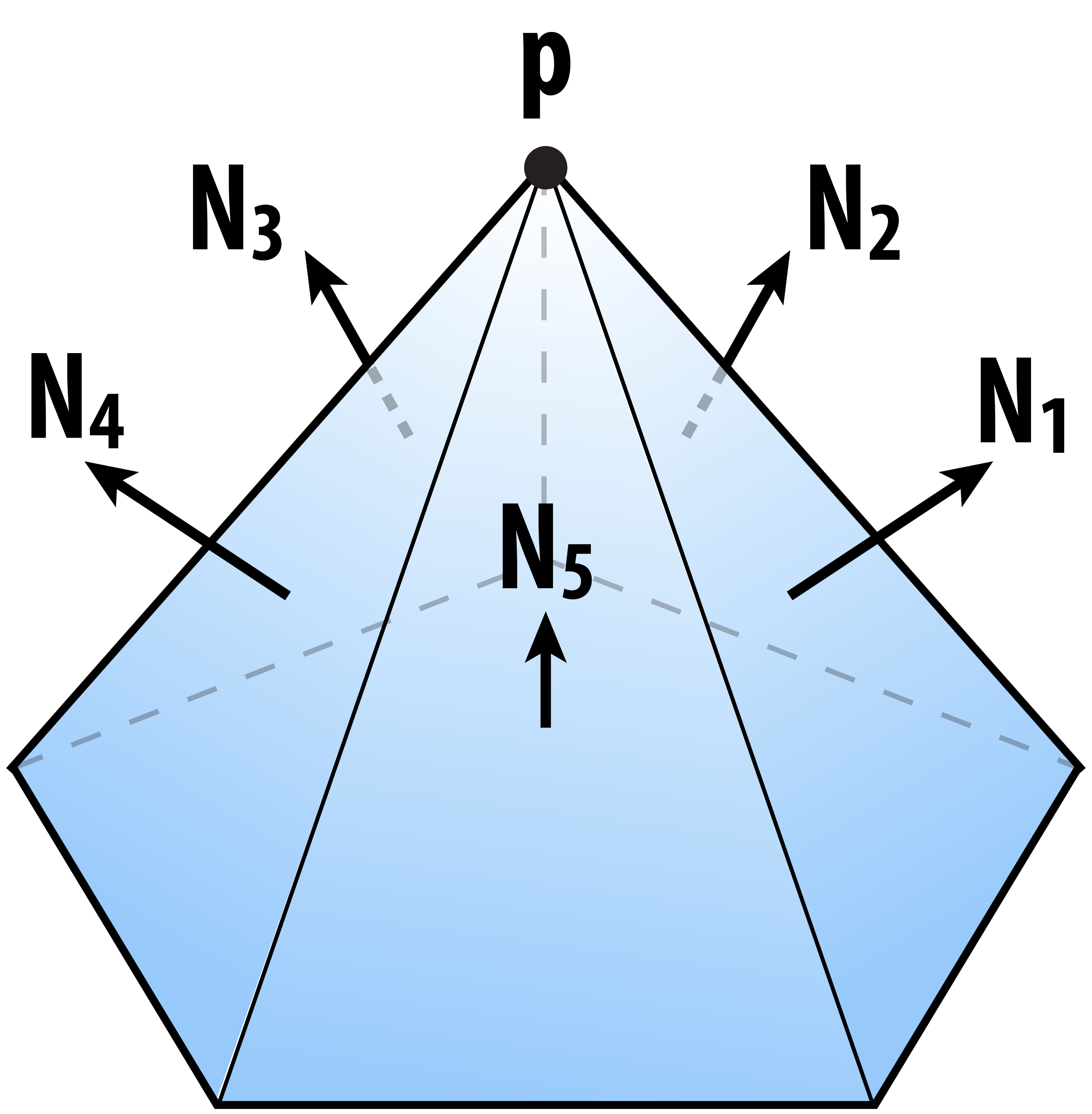
Simplification via Quadric Error Metric

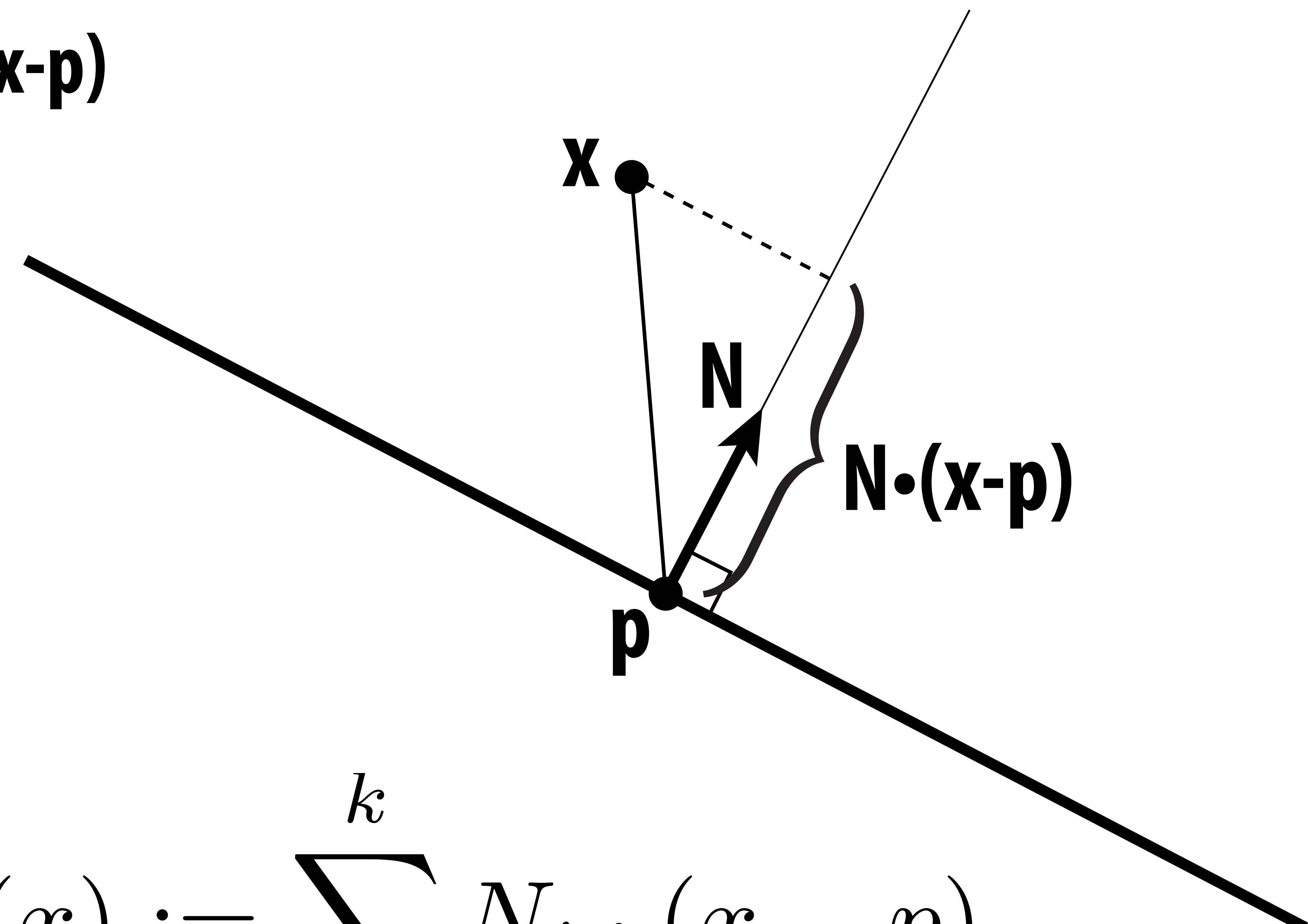
- One popular scheme: iteratively collapse edges
- Which edges? Assign score with quadric error metric*
  - approximate distance to surface as sum of distance to aggregated triangles
  - iteratively collapse edge with smallest score
  - greedy algorithm... great results!

*invented here at CMU! (Garland & Heckbert 1997)

Quadric Error Metric

- Approximate distance to a collection of triangles
- Distance is sum of point-to-plane distances
  - Q: Distance to plane w/ normal N passing through point p?
  - A: d(x) = N•x - N•p = N•(x-p)
- Sum of distances:





$$d(x) := \sum_{i=1}^{k} N_i \cdot (x - p)$$

Quadric Error - Homogeneous Coordinates

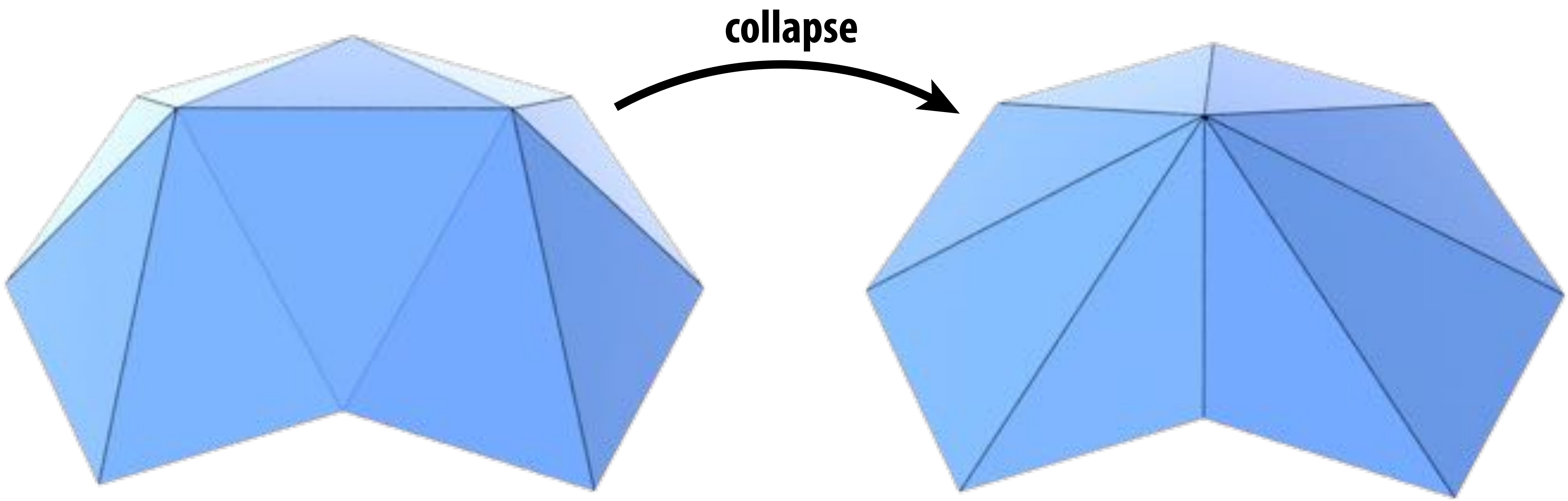
- Suppose in coordinates we have
  - a query point (x,y,z)
  - a normal (a,b,c)
  - an offset d := -(px,py,pz) • (a,b,c)

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

- Then in homogeneous coordinates, let
  - u := (x,y,z,1)
  - v := (a,b,c,d)
- Signed distance to plane is then just u•v = ax+by+cz+d
- Squared distance is $(u^Tv)^2 = u^T(vv^T)u =: u^TQu$
- Key idea: matrix Q encodes distance to plane
- Q is symmetric, contains 10 unique coefficients (small storage)

Quadric Error of Edge Collapse

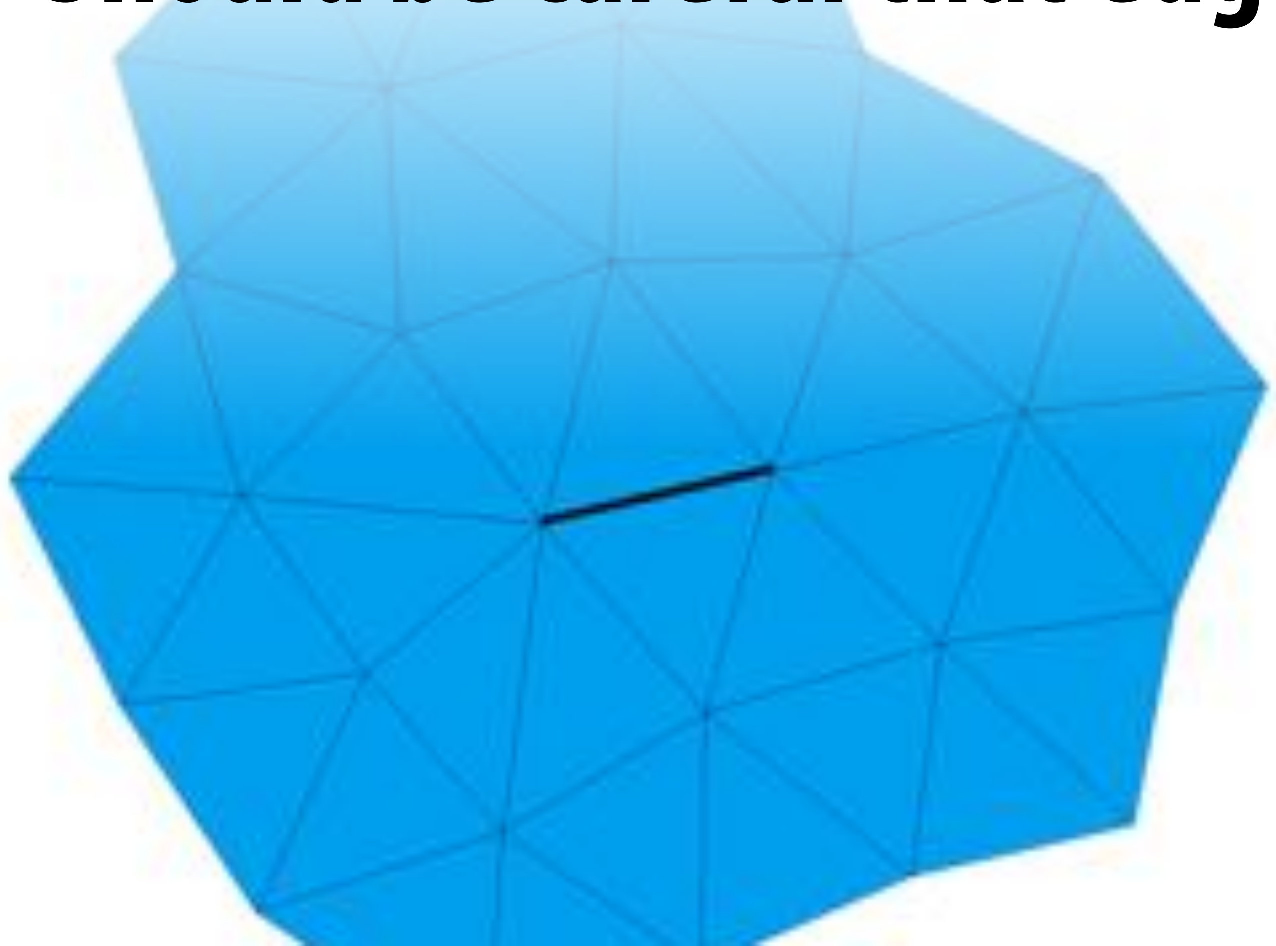
- How much does it cost to collapse an edge?
- Idea: compute edge midpoint, measure quadric error

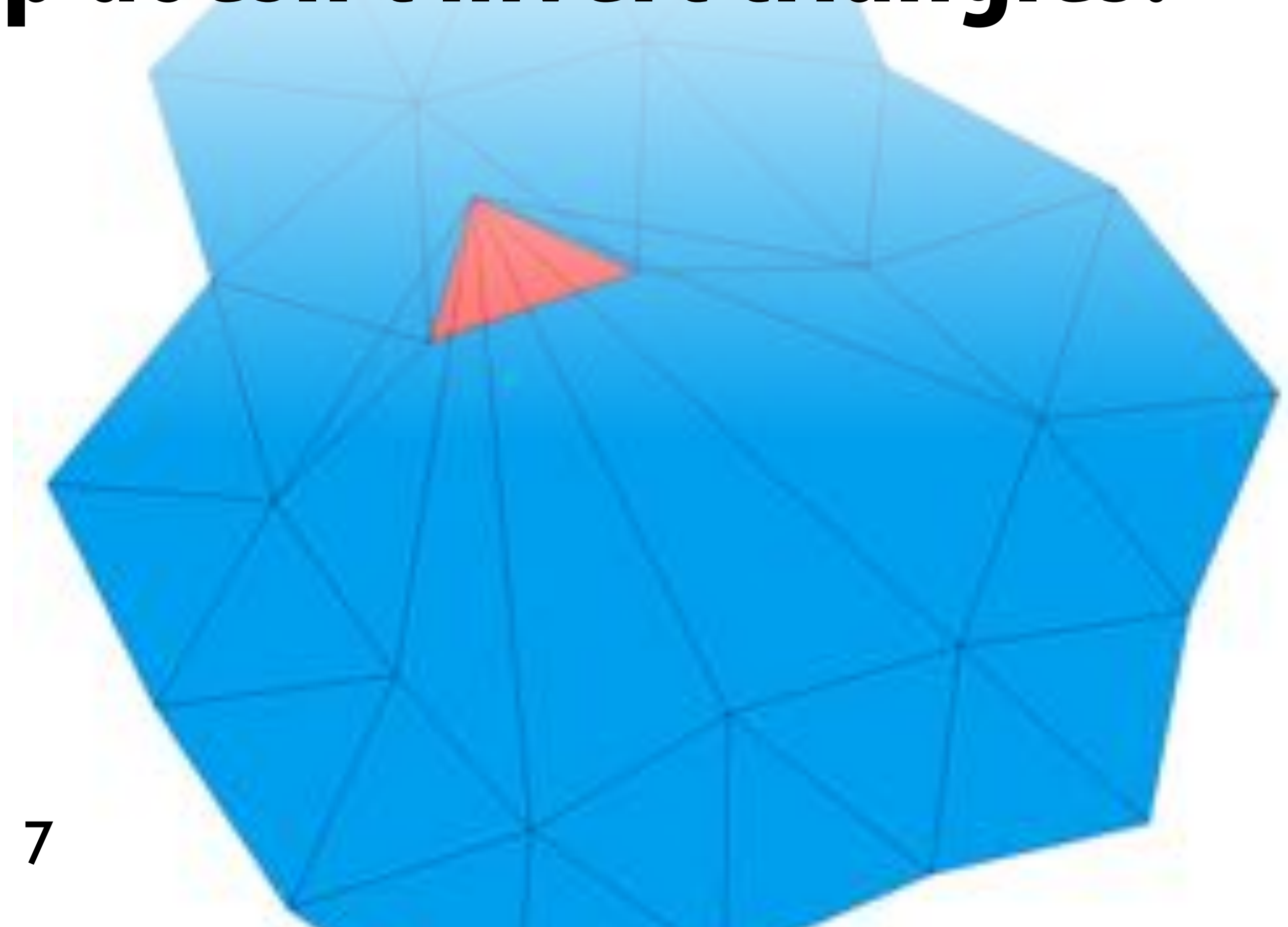


- Better idea: use point that minimizes quadric error as new point!
- (More details in assignment; see also Garland & Heckbert 1997.)

Quadric Error Simplification

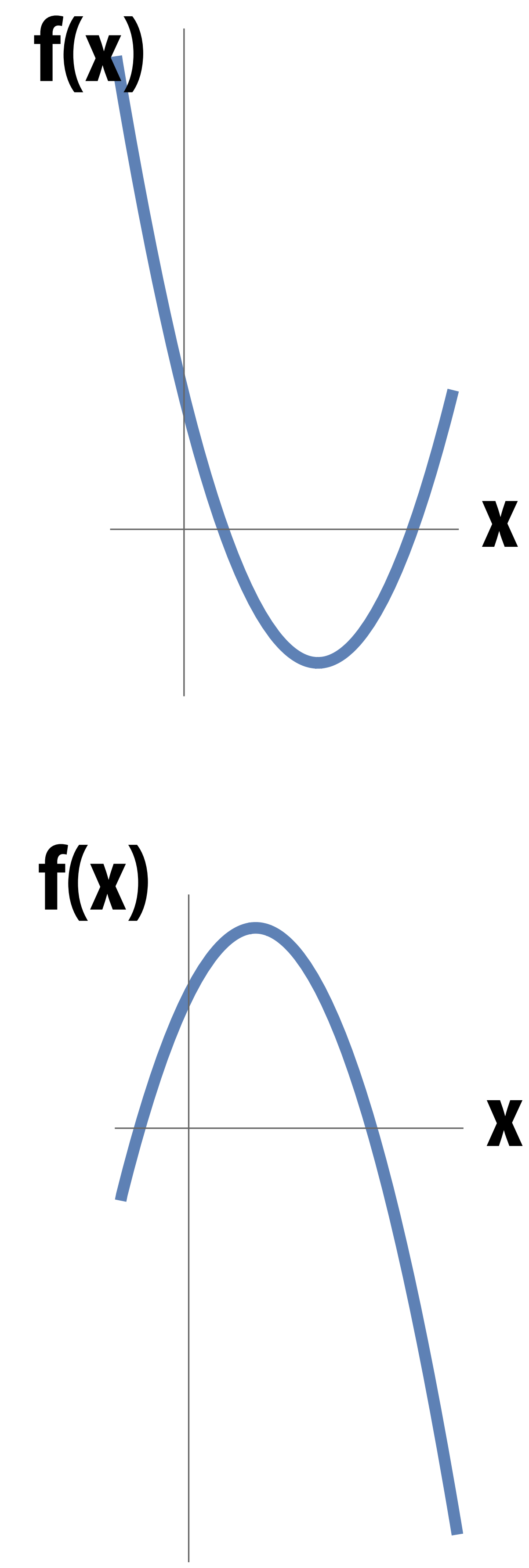
- Compute Q for each triangle
- Set Q at each vertex to sum of Qs from incident triangles
- Until we reach target # of triangles:
  - collapse edge (i,j) with smallest cost to get new vertex k
  - add Q_i and Q_j to get new quadric Q_k
  - update cost of any edge touching new vertex k
- Store edges in priority queue to keep track of minimum cost
- Should be careful that edge flip doesn't invert triangles:





Review: Minimizing a Quadratic Function

- Suppose I give you a function $f(x) = ax^2+bx+c$
- Q: What does the graph of this function look like?
- Could also look like this!
- Q: How do we find the minimum?
- A: Look for the point where the function isn't changing (if we look "up close")
- I.e., find the point where the derivative vanishes



$$f'(x) = 0$$

$$2ax + b = 0$$

$$x = -b/2a$$

(What about our second example?)

Minimizing a Quadratic Form

- A quadratic form is just a generalization of our quadratic polynomial from 1D to nD
- E.g., in 2D: $f(x,y) = ax^2 + bxy + cy^2 + dx + ey + g$
- Can always (always!) write quadratic polynomial using a symmetric matrix (and a vector, and a constant):

$$f(x,y) = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} a & b/2 \\ b/2 & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} d & e \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + g$$

$$= \mathbf{x}^{\mathsf{T}} A \mathbf{x} + \mathbf{u}^{\mathsf{T}} x + g$$

(this expression works for any n!)

- Q: How do we find a critical point (min/max/saddle)?
- A: Set derivative to zero!

$$2A\mathbf{x} + \mathbf{u} = 0$$

$$\mathbf{x} = -\tfrac{1}{2} A^{-1} \mathbf{u}$$

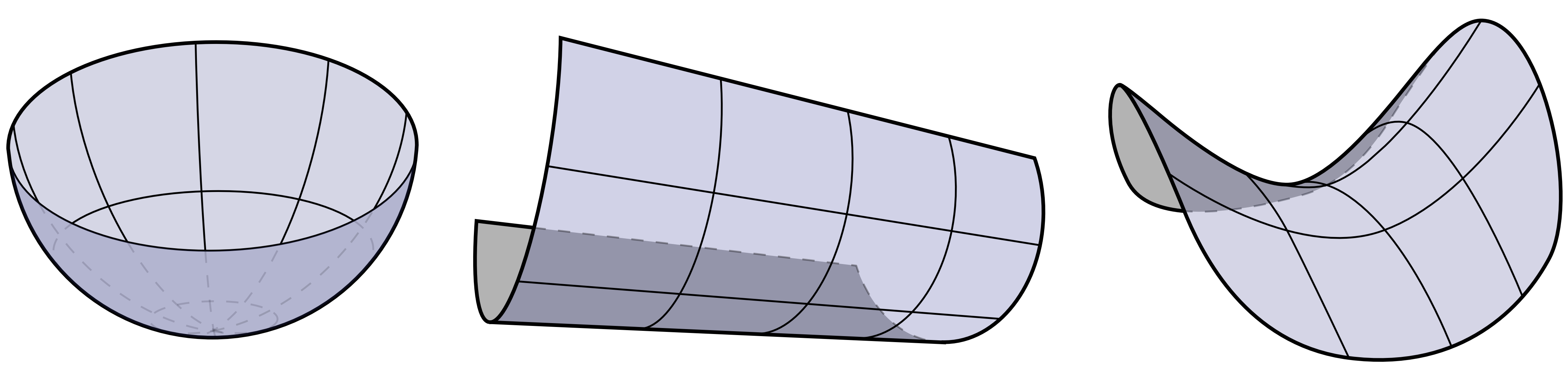
(Can you show this is true, at least in 2D?)

Positive Definite Quadratic Form

- Just like our 1D parabola, critcal point is not always a min!
- Q: In 2D, 3D, nD, when do we get a minimum?
- A: When matrix A is positive-definite:

$$\mathbf{x}^\mathsf{T} A\ \mathbf{x} > 0 \quad \forall\ \mathbf{x}$$

- 1D: Must have xax = $ax^2 > 0$. In other words: a is positive!
- 2D: Graph of function looks like a "bowl":



positive definite — positive semidefinite — indefinite

- Positive-definiteness is **extremely important** in computer graphics: it means we can find a minimum by solving linear equations. Basis of many, many modern algorithms (geometry processing, simulation, ...).

Minimizing Quadratic Error

- Find "best" point for edge collapse by minimizing quad. form

$$\min_u \mathbf{u}^\mathsf{T} K \mathbf{u}$$

- Already know fourth (homogeneous) coordinate is 1!
- So, break up our quadratic function into two pieces:

$$\begin{bmatrix} \mathbf{x}^\mathsf{T} & 1 \end{bmatrix} \begin{bmatrix} B & \mathbf{w} \\ \mathbf{w} & d^2 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}$$

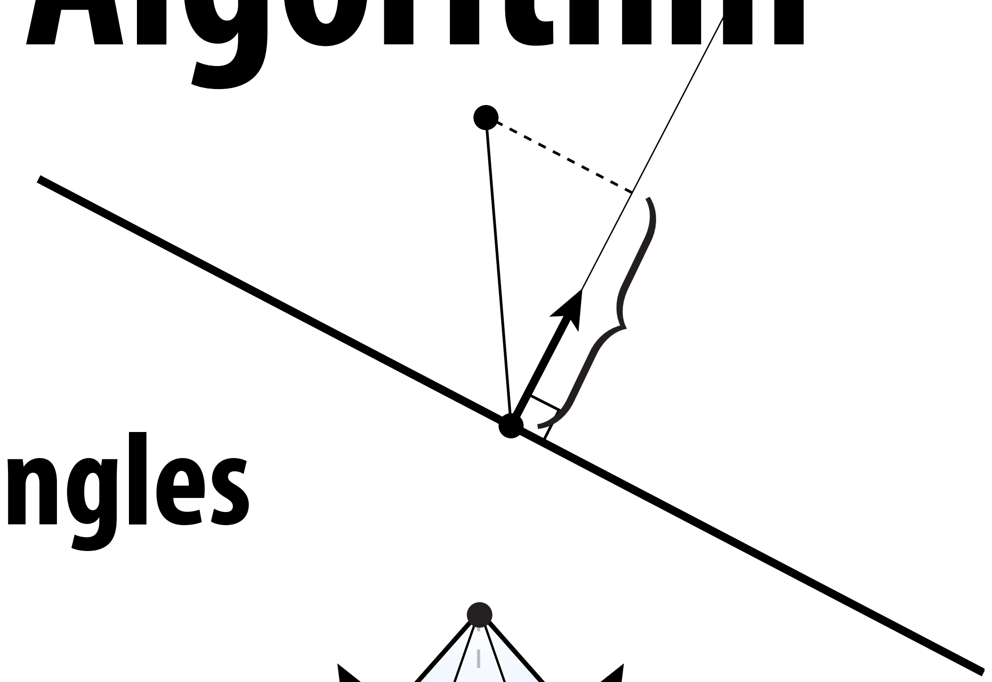
$$= \mathbf{x}^\mathsf{T} B \mathbf{x} + 2\mathbf{w}^\mathsf{T}\mathbf{x} + d^2$$

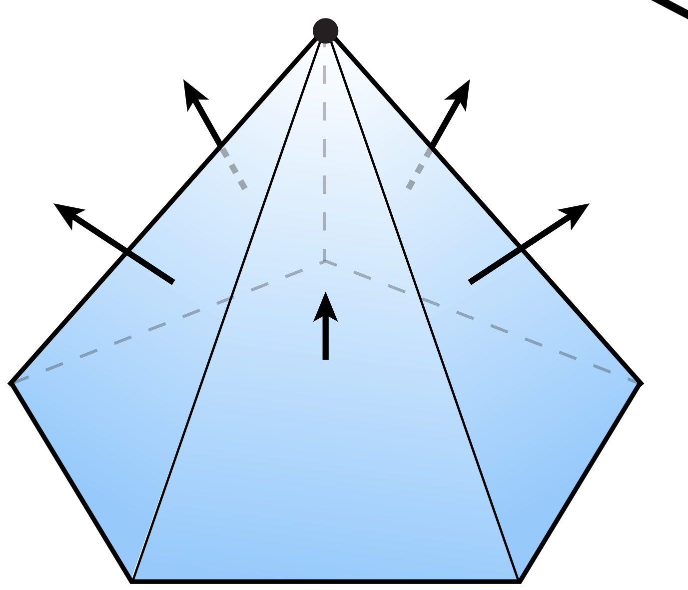
- Now we have a quadratic form in the 3D position x.
- Can minimize as before:

$$2B\mathbf{x} + 2\mathbf{w} = 0 \qquad \Longleftrightarrow \qquad \mathbf{x} = -B^{-1}\mathbf{w}$$

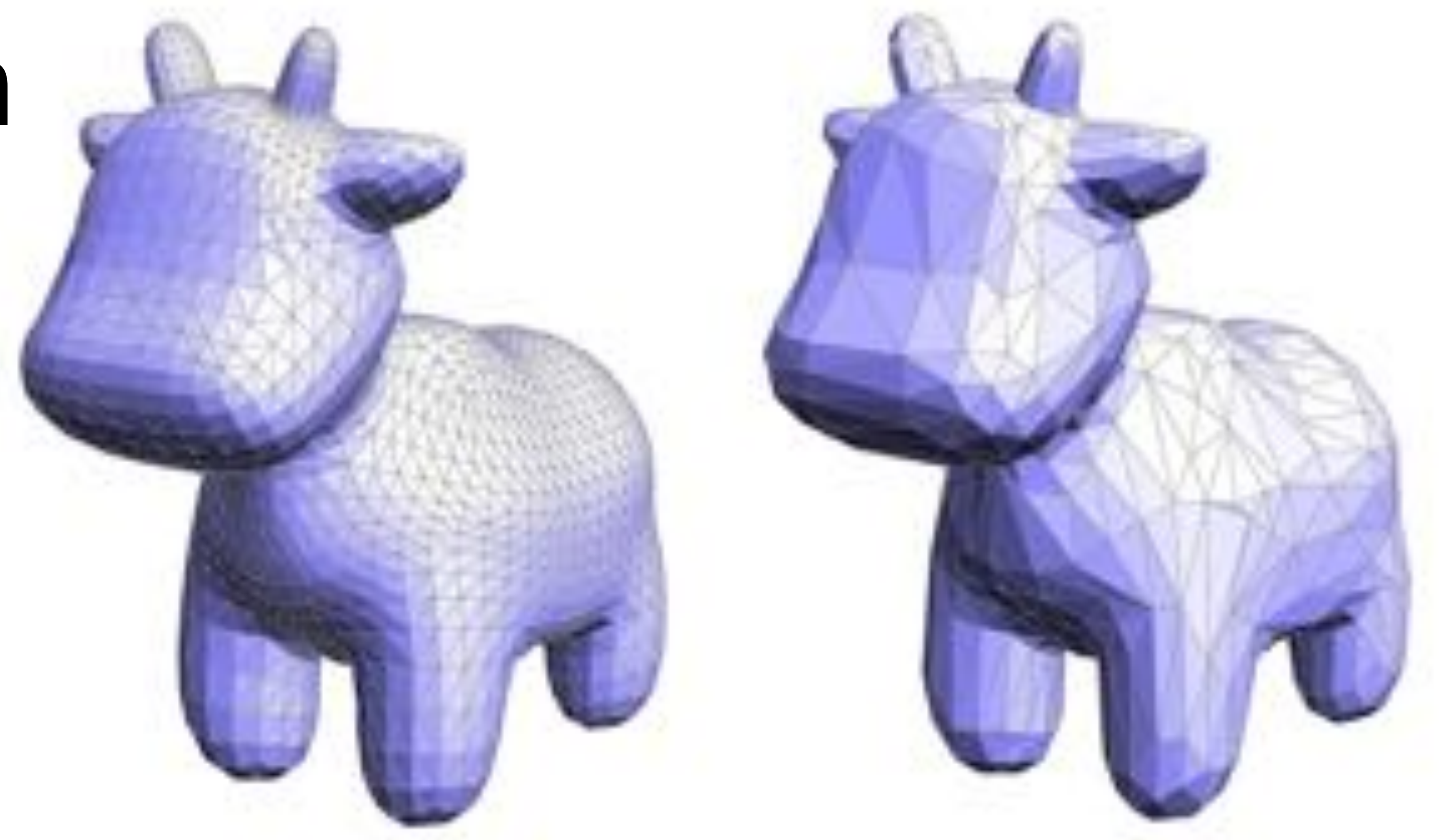
(Q: Why should B be positive-definite?)

Quadric Error Simplification: Final Algorithm





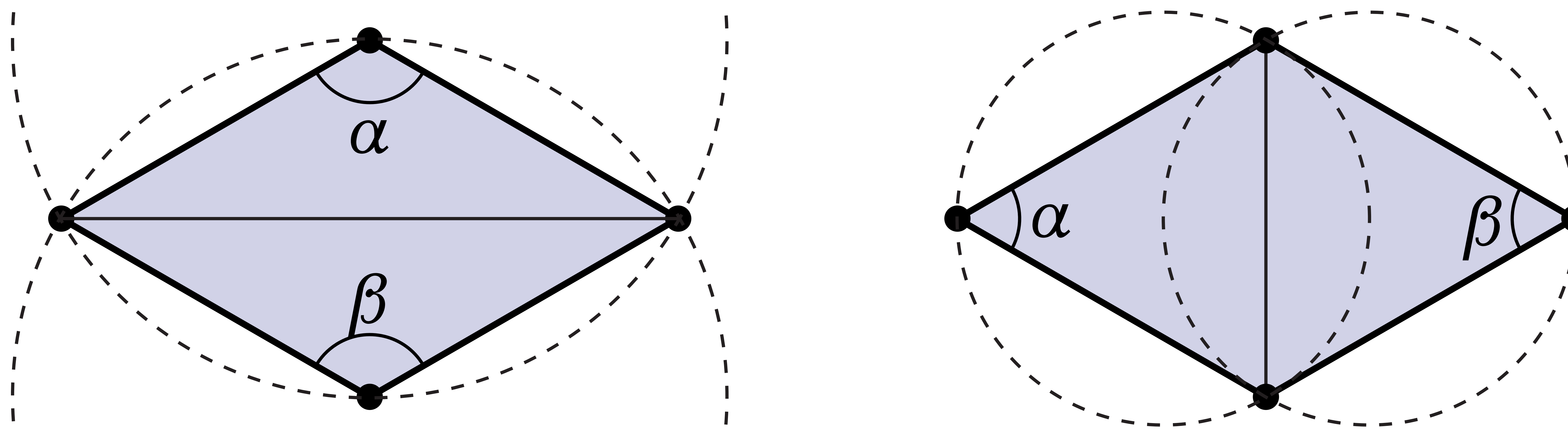
- Compute K for each triangle (distance to plane)
- Set K at each vertex to sum of Ks from incident triangles
- Set K at each edge to sum of Ks at endpoints
- Find point at each edge minimizing quadric error
- Find the cost to replace the edge with this point
- Until we reach target # of triangles:
  - collapse edge (i,j) with smallest cost to get new vertex m
  - add K_i and K_j to get quadric K_m at m
  - update cost of edges touching m
- More details in assignment writeup!



What if we're happy with the number of triangles, but want to improve quality?

How do we make a mesh "more Delaunay"?

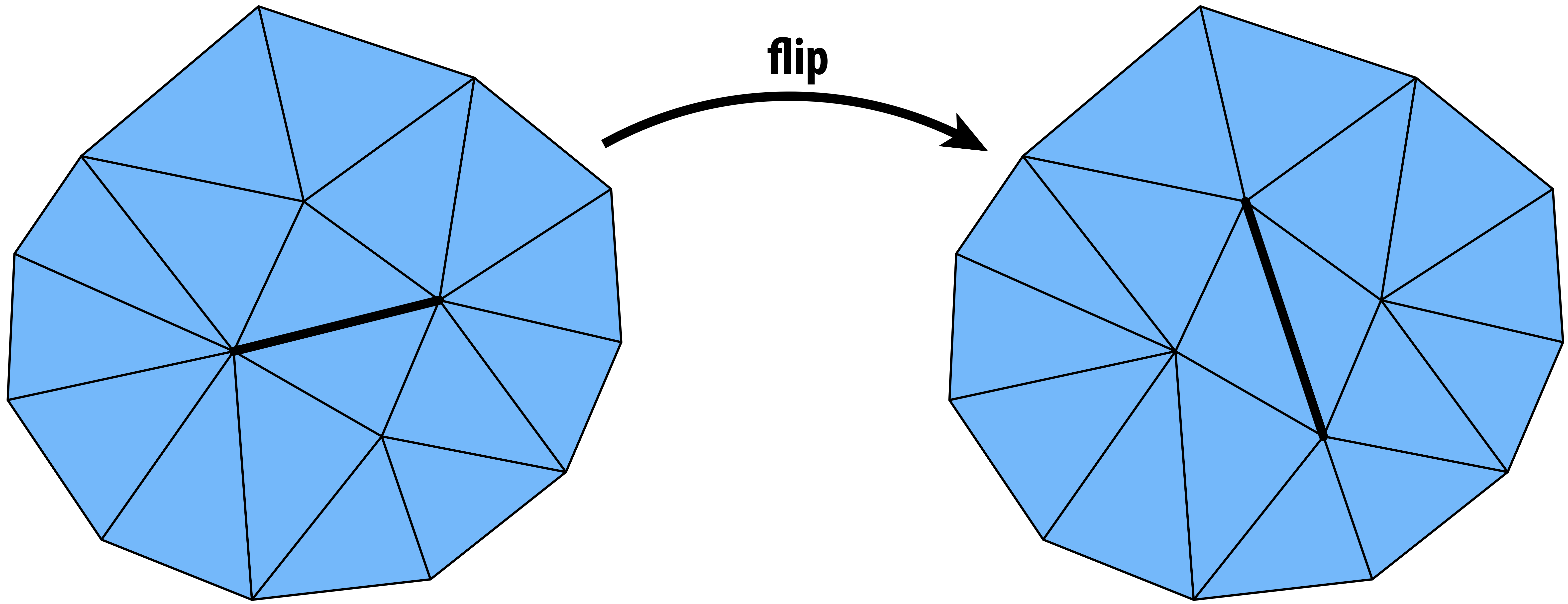
- Already have a good tool: edge flips!
- If $\alpha+\beta > \pi$, flip it!



- FACT: in 2D, flipping edges eventually yields Delaunay mesh
- Theory: worst case $O(n^2)$; no longer true for surfaces in 3D.
- Practice: simple, effective way to improve mesh quality

Alternatively: how do we improve degree?

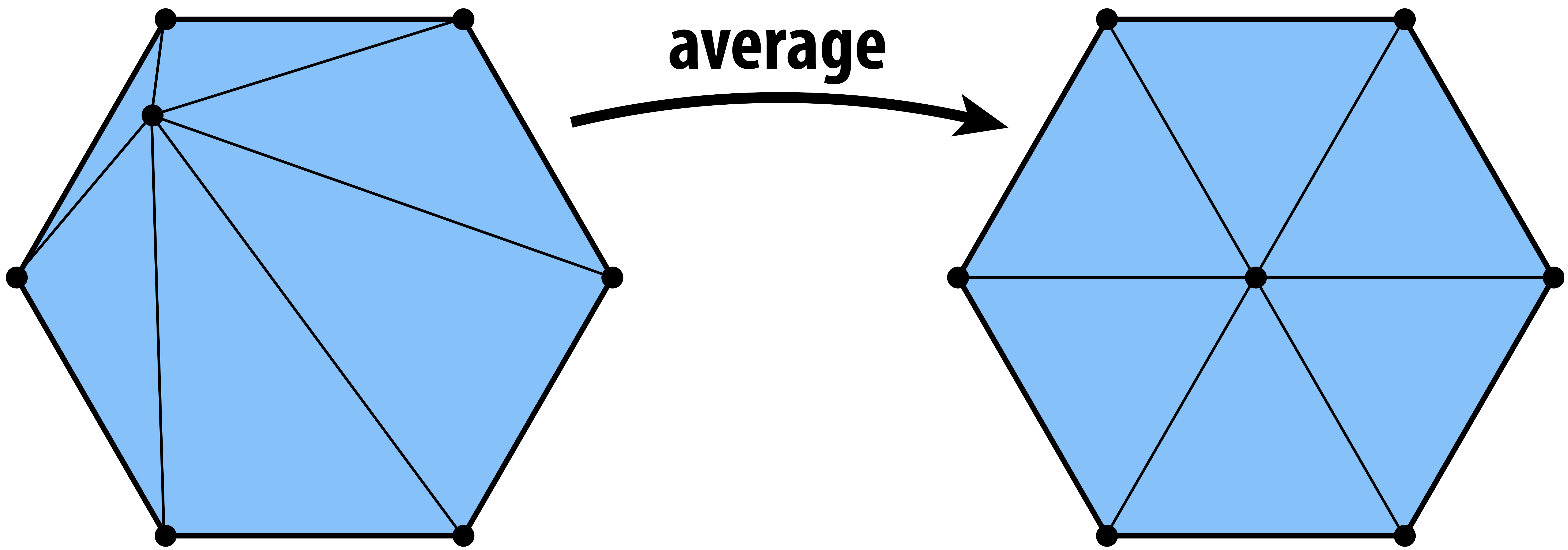
- Same tool: edge flips!
- If total deviation from degree-6 gets smaller, flip it!



- FACT: average valence of any triangle mesh is 6
- Iterative edge flipping acts like "discrete diffusion" of degree

How do we make a triangles "more round"?

- Delaunay doesn't mean triangles are "round" (angles near 60°)
- Can often improve shape by centering vertices:

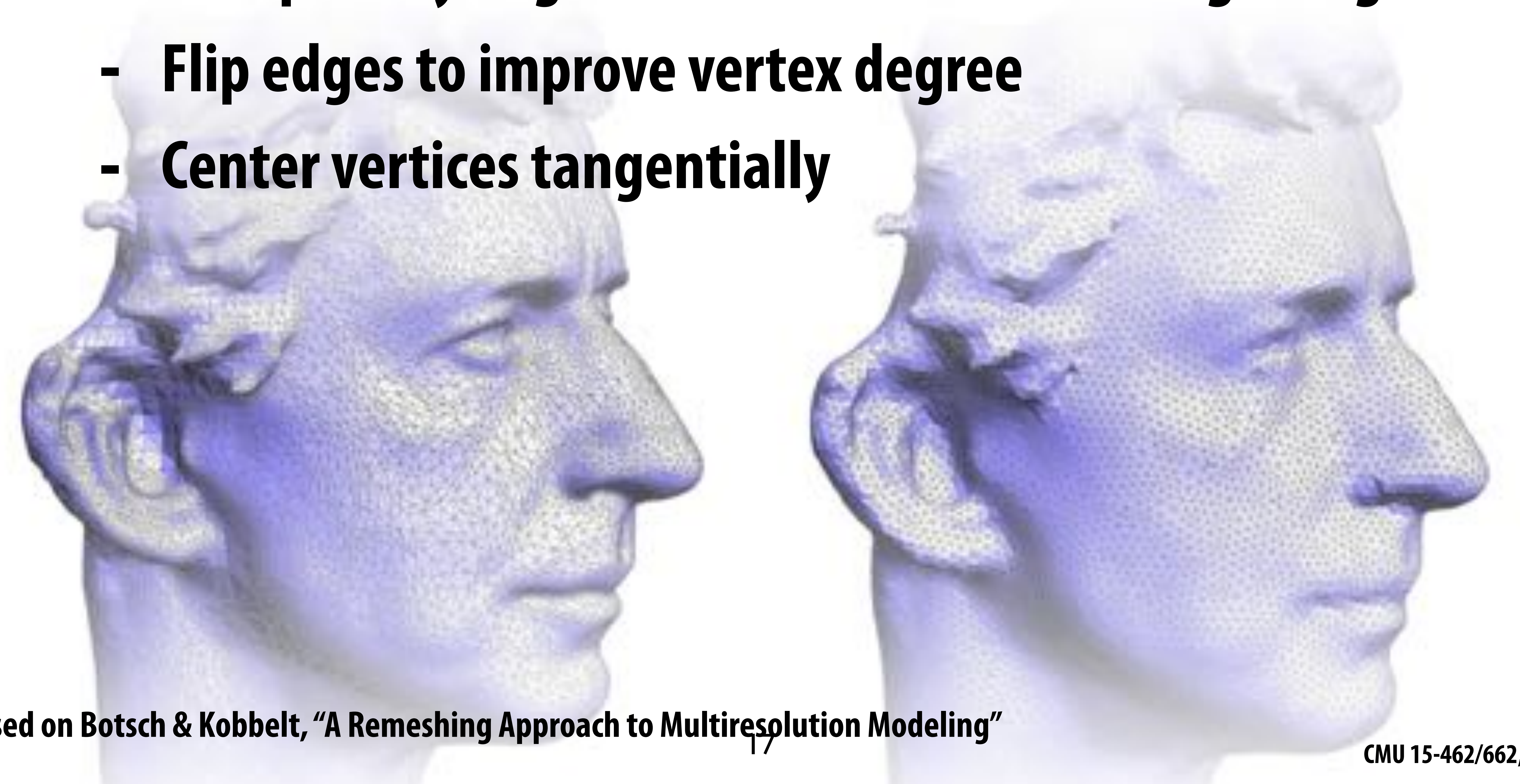


- Simple version of technique called "Laplacian smoothing".*
- On surface: move only in tangent direction
- How? Remove normal component from update vector.

*See Crane, "Digital Geometry Processing with Discrete Exterior Calculus" http://keenan.is/dgpdec

Isotropic Remeshing Algorithm*

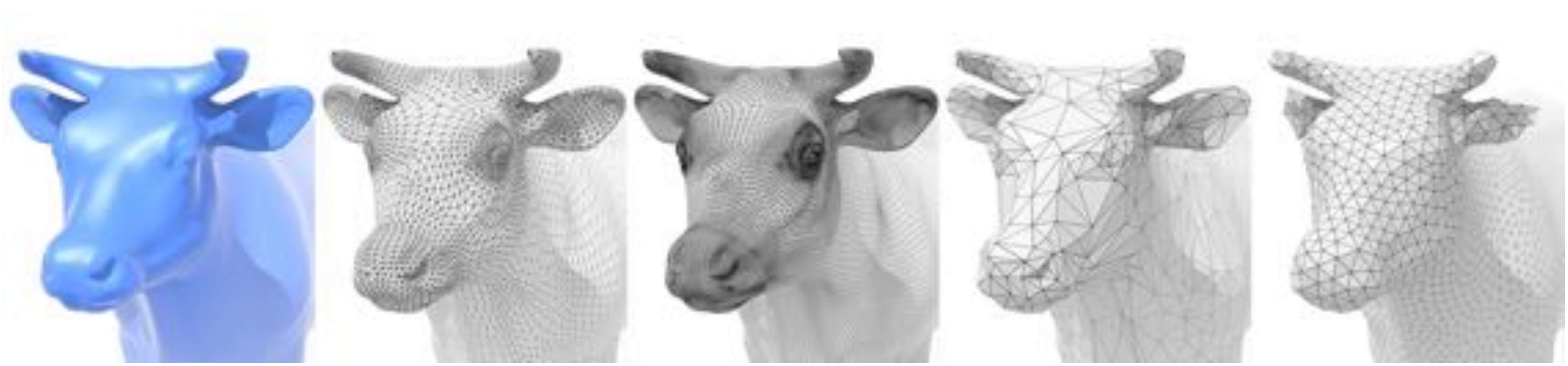
- Try to make triangles uniform shape & size
- Repeat four steps:
  - Split any edge over 4/3rds mean edge legth
  - Collapse any edge less than 4/5ths mean edge length
  - Flip edges to improve vertex degree
  - Center vertices tangentially



*Based on Botsch & Kobbelt, "A Remeshing Approach to Multiresolution Modeling"

Review: Geometry Processing

- Extend signal processing to curved shapes
  - encounter familiar issues (sampling, aliasing, etc.)
  - some new challenges (irregular sampling, no FFT, etc.)
- Focused on resampling triangle meshes
  - local: edge flip, split, collapse
  - global: subdivision, quadric error, isotropic remeshing



What you should know:

- Express distance from a plane, given a point on the plane and a normal vector
- Show how the Q matrix (the quadric error matrix) represents squared distance from a plane
- If we have a Q matrix and a point, how do we calculate cost?
- Given Q matrices encoding triangles in a mesh, how do we get the Q matrix for each vertex?
- If we collapse an edge, what is the Q matrix for the new vertex that is added in the edge collapse?
- Given a Q matrix and a proposed point, what is the cost (the quadric error)?
- How does this cost represent distance to the original surface?
- Describe some techniques for improving the quality of a mesh to make it more uniform and regular.