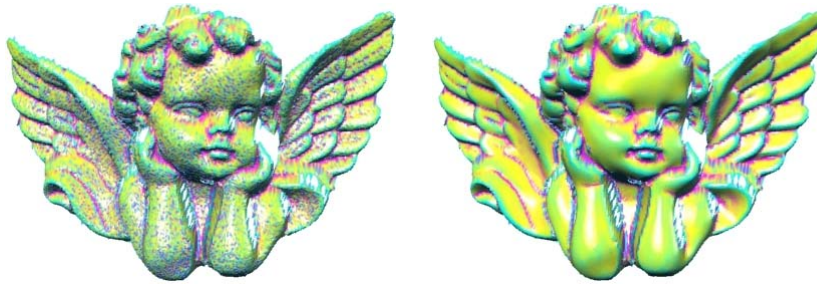


Mesh Smoothing/Denoising



Mesh Smoothing

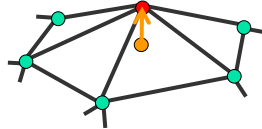
- Mesh Smoothing – moving mesh vertices on surface to reduce curvature variation
- Similar to high frequency elimination in signal processing
- Note: Can't reduce overall Gauss curvature
 - Why?
- Use to
 - Reduce noise
 - Improve mesh triangle shape





Differential coordinates

- Noise (detail) = surface – *smooth*(surface)
- Smoothing = averaging



$$\delta_i = \mathbf{v}_i - \frac{1}{d_i} \sum_{j \in N(i)} \mathbf{v}_j$$

$$\delta_i = \sum_{j \in N(i)} \frac{1}{d_i} (\mathbf{v}_i - \mathbf{v}_j)$$

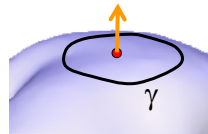
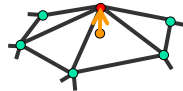


University of
British Columbia



Connection to the smooth case

- The direction of δ_i approximates the normal
- The size approximates the mean curvature



$$\delta_i = \frac{1}{d_i} \sum_{\mathbf{v} \in N(i)} (\mathbf{v}_i - \mathbf{v})$$

$$\frac{1}{\text{len}(\gamma)} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_i - \mathbf{v}) ds$$

$$\lim_{\text{len}(\gamma) \rightarrow 0} \frac{1}{\text{len}(\gamma)} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_i - \mathbf{v}) ds = H(\mathbf{v}_i) \mathbf{n}_i$$



University of
British Columbia



Laplacian Smoothing

- Equivalent to box filter in signal processing
- Let

$$v_i = v_i - \lambda \delta_i$$

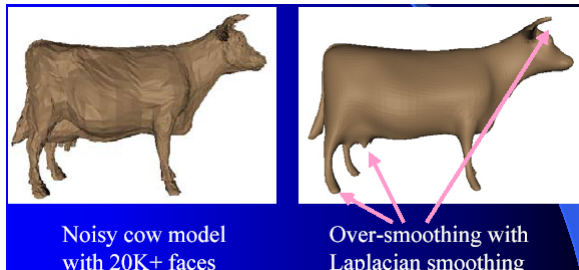
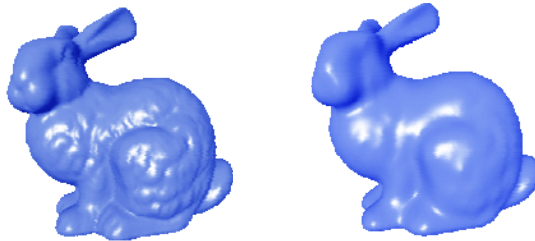
- Apply to all vertices in mesh
- Typically repeat several times
- Can describe as energy minimization
 - Energy = sum of squared edge lengths in mesh
 - Parameter $\lambda > 0$ controls convergence "speed"



University of
British Columbia



Laplacian



Noisy cow model
with 20K+ faces

Over-smoothing with
Laplacian smoothing



University of
British Columbia

Weighting schemes (reflect mesh shape)

$$\delta_i = \frac{\sum_{j \in N(i)} w_{ij} (\mathbf{v}_i - \mathbf{v}_j)}{\sum_{j \in N(i)} w_{ij}}$$

- Ignore geometry

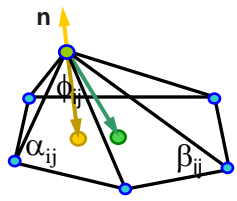
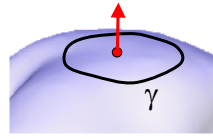
$$\delta_{\text{umbrella}} : w_{ij} = 1$$

- Integrate over circle around vertex

$$\delta_{\text{mean value}} : w_{ij} = \tan \phi_{ij}/2 + \tan \phi_{ij+1}/2$$

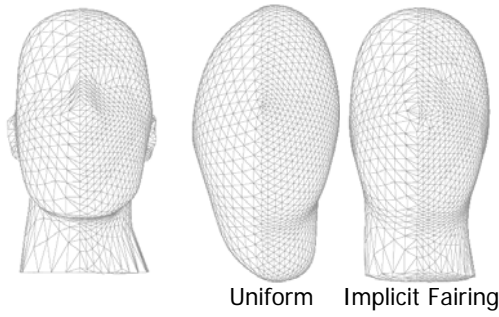
- Integrate over Voronoi region of vertex

$$\delta_{\text{cotan}} : w_{ij} = \cot \alpha_{ij} + \cot \beta_{ij}$$

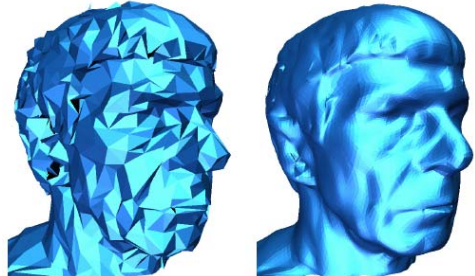


University of British Columbia

Implicit Fairing



- Implicit Fairing – combine
 - cotan weights
 - move vertices in normal direction only



University of British Columbia



Bilateral de-Noising

- Use larger neighbourhood & similarity considerations
- Specifics:
 - Move vertices in **normal** direction
 - Similarity weights inspired by image processing
- **One iteration is ENOUGH**

```

DenoisePoint(Vertex v, Normal n)
{qi} = neighborhood(v)
K = |{qi}|
sum = 0
normalizer = 0
for i := 1 to K
  t = ||v - qi||
  h = <n, v - qi>
  wc = exp(-t2/(2σc2))
  ws = exp(-h2/(2σs2))
  sum += (wc · ws) · h
  normalizer += wc · ws
end
return Vertex v̂ = v + n · (sum/normalizer)

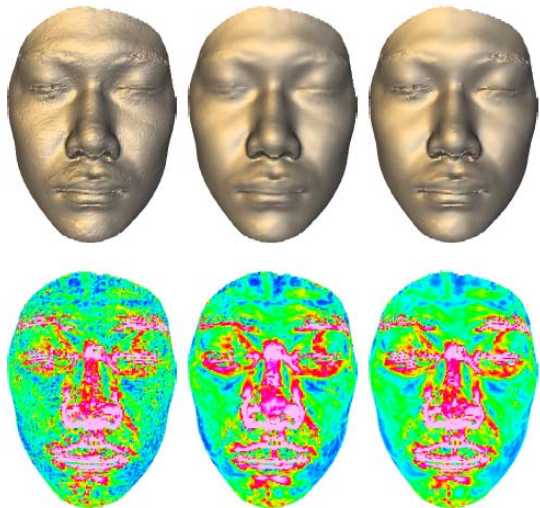
```



University of British Columbia



Smoothing



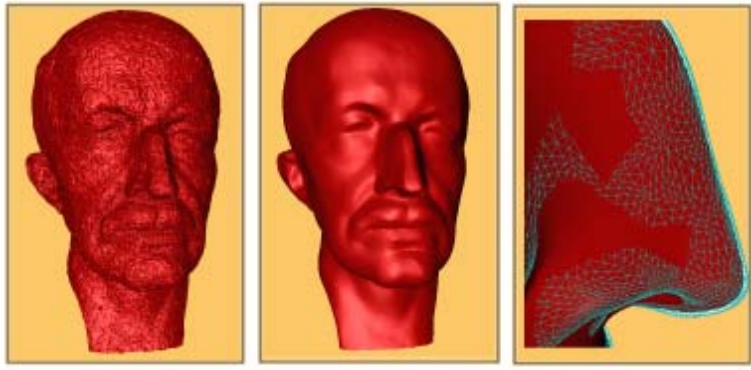
Mean Curvature

Input Implicit fairing Denoising



University of British Columbia

Shrinkage



- Special treatment – volume preserving scaling



Smoothing

- Can apply not only to positions but to any property assigned to vertices
 - Curvature, normals, physical properties

$$K_i = K_i + \lambda \frac{1}{d_i} \sum_{j \in N_i} (K_j - K_i)$$



