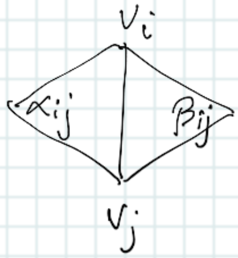
COMP 5821M Geometric Processing — Nov. 25, 2021 p.1

Loose End: Cotangent Weighting



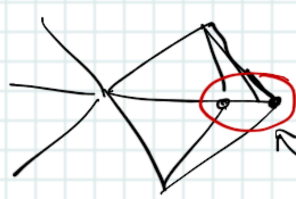

$$\Delta f(v_i) = \underbrace{\frac{1}{2A_i} \sum_{v_j \in N_1(v_i)} (\cot \alpha_{ij} + \cot \beta_{ij})(f_j - f_i)}_{\text{weighted average}}$$

- penalizes slivers (long thin triangles)
- works well in practice, better than area/angle
- but sometimes breaks down
- <u>not</u> an intrinsic property
- <u>much</u> more expensive to compute

**Simplification** suppose you have 1,000,000 triangles in a model
you can only afford 1,000

goal - reduce the poly count
- keep the shape

① vertex clustering - choose vertices that are close to each other
& combine them
- e.g. quantised - i.e. drop all but
(? 8) significant bits of x, y, z
then some vertices will have the same
position & can be combined



merge - mesh operation

two points
close but not connected pinch point → non-manifold

mostly works, but has topological problems

② resampling & repair techniques
- in practice, these techniques increase the poly count

## ③ Greedy Decimation

Ⓐ Pick an element (vertex, edge, face, &c.)

Ⓑ Get rid of it (mesh operations - preferably Eulerian)

Ⓒ Retriangulate if necessary (mesh operations)

Ⓓ Recurse - i.e. ~~rinse and~~ repeat

Ⓔ Terminate - when you hit target condition - size, quality &c.

for each element x in mesh M
  push x on priority queue

while ||M|| > target_size  (what type of element?)
  pop x off priority queue  (what is the priority?)
  delete x from mesh  (what type of mesh operation?)
  update priorities if needed

what type - vertex, edge, face $\Big\}$ people use all three

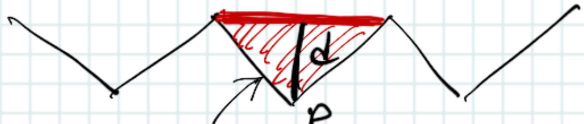
$$v \sim \Theta(e) \sim \Theta(f)$$

reducing v reduces e & f & vice versa, so any works

<u>Priority</u>: least important element
- low curvature near a vertex/edge/face
- low area in the 1-ring (or the 2-ring)
- low degree/high degree
- any combination of these
- minimum geometric change

Priorities, continued

- minimum geometric change




$\mathcal{L}$ can measure distance or the area (curve)
volume (surface)

- volume
- distance - maximum distance everywhere on a mesh
  - set of points $A, B$
  
  for each point $a \in A$, $b \in B$, find distance $\|a-b\|$
  
  use this to find closest point $b$ to a given $a$,
  
  i.e. $\min_{b \in B} \|a-b\|$
  
  then $\mathcal{L}_b(A,B) = \max_{a \in A} \min_{b \in B} \|a-b\|$ is the Hausdorff distance between $A$ & $B$
  
  → mathematicians use this one a lot
  
  → computing it is $O(v^2)$
  
  ow → doing this for all possible edits is $O(v^3)$
- conservative approximations
  - size of 1-ring
  - cotangent weights summed around a vertex
  - length of the edges removed

- standard cheap solution
  - edge collapse is easy to test Eulerian condition
  - edge length is a reasonable priority measure