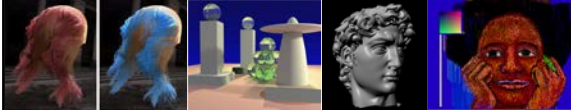


## Advanced Computer Graphics

CSE 190 [Winter 2016], Lecture 9

Ravi Ramamoorthi

<http://www.cs.ucsd.edu/~ravir>



## To Do

- Assignment 2 due Feb 19
  - Should already be well on way. This is last part
  - Contact us for difficulties etc.

## Resources

- Garland and Heckbert SIGGRAPH 97 paper
- Garland website, implementation notes (in thesis)
- Notes in this and previous lectures

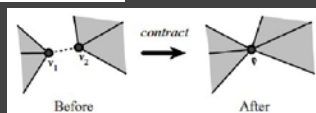
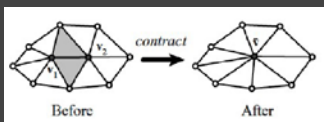
## Surface Simplification: Goals (Garland)

- Efficiency (70000 to 100 faces in 15s in 1997)
- High quality, feature preserving (primary appearance emphasized rather than topology)
- Generality, non-manifold models, collapse disjoint regions



## Simplifications

- Pair contractions in addition to edge collapses
- Previously connected regions may come together

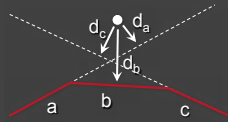


## Algorithm Outline

- Restrict process to a set of *valid pairs*:
  - $(v_i, v_j)$  is an edge, or
  - $\|v_i - v_j\| < t$ , a threshold
    - $t = 0$  restricts to edge contraction
    - $t \gg 0$  can connect distant regions or yield  $O(n^2)$  pairs
- Iteratively remove *best* pair and update valid pairs list:
  - Each vertex has a set with the pairs it belongs to:
    - $v_i \rightarrow \text{Pairs}(v_i)$
    - $(v_i, v_j) \rightarrow v \Rightarrow \text{Pairs}(v) = \text{Pairs}(v_i) \cup \text{Pairs}(v_j)$
- But how to choose *best* pair?

## Quadric Error Metrics

- Based on point-to-plane distance
- Better quality than point-to-point



## Background: Computing Planes

- Each triangle in mesh has associated plane  
 $ax + by + cz + d = 0$
- For a triangle, find its (normalized) normal using cross products

$$\vec{n} = \frac{AB \times AC}{|AB \times AC|} \quad \vec{n} \cdot \vec{v} - \vec{A} \cdot \vec{n} = 0$$

- Plane equation?

$$\vec{n} = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \quad d = -\vec{A} \cdot \vec{n}$$

## Survey

- Also examples of homework 1

## Quadric Error Metrics

- Sum of squared distances from vertex to planes:

$$\Delta_v = \sum_p \text{Dist}(\mathbf{v}, \mathbf{p})^2$$

$$\mathbf{v} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \quad \mathbf{p} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

$$\text{Dist}(\mathbf{v}, \mathbf{p}) = ax + by + cz + d = \mathbf{p}^T \mathbf{v}$$

## Quadric Error Metrics

$$\begin{aligned} \Delta &= \sum_p (\mathbf{p}^T \mathbf{v})^2 \\ &= \sum_p \mathbf{v}^T \mathbf{p} \mathbf{p}^T \mathbf{v} \\ &= \mathbf{v}^T \left( \sum_p \mathbf{p} \mathbf{p}^T \right) \mathbf{v} \\ &= \mathbf{v}^T \mathbf{Q} \mathbf{v} \end{aligned}$$

- Common mathematical trick: quadratic form = symmetric matrix Q multiplied twice by a vector

## Quadric Error Metrics

- Simply a 4x4 symmetric matrix
- Storage efficient: 10 floating point numbers per vertex
- Initially, error is 0 for all vertices

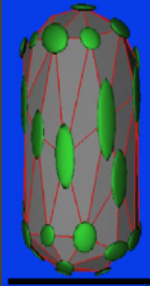
$$\mathbf{v}^T \mathbf{Q} \mathbf{v} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ q_{14} & q_{24} & q_{34} & q_{44} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

## Quadric Error Metrics

- 2<sup>nd</sup> degree polynomial in x, y and z
- Level surface ( $\mathbf{v}^T \mathbf{Q} \mathbf{v} = k$ ) is a quadric surface
  - Ellipsoid, paraboloid, hyperboloid, plane etc.

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

## But What Are These Quadrics Really Doing?



### Almost always ellipsoids

- When  $\mathbf{Q}$  is positive definite

### Characterize error at vertex

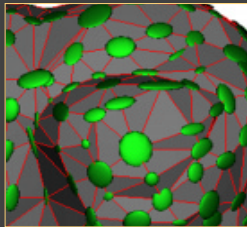
- Vertex at center of each ellipsoid
- Move it anywhere on ellipsoid with constant error

### Capture local shape of surface

- Stretch in least curved direction

## Quadric Visualization

- Ellipsoids: iso-error surfaces
- Smaller ellipsoid = greater error for a given motion
- Lower error for motion parallel to surface
- Lower error in flat regions than at corners
- Elongated in "cylindrical" regions



## Using Quadrics

### Approximate error of edge collapses

- Each vertex  $\mathbf{v}$  has associated quadric  $\mathbf{Q}$
- Error of collapsing  $\mathbf{v}_1$  and  $\mathbf{v}_2$  to  $\mathbf{v}'$  is  $\mathbf{v}'^T \mathbf{Q}_1 \mathbf{v}' + \mathbf{v}'^T \mathbf{Q}_2 \mathbf{v}'$
- Quadric for new vertex  $\mathbf{v}'$  is  $\mathbf{Q}' = \mathbf{Q}_1 + \mathbf{Q}_2$

## Using Quadrics

- Find optimal location  $\mathbf{v}'$  after collapse:

$$\mathbf{Q}' = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ q_{14} & q_{24} & q_{34} & q_{44} \end{bmatrix}$$

$$\min_{\mathbf{v}'} \mathbf{v}'^T \mathbf{Q}' \mathbf{v}': \quad \frac{\partial}{\partial x} = \frac{\partial}{\partial y} = \frac{\partial}{\partial z} = 0$$

## Using Quadrics

- Find optimal location  $\mathbf{v}'$  after collapse:

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

$$\mathbf{v}' = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

## Algorithm Outline

- Restrict process to a set of *valid pairs*:
  - $(v_i, v_j)$  is an edge, or
  - $\|v_i - v_j\| < t$ , a threshold
    - $t = 0$  restricts to edge contraction
    - $t \gg 0$  can connect distant regions or yield  $O(n^2)$  pairs
- Iteratively remove *best* pair and update valid pairs list:
  - Each vertex has a set with the pairs it belongs to:
    - $v_i \rightarrow \text{Pairs}(v_i)$
    - $(v_i, v_j) \rightarrow \bar{v} \Rightarrow \text{Pairs}(\bar{v}) = \text{Pairs}(v_i) \cup \text{Pairs}(v_j)$
- But how to choose *best* pair?

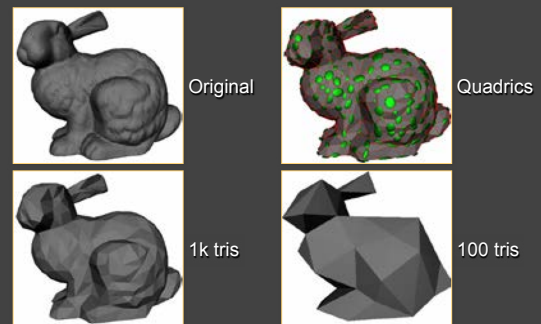
## Algorithm Summary

- Compute the Q matrices for all the initial vertices
- Select all valid pairs
- Compute the optimal contraction target  $v$  for each valid pair. The error  $v^T(Q_i + Q_j)v$  of this target vertex becomes the *cost* of contracting that pair
- Place all pairs in a heap keyed on cost with minimum cost pair on the top
- Iteratively remove the least cost pair, contract this pair, and update the costs of all valid pairs of interest

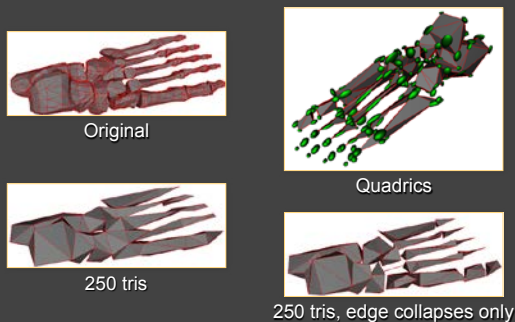
## Final Algorithm

- Compute  $Q_i$  for all vertices  $v_i$
- Determine valid pairs
- Compute optimal contraction target and associated quadric error for each pair
- Place all pairs in a heap, ordered by smallest error
- Repeat
  - Get least error pair  $(v_i, v_j)$  from heap
  - Contract pair (move edges to  $v$ , remove degenerate planes)
  - Update cost for all pairs involving  $v_i$  and  $v_j$
- Until done.

## Results



## Results



## Additional Details

- Preserving boundaries/discontinuities (weight quadrics by appropriate penalty factors)
- Preventing mesh inversion (flipping of orientation): compare normal of neighboring faces, before after
- Has been modified for many other applications
  - E.g. in silhouettes, want to make sure volume always increases, never decreases
  - Take color and texture into account (followup paper)
- See paper, other more recent works for details

---

### Implementation Tips

---

- Incremental, test, debug, simple cases
- Find good data structure for heap etc.
- May help to visualize error quadrics if possible
- Challenging, but hopefully rewarding assignment

---

### Questions?

---

- Issues or questions?
- All the material for assignment covered
- Start early, work hard