Alexander Belyaev (belyaev@mpi-sb.mpg.de, http://www.mpi-sb.mpg.de/∼belyaev)

Mesh Smoothing and Enhancing. Curvature Estimation.

Laplacian smoothing. Smoothing removes surface noise and improves the appearance of surfaces. A simple, yet effective technique for polyhedral surface smoothing is Laplacian smoothing.

Let us consider a triangulated surface and for any vertex P let us define the so-called umbrella-operator

$$\mathcal{U}(P) = \frac{1}{\sum_i w_i} \sum_i w_i Q_i - P$$

where summation is taken over all neighbors of P and w_i are positive weights. See Fig. 1 for the geometric idea behind the umbrella-operator.

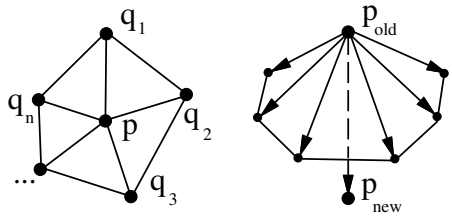


Figure 1: Umbrella-operator used for Laplacian smoothing.

The weights, can be defined, for example, as the inverse distances between P and its neighbors Q_i: $w_i = \|P - Q_i\|^{-1}$. The simplest umbrella-operator is obtained if $w_i = 1$: the umbrella-operator has the form

$$\mathcal{U}(P) = \frac{1}{n} \sum_i Q_i - P \tag{1}$$

where n is the number of neighbors.

The local update rule

$$P_{\rm new} \longleftarrow P_{\rm old} + \lambda \mathcal{U}(P_{\rm old}) \tag{2}$$

applied to every point of the triangulated surface is called *Laplacian smoothing* of the surface. Typically the factor λ is a small positive number, and the process (2) is executed repeatedly.

The Laplacian smoothing algorithm reduces the high frequency surface information and tends to flatten the surface. See Fig. 2 where Laplacian smoothing is applied to a triangulated model of a Noh mask.

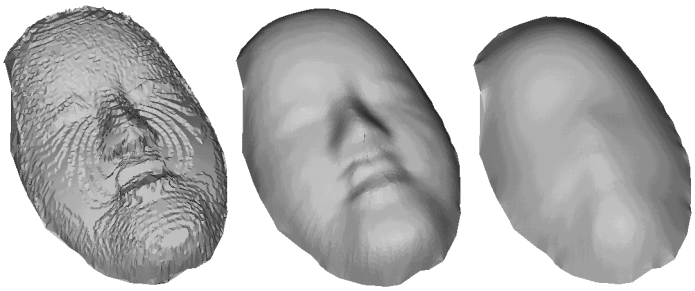


Figure 2: Left: initial triangle mesh. Middle: after 5 iterations with $\lambda = 1$. Right: after 100 iterations with $\lambda = 1$.

If λ is too small, one needs more iterations for smoothing and the smoothing process becomes time-consuming. If λ is not small enough, the smoothing process becomes unstable. See Fig. 3

Smoothing by mean curvature flow. A better mesh smoothing scheme, the so-called mean curvature flow, is obtained if the following vertex update procedure is used instead of (2)

$$P_{\rm new} \longleftarrow P_{\rm old} + \lambda H(P_{\rm old}) \mathbf{n}(P_{\rm old}), \tag{3}$$

where $H(P)$ $\mathbf{n}(P)$ are discrete approximations of the mean curvature and unit normal vector at mesh vertex P, respectively.

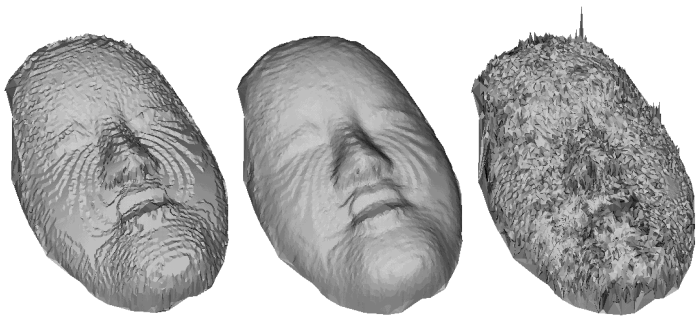


Figure 3: Left: initial triangle mesh. Middle: after 5 iterations with $\lambda = 0.3$. Right: after 5 iterations with $\lambda = 1.7$.

It turns out that (3) also becomes unstable if λ is not small enough. Let us modify (3) slightly

$$P_{\rm new} \longleftarrow P_{\rm old} + \lambda H(P_{\rm new}) \mathbf{n}(P_{\rm new}), \tag{4}$$

Mesh smoothing evolution (4) is much more stable than (3). However in order to update a mesh according to (4) we have to solve a system of linear equations.

Enhancement. Smoothing schemes can be also used for enhancement purposes. Consider a mesh $\mathcal{M}$. Let $\mathcal{M}'$ is obtained from $\mathcal{M}$ via smoothing. Denote by $P' \in \mathcal{M}'$ a vertex of the smoothed mesh corresponding to vertex $P \in \mathcal{M}$ of the original mesh. An enhanced mesh is composed by vertices $\{P' + t\,(P - P')\}$, where $t \geq 1$.

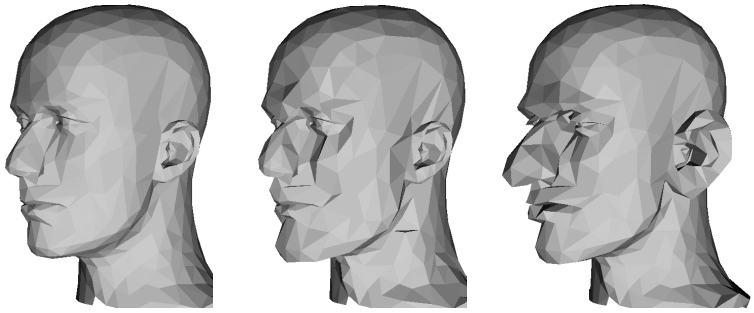


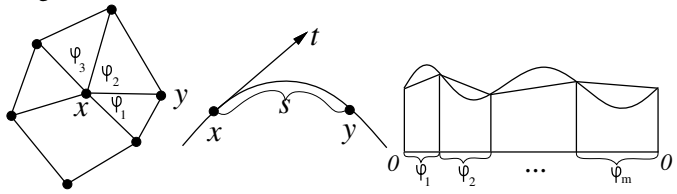
Figure 4: Left: initial triangle mesh. Middle and Right: different mesh enhancement procedures were applied.

Geometric Curvature Estimation

Integral estimation of the mean curvature. Note that

$$\frac{1}{2\pi} \int_0^{2\pi} k_n(\varphi)\, d\varphi = \frac{k_{\max} + k_{\min}}{2} = H$$

This observation leads to a method to estimate the mean curvature of a triangulated surface.



$$\mathbf{r}(s) = \mathbf{r}(0) + s\mathbf{r}'(0) + \frac{s^2}{2}\mathbf{r}''(0) + \ldots, \quad \mathbf{y} = \mathbf{x} + s\mathbf{t} + \frac{s^2}{2} k\mathbf{n} + \ldots,$$

$$\mathbf{n}(\mathbf{y} - \mathbf{x}) \approx k\frac{s^2}{2}, \quad s \approx \|\mathbf{y} - \mathbf{x}\|, \quad k \approx \frac{2\mathbf{n}(\mathbf{y} - \mathbf{x})}{\|\mathbf{y} - \mathbf{x}\|^2}$$

$$\int_0^{2\pi} k_n(\varphi)\, d\varphi \approx k_1 \left(\frac{\varphi_1 + \varphi_m}{2}\right) + k_2 \left(\frac{\varphi_2 + \varphi_3}{2}\right) + \ldots$$

Integral estimation of the Gaussian curvature. Note that

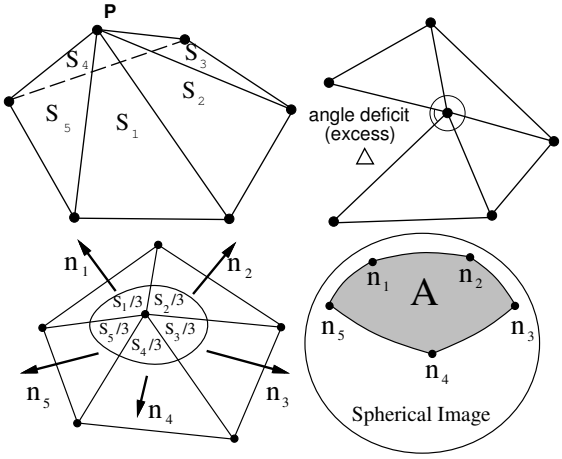
$$\frac{1}{2\pi}\int_0^{2\pi} k_n(\varphi)^2\, d\varphi = \frac{3}{2}H^2 - \frac{1}{2}K.$$

It allows us to estimate the Gaussian curvature of a triangulated surface similarly to the above estimation of the mean curvature.

Gaussian curvature estimation via Gauss mapping. The angle deficit $\Delta(\mathbf{p})$ of a vertex $\mathbf{p}$ of a polygonal surface is defined as the vertex angle deficit (excess)

$$\Delta(\mathbf{p}) = 2\pi - \theta(\mathbf{p}) = 2\pi - \sum_{i=1}^{m}\theta_i(\mathbf{p})$$

For example, the vertices of a cube each have the angle deficit $\pi/2$.



Let $\mathbf{p}$ be surrounded by triangular faces with areas $S_1, S_2, S_3, \ldots$ and unit normals $\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$, ... The spherical image of the polygonal surface is a set of points on the unit sphere (the heads of unit vectors parallel to $\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3, \ldots$). Let us connect these points by arcs of great circles to form a spherical polygon on the unit sphere. The area A of the spherical polygon is equal to the angle deficit of $\mathbf{p}$. The area of each triangular face adjacent to $\mathbf{p}$ can be portioned into three equal parts corresponding to the vertices of the face. So the total area related to $\mathbf{p}$ is $\sum S_k/3$. Thus the Gaussian curvature at $\mathbf{p}$ can be approximated by

$$K(\mathbf{p}) = 3\Delta(\mathbf{p}) \Big/ \sum S_k$$

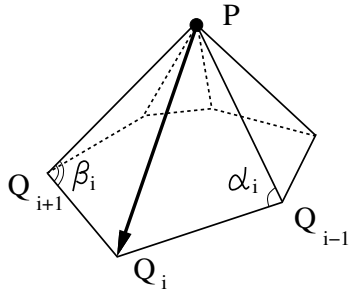
Mean curvature vector estimation via area variation. Desbrun *et al.* [1] proposed recently an accurate and robust discrete approximation of the mean curvature vector at a mesh vertex P:

$$H\mathbf{n} = -\frac{\nabla A}{2A},$$

where $A = \sum A_i$ is the sum of the areas of the triangles surrounding P. Calculations show that

$$H\mathbf{n}(P) = \frac{1}{4A}\sum_i (\cot\alpha_i + \cot\beta_i)(Q_i - P), \qquad (5)$$

where $\{Q_i\}$ are the neighbors of P, α_i and β_i are the two angles opposite to the edge Q_iP, as seen in the figure below.



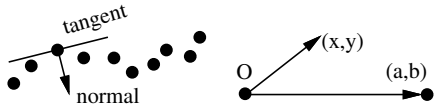
[1] M. Desburn, M. Meyer, P. Schröder, and A. H. Barr, "Implicit Fairing of Irregular Meshes using Diffusion and Curvature Flow", *SIGGRAPH'99*.

Least Squares Estimation

Least squares for normal and tangent plane estimation. Consider a set of 3D points located along a smooth surface. Our aim is to estimate the surface tangent plane and normal at each point.

First let us consider the simplest case of two 2D points. The unit normal vector (x, y) delivers minimum in

$$\min_{x^2+y^2=1} (ax + by)^2 \qquad (6)$$



To solve (6) via the method of Lagrange multipliers let us form the function

$$L(x, y, \lambda) = (ax + by)^2 - \lambda(x^2 + y^2 - 1)$$

$$\frac{\partial L}{\partial x} = 0 = \frac{\partial L}{\partial y} = 0 = \frac{\partial L}{\partial \lambda}, \quad \begin{pmatrix} a^2 & ab \\ ab & b^2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \lambda\begin{pmatrix} x \\ y \end{pmatrix}$$

Thus $(x, y) = (-b, a)/\sqrt{a^2 + b^2}$.

Now let us consider a set of 3D points $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ assumed to be on or near an unknown surface. Let $\mathbf{x} \in X$ be a point where we want to estimate the tangent plane and normal. The scatter matrix is defined as a symmetric matrix

$$M(\mathbf{x}) = \sum_{\mathbf{y}\in Nbhd(\mathbf{x})} (\mathbf{y} - \mathbf{x}) \otimes (\mathbf{y} - \mathbf{x})$$

where $\otimes$ denotes the outer product vector operation (if $\mathbf{a}$ and $\mathbf{b}$ have components a_i and b_j respectively, then the matrix $\mathbf{a} \otimes \mathbf{b}$ has a_ib_j as its ij-entry). Here $Nbhd(\mathbf{x})$ can be defined, for example, as

$$Nbhd(\mathbf{x}) = \{\mathbf{y} \in X \,:\, \|\mathbf{x} - \mathbf{y}\| \le \rho\}$$

where ρ is a given positive parameter.

Let $\lambda_1 \ge \lambda_2 \ge \lambda_3$ denote the eigenvalues of $M(\mathbf{x})$ associated with unit eigenvectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, respectively. The unit normal vector at $\mathbf{x}$, $\mathbf{n}(\mathbf{x})$, can be chosen either $\mathbf{v}_3$ or $-\mathbf{v}_3$. The eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$ corresponding to the two largest eigenvalues form a basis in the tangent plane at $\mathbf{x}$.

Least squares for curvature estimation. Once we decide the normal and two tangent directions, we use them as a local reference frame. The biquadratic fit involves finding the five coefficients of the polynomial

$$z = f(x, y) = ax^2 + bxy + cy^2 + dx + ey$$

where the coordinates (x, y) are measured along the tangent directions, and the coordinate z is measured along the normal direction. If there are at least four neighboring points, say $n - 1$ points, the least-square estimate of the five coefficients can be done:

$$F(a, b, c, d, e) \longrightarrow \min$$

where

$$F(a, b, c, d, e) = \sum \left(ax_k^2 + bx_ky_k + cy_k^2 + dx_k + ey_k - z_k\right)^2$$

In matrix form, the coefficient vector $\vec{c} = (a, b, c, d, e)^T$ which minimize the fit error is

$$A^T A\vec{c} = A^T\vec{Z}$$

where A is a $5 \times n$ matrix with kth row being $[x_k^2, x_ky_k, y_k^2, x_k, y_k]$, $\vec{Z}$ is a column vector of the z_k values, and the superscript T stands for transposition. Once the coefficients a, b, c, d, e are found, curvature computations are straightforward.