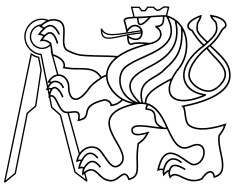
CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

REPRINT

Fairing of Discrete Surfaces with Boundary that Preserves Size and Qualitative Shape

Jana Kostlivá
Radim Šára
Martina Matýsková

{kostliva,sara}@cmp.felk.cvut.cz

J. Kostlivá, R. Šára, M. Matýsková: Fairing of Discrete Surfaces with Boundary that Preserves Size and Qualitative Shape. In: Preceedings of the ISVC 2008. Volume 5358 of LNCS, Springer-Verlag, p. 107-118, USA, 2008.

Available at
ftp://cmp.felk.cvut.cz/pub/cmp/articles/kostliva/Kostliva-ISVC08.pdf

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Fairing of Discrete Surfaces with Boundary that Preserves Size and Qualitative Shape

Jana Kostlivá, Radim Šára, and Martina Matýsková

Center for Machine Perception, CTU in Prague, Czech Republic
{kostliva,sara}@cmp.felk.cvut.cz, http://cmp.felk.cvut.cz

Abstract. In this paper, we propose a new algorithm for fairing discrete surfaces resulting from stereo-based 3D reconstruction task. Such results are typically too dense, uneven and noisy, which is inconvenient for further processing. Our approach jointly optimises mesh smoothness and regularity. The definition is given on a discrete surface and the solution is found by discrete diffusion of a scalar function. Experiments on synthetic and real data demonstrate that the proposed approach is robust, stable, preserves qualitative shape and is applicable to even moderate-size real surfaces with boundary (0.8M vertices and 1.7M triangles).

1 Introduction

In recent years, 3D scene reconstruction from stereo became a feasible problem in computer vision [1–3]. Its result is typically a triangulated mesh of the scene surface, which however is often too dense, irregular and corrupted by noise (cf. Fig. 1). Mesh fairing is a general problem that occurs in surface reconstruction, mesh interpolation or blending.

We will represent discrete surface by a triangulated mesh, a *complex*, which is a union of 0-simplices (vertices), 1-simplices (edges), and 2-simplices (triangles). Vertex w is a *direct neighbour* of vertex v if there exists an edge between them. We assume, each edge is incident with at most two triangles. If it is incident with only a single triangle, we call it a *boundary edge*.

Typically, fairing involves the following tasks: (1) *mesh simplification*, i.e. redundant triangle removal, (2) *mesh smoothing*, i.e. (random or discretisation) noise suppression, and (3) *mesh regularity improvement*. In this paper, we address (2) and (3), (1) has been studied e.g. in [4–7]. Many approaches focused on (2). They are algorithms minimising surface energy, which is based on surface curvature. Curvature is defined only on continuous surface, however, and thus some approximation is required for a discrete surface. In one group of approaches the energy is defined on a continuous surface [6, 8, 9, 7]. The energy [6, 8] or its Laplacian [9, 7] is then discretised. The second group of approaches define curvature approximation on a discrete surface directly [9, 5, 10, 11]. The last group of approaches is based on different principles, such as image processing [12, 13] or position prediction [14]. Improving mesh regularity (3) has not been investigated so deeply as surface smoothing, since most of approaches assume approximately uniform meshes. It could be based on vertex re-positioning [15] or on changing mesh topology [4–7], which belongs to mesh simplification.

We propose an alternative approach which belongs to the second group. Our curvature definition is simpler, which brings computational simplicity over [9]. We formulate

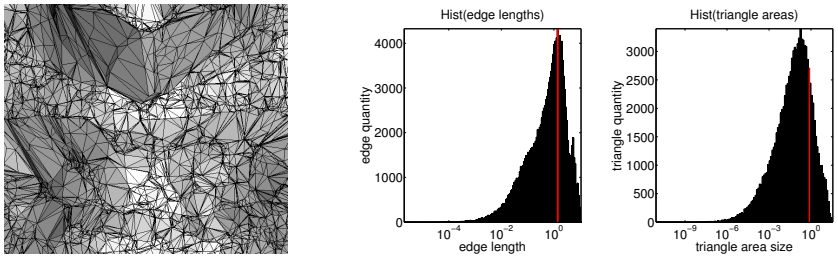


Fig. 1. Results of 3D scene reconstruction task. Histograms of edge lengths and triangle areas (note x-logarithmic scale) demonstrate a huge disproportion in these measures, the red lines correspond to the size of average edge length E_{avg} and area of equilateral triangle with this edge. Hence, the mesh is full of very small area triangles which used to be thin and long

fairing as surface optimisation by two criteria: *curvature consistency*, and *mesh regularity*. The curvature consistency term fairs locally high curvature and the irregularity term fairs local variations in edge lengths. Since these criteria affect each other, a joint optimisation is required. We say that the surface is ideally faired, if the curvature in each vertex is equal to average curvature of its neighbours and local irregularity vanishes. Our method tends to preserve qualitative shape (convex/concave/elliptic/hyperbolic). To eliminate surface shrinking during fairing, we employ scalar curvature diffusion (not curvature flow diffusion).

The main contributions of this paper are: (1) Diffusion of a scalar function (unlike of a curvature flow), which does not result in surface contraction and under certain conditions it provably preserves qualitative shape [16]. (2) Fairing formulation as a joint maximisation of curvature consistency and mesh regularity. (3) Solving the problem with non-monotonic behaviour of our curvature. Fairing surfaces with boundary involves both curve and surface fairing. Hence, formulation of fairing curves is in Sec. 2, of closed surfaces in Sec. 3 and finally by combining these is hinted in Sec. 4. In Sec. 5, we demonstrate proposed approach performance and discuss its properties, Sec. 6 concludes the paper.

2 Curve Fairing

Let C be C^2 continuous closed curve minimising $J = \int \left(\frac{\partial H}{\partial s}\right)^2 ds$, where $H = H(s,t)$ is the curvature, in which s is the curve parametrisation and t represents time. A necessary condition for the existence of an extreme of J gives Laplace's equation $\Delta H = \frac{\partial^2 H}{\partial s^2} = 0$. Consequently, function H minimising J is a harmonic function, and it can be found as a solution of diffusion equation $\frac{\partial H}{\partial t} = \lambda \frac{\partial^2 H}{\partial s^2}$, which fulfils $\lim_{t\to\infty} \frac{\partial H}{\partial t} = 0$. The λ is a diffusion coefficient, $0 < \lambda < 1$.

Since H is a harmonic function, by discretisation of the Gauss's Harmonic Function Theorem we derive $\Delta H = \bar{H}_i - H_i$, where H_i is (scalar) curvature in vertex i and $\bar{H}_i = \frac{1}{2}\left(H_{i-1} + H_{i+1}\right)$ is the mean of curvatures of direct neighbours. We discretised $\frac{\partial H}{\partial t}$ as $H_i^{k+1} - H_i^k$. Together with $\Delta H = 0$ this gives a formula for iterative curvature

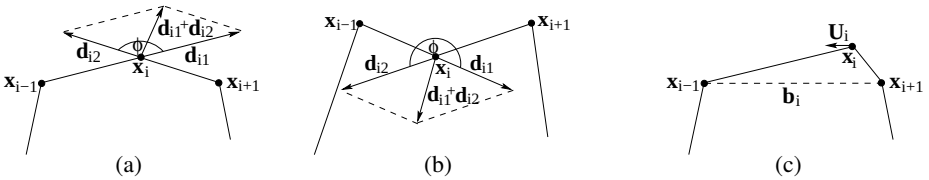


Fig. 2. The direction of curvature flow (a, b) and irregularity flow (c) at vertex $\mathbf{x}_i$

diffusion at vertex i:

$$H_i^{k+1} = (1-\lambda)H_i^k + \lambda \bar{H}_i^k, 0 < \lambda < 1. \tag{1}$$

where k represents the iteration step number.

2.1 Curvature Definition

Let $C = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}, \mathbf{x}_j = [x, y, z]$, be a given closed discrete curve represented by a vertex sequence (i.e. $\mathbf{x}_0 = \mathbf{x}_n$). Without loss of generality we assume that the curve is in a three-dimensional space. Let $\mathbf{x}_{i-1}, \mathbf{x}_{i+1}$ be direct neighbours of $\mathbf{x}_i$. The curvature flow $\mathbf{H}_i = [h_{i_x}, h_{i_y}, h_{i_z}]$ of curve C at vertex $\mathbf{x}_i$ is defined as

$$\mathbf{H}_i = \frac{\nabla\left(d_{i1}^2 + d_{i2}^2\right)}{d_{i1}^2 + d_{i2}^2}, \qquad \text{where} \tag{2}$$

$$d_{i1} = \|\mathbf{d}_{i1}\| = \|\mathbf{x}_i - \mathbf{x}_{i-1}\|, \qquad d_{i2} = \|\mathbf{d}_{i2}\| = \|\mathbf{x}_i - \mathbf{x}_{i+1}\|, \tag{3}$$

and $\nabla\left(d_{i1}^2 + d_{i2}^2\right)$ represents formal gradient with respect to the three coordinates of $\mathbf{x}_i$. By substituting (3) to (2) we express curvature flow at vertex $\mathbf{x}_i$ as:

$$\mathbf{H}_i = 2 \cdot \frac{\mathbf{d}_{i1} + \mathbf{d}_{i2}}{d_{i1}^2 + d_{i2}^2}. \tag{4}$$

The direction of curvature flow equals to the direction of vector $\mathbf{d}_{i1} + \mathbf{d}_{i2}$. Fig. 2 shows the geometric interpretation of $\mathbf{H}_i$. We denote $H_i = \|\mathbf{H}_i\|$ (2-norm). In Eq. (1), we use only curvature H_i instead of curvature flow. If we used the flow, curves would contract during the fairing process.

The proposed curvature has the following properties: (1) $\mathbf{H}_i = \mathbf{0}$ iff $\mathbf{d}_{i1} = -\mathbf{d}_{i2}$; (2) H_i is invariant to curve rotation and translation; (3) in a regular polygon, $\mathbf{H}_i$ is oriented from its centre and $H_i = \frac{1}{r}$, where r is the radius of a circumscribed circle; (4) H_i is not a monotonic function of the vertex angle ϕ (see Fig. 2(a,b)). Fig. 3(a) shows the plot of H_i vs ϕ when $d_{i1} = d_{i2}$ and the vertex $\mathbf{x}_i$ is moved in direction $\mathbf{d}_{i1} + \mathbf{d}_{i2}$; (5) The curvature does not vanish when $\mathbf{x}_i$ lies on $\mathbf{b}_i = \mathbf{x}_{i+1} - \mathbf{x}_{i-1}$ unless $\mathbf{x}_i = \frac{\mathbf{x}_{i-1}+\mathbf{x}_{i+1}}{2}$ (cf. Fig. 3(b)). Thus, diffusion of H_i also partially controls mesh regularity. On the other hand, we observed if $\mathbf{x}_i$ is slightly off the line $\mathbf{b}_i$, the diffusion tends to pull $\mathbf{x}_i$ away from the line. Hence, additional term controlling regularity is required; (6) the curvature diffusion preserves inflection points (Fig. 3(c)), hence the qualitative curve shape: For symmetric inflection points (solid), it holds $H_i^k = 0$. Since $\bar{H}_i^k = 0$ we have $H_i^{k+1} = 0$. If $\mathbf{x}_i$ is slightly deflected (dashed), $\bar{H}_i^k < 0$, since $H_{i-1}^k > H_{i+1}^k$ and thus $H_i^{k+1} < H_i^k$, until $H_i^k = 0$.

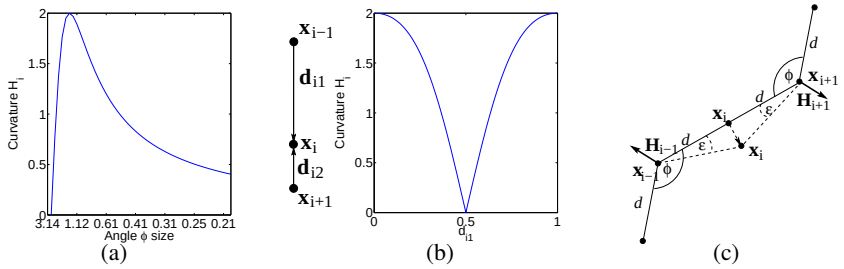


Fig. 3. Curvature properties: (a) Curvature H_i in dependence on the size of angle ϕ, when $d_{i1} = d_{i2}$. (b) Curvature in dependence on vertex $\mathbf{x}_i$ position, when $\mathbf{d}_{i1}$ and $\mathbf{d}_{i2}$ are colinear, $d_{i1}+d_{i2} = 1$, and $\mathbf{x}_i$ is moved from $\mathbf{x}_{i-1}$ towards $\mathbf{x}_{i+1}$. (c) Inflection point is preserved

2.2 Irregularity Definition

The second fairing criterion is irregularity, which is treated similarly to curvature. Curve definition as well as other variables are the same as in Sec. 2.1.

The irregularity flow $\mathbf{U}_i = [u_{i_x}, u_{i_y}, u_{i_z}]$ of curve C at vertex $\mathbf{x}_i$ is

$$\mathbf{U}_i = -\frac{1}{4}\mathbf{P}_i \left(\nabla \left(d_{i1}^2 + d_{i2}^2\right)\right), \quad \mathbf{P}_i = \frac{\mathbf{b}_i \mathbf{b}_i^T}{\|\mathbf{b}_i\|^2}, \quad \mathbf{b}_i = \mathbf{x}_{i+1} - \mathbf{x}_{i-1}. \tag{5}$$

Hence, the irregularity flow becomes:

$$\mathbf{U}_i = -\frac{1}{2}\,\mathbf{P}_i \left(\mathbf{d}_{i1} + \mathbf{d}_{i2}\right). \tag{6}$$

Irregularity flow direction is collinear with $\mathbf{b}_i$ (Fig. 2(c)), again we denote $U_i = \|\mathbf{U}_i\|$, which captures distance difference between vertex $\mathbf{x}_i$ and its direct neighbours. We setup discrete diffusion with equilibrium at zero:

$$U_i^{k+1} = (1 - \lambda_u) U_i^k, \; 0 < \lambda_u < 1. \tag{7}$$

The proposed irregularity has the following properties: (1) $\mathbf{U}_i = \mathbf{0}$ iff $d_{i1} = d_{i2}$; (2) U_i is invariant to curve rotation and translation; (3) $\mathbf{U}_i$ points directly to equilibrium state $\mathbf{x}_i^{eq}$, and $U_i = \frac{\|\mathbf{x}_i - \mathbf{x}_i^{eq}\|}{2}$; (4) U_i is a monotonic function of vertex position on the line represented by $\mathbf{U}_i$.

2.3 Algorithm

The goal of the algorithm is to iteratively move curve vertices so that both criteria, curvature and irregularity, are in equilibrium. The algorithm works with the criteria independently. Fairing under curvature and irregularity are swapped after a given number of iterations. In each iteration, the algorithm visits all curve vertices in turn. For each vertex, new values, irrespective to the used criterion, are computed. The new curve is described *only* by new scalar values. The new positions of vertices, which correspond to these values, have to be reconstructed. Due to a non-monotonic behaviour of H_i illustrated in Fig. 3 there may exist up to two solutions for a given curvature H_i^{k+1}. We accept the one which corresponds to the ascending interval, details are found in [16].

The algorithm has two termination mechanisms: The first one is defined as the minimal average vertex position move θ_{mov}. The second criterion is the maximal number of iterations k_{max}. Both these thresholds are defined by the user. The algorithm flow is given in Alg. 1. Note that the curve need not be oriented for Alg. 1 to work correctly. Local orientation is facilitated by signs s_1, s_2 in Step 3. A detailed algorithm description together with equation derivation can be found in [16].

Algorithm 1 Curve Fairing

Set $k = 1$.

1: Set $i = 1$. If the current fairing criterion is irregularity, go to step 6.
2: Evaluate $\mathbf{H}_i^k = 2 \cdot \frac{\mathbf{d}_{i1}^k + \mathbf{d}_{i2}^k}{(d_{i1}^k)^2 + (d_{i2}^k)^2}$, $H_i^k = \|\mathbf{H}_i^k\|$.
3: Compute $H_i^{k+1} = (1-\lambda)H_i^k + \lambda \bar{H}_i^k$, where $\bar{H}_i^k = \frac{1}{2}\left(s_i^k H_{i-1}^k + s_2^k H_{i+1}^k\right)$ and s_1^k, s_2^k are signs with which the curvature vectors $\mathbf{H}_{i-1}^k, \mathbf{H}_{i+1}^k$ are transfered to $\mathbf{x}_i^k$. They are positive, if the curvatures are identically oriented with $\mathbf{H}_i^k$, and negative otherwise.
4: Update $\mathbf{x}_i^{k+1} \leftarrow \mathbf{x}_i^k + \alpha_i^k(\mathbf{d}_{i1}^k + \mathbf{d}_{i2}^k)$, where α_i^k is the solution of quadratic equation, $P = \frac{(\mathbf{d}_{i1}^k)^T \mathbf{d}_{i2}^k}{(d_{i1}^k)^2 + (d_{i2}^k)^2}, Q = \frac{d_{i1}^k}{d_{i2}^k} + \frac{d_{i2}^k}{d_{i1}^k}, \cos\phi = \frac{(\mathbf{d}_{i1}^k)^T \mathbf{d}_{i2}^k}{d_{i1}^k \cdot d_{i2}^k}$:
$(\alpha_i^k)^2(2H_i^{k+1} + 4H_i^{k+1}P) + \alpha_i^k(2H_i^{k+1} + 4H_i^{k+1}P - 2H_i^k) + H_i^{k+1} - H_i^k = 0.$
Accepted $\alpha_i^k \in \left[-\frac{1}{2} - \frac{1}{2}\sqrt{\frac{Q-2\cos\phi}{Q+2\cos\phi}}, -\frac{1}{2} + \frac{1}{2}\sqrt{\frac{Q-2\cos\phi}{Q+2\cos\phi}}\right]$.
5: Set $i = i + 1$. If $i \leq n$, go to step 2. Otherwise, go to step 10.
6: Evaluate $\mathbf{U}_i^k = \frac{-\mathbf{P}_i(\mathbf{d}_{i1}^k + \mathbf{d}_{i2}^k)}{2}$, $U_i^k = \|\mathbf{U}_i^k\|$.
7: Compute $U_i^{k+1} = (1-\lambda_u)U_i^k$.
8: Update $\mathbf{x}_i^{k+1} \leftarrow \mathbf{x}_i^k + \lambda_u \mathbf{U}_i^k$.
9: Set $i = i + 1$. If $i \leq n$, go to step 6.
10: If average vertex move is smaller than a threshold θ_{mov} or $k > k_{max}$, terminate.
11: Set $k = k + 1$, go to step 1.

3 Closed Surface Fairing

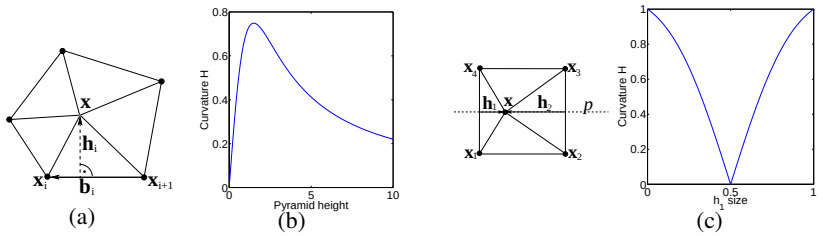
In this section, we formulate the task for closed discrete surfaces. Discretisation is performed in the same way as for curves (Sec. 2), only in higher dimension. The definition for curvature diffusion at current vertex is:

$$H^{k+1} = (1-\lambda)H^k + \lambda \bar{H}^k, 0 < \lambda < 1, \tag{8}$$

k represents iteration, and $\bar{H}{=}\frac{1}{n}\sum_{i=1}^{n} H_i$ is the curvature mean of direct neighbours.

3.1 Curvature Definition

Let S be the given discrete closed surface represented by a triangular mesh, $V = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}, \mathbf{x}_j = [x, y, z]$ be a set of its vertices. The current vertex is labelled $\mathbf{x}$, its direct neighbours $N = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, $\mathbf{x}_{n+1} = \mathbf{x}_1$, are indexed by i.



(a) (b) (c)

Fig. 4. Curvature definition: (a) Neighbourhood of the vertex $\mathbf{x}$. (b) Curvature with dependence on the height of a pyramid with regular 6-polygon base, $b_i = 1$. (c) Curvature with dependence on vertex $\mathbf{x}$ position, all vertices are in the same plane. For clearness, the vertex $\mathbf{x}$ was moved only along the line p, again $b_i = 1$.

The curvature flow $\mathbf{H} = [h_x, h_y, h_z]$ of the surface S at vertex $\mathbf{x}$ is defined as:

$$\mathbf{H} = \frac{3}{2} \cdot \frac{\nabla \sum_{i=1}^{n} A_i^2}{\sum_{i=1}^{n} A_i^2}, \tag{9}$$

where (cf. Fig. 4(a)) $A_i = \frac{1}{2}\|\mathbf{h}_i\|\|\mathbf{b}_i\| = \frac{1}{2} h_i b_i$ is the area of a triangle formed by vertices $\mathbf{x}$, $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$, $\mathbf{h}_i$ is the height at vertex $\mathbf{x}$, $\mathbf{b}_i = \mathbf{x}_i - \mathbf{x}_{i+1}$ is the side opposite $\mathbf{x}$, and $\nabla \sum_{i=1}^{n} A_i^2 = \frac{1}{4} \sum_{i=1}^{n} b_i^2 \nabla h_i^2$.

Let us define $\mathbf{P}_i = \mathbf{E} - \frac{\mathbf{b}_i \mathbf{b}_i^T}{b_i^2}$, $\mathbf{h}_i = \mathbf{P}_i(\mathbf{x} - \mathbf{x}_i)$, $\mathbf{h}_i^2 = (\mathbf{x} - \mathbf{x}_i)^T \mathbf{P}_i (\mathbf{x} - \mathbf{x}_i)$, $\mathbf{B}_i = b_i^2 \mathbf{P}_i = b_i^2 \mathbf{E} - \mathbf{b}_i \mathbf{b}_i^T$, where $\mathbf{P}_i$ is a projection matrix and thus $\mathbf{P}_i^2 = \mathbf{P}_i$ and $\mathbf{P}_i^T = \mathbf{P}_i$. By substituting to (9) we get curvature flow at vertex $\mathbf{x}$:

$$\mathbf{H} = \frac{3}{2} \cdot \frac{\sum_{i=1}^{n} \mathbf{B}_i (\mathbf{x} - \mathbf{x}_i)}{\sum_{i=1}^{n} (\mathbf{x} - \mathbf{x}_i)^T \mathbf{B}_i (\mathbf{x} - \mathbf{x}_i)}. \tag{10}$$

The direction of curvature flow in vertex $\mathbf{x}$ equals to the direction of vector $\sum_{i=1}^{n} \mathbf{h}_i$ and $H = \|\mathbf{H}\|$. As in curves, in Eq. (8) we diffuse only curvature H instead of curvature flow.

The curvature definition has the following properties: (1) H is invariant to surface rotation and translation; (2) in a regular polyhedron, $\mathbf{H}$ is oriented from its centre and $H = \frac{1}{r}$, where r is the radius of a circumscribed sphere; (3) H is not a monotonic function of the height of the point $\mathbf{x}$ neighbourhood. This is illustrated in Fig. 4(b) showing the plot of H vs height of pyramid with regular 6-polygon base; (4) the curvature does not necessarily vanish when $\mathbf{x}$ and its direct neighbours lie on the same plane (cf. Fig. 4(c)); (5) curvature definition preserves inflection points. This property is demonstrated in experiments in Figs. 5, 6.

3.2 Irregularity Definition

The second fairing criterion is irregularity, treated similarly to curvature (as for curves). Surface definition as well as other variables are the same as in Sec. 3.1.

The irregularity flow $\mathbf{U} = [u_x, u_y, u_z]$ of the surface S at vertex $\mathbf{x}$ is

$$\mathbf{U} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{U}_i, \;\; \mathbf{U}_i = -\frac{1}{4}\mathbf{P}_{\mathbf{d}i}\Big(\nabla\left(\|\mathbf{x}-\mathbf{x}_{i-1}\|^2 + \|\mathbf{x}-\mathbf{x}_{i+1}\|^2\right)\Big), \qquad (11)$$

where n is the number of direct neighbours of vertex $\mathbf{x}$ and $\mathbf{P}_{\mathbf{d}i} = \frac{\mathbf{d}_i\mathbf{d}_i^T}{\|\mathbf{d}_i\|^2}, \mathbf{d}_i = \mathbf{x}_{i+1} - \mathbf{x}_{i-1}$. Note that $\mathbf{x}_0 = \mathbf{x}_n$ and $\mathbf{x}_{n+1} = \mathbf{x}_1$. $\mathbf{P}_{\mathbf{d}i}$ is a projection matrix into direction given by the vector $\mathbf{d}_i$. Consequently,

$$\mathbf{U} = -\frac{1}{2n}\sum_{i=1}^{n}\mathbf{P}_{\mathbf{d}i}\left((\mathbf{x}-\mathbf{x}_{i-1}) + (\mathbf{x}-\mathbf{x}_{i+1})\right). \qquad (12)$$

The magnitude of irregularity, $U = \|\mathbf{U}\|$, defines the distance of the vertex $\mathbf{x}$ from imaginary centre of its direct neighbours. We define discrete diffusion as for curves:

$$U^{k+1} = (1-\lambda_u)\,U^k, \; 0 < \lambda < 1. \qquad (13)$$

The properties of the irregularity definition are as follows: (1) U is invariant to surface rotation and translation; (2) $\mathbf{U}$ points directly to equilibrium state; (3) irregularity is a monotonic function of vertex position on the line given by $\mathbf{U}$.

3.3 Algorithm

The principle of fairing surfaces is the same as for curves (Sec. 2.3). The only small difference is in the sign computation: Explicit global surface orientation is not possible if we want to fair non-orientable surfaces (e.g. Möbius strip, Klein bottle). Hence, we will orient the surface only locally, which is always possible and guarantees correct determination of signs s_i in Step 3 of Alg. 2. The sign will be positive, if the curvature vectors are on the same side of the surface (given by the orientation).

The following substitution will be used to simplify algorithm description: $\mathbf{B} = \sum_{i=1}^{n}\mathbf{B}_i, \mathbf{b} = \sum_{i=1}^{n}\mathbf{B}_i\mathbf{x}_i^k, c = \sum_{i=1}^{n}(\mathbf{x}_i^k)^T\mathbf{B}_i\mathbf{x}_i^k, d = c - \mathbf{b}^T\mathbf{B}^{-1}\mathbf{b}, \hat{\mathbf{x}}^k = \mathbf{B}\mathbf{x}^k - \mathbf{b}, \mathbf{P} = \frac{2}{n}\sum_{i=1}^{n}\mathbf{P}_{\mathbf{d}i}, \mathbf{p}_1 = \frac{1}{n}\sum_{i=1}^{n}\mathbf{P}_{\mathbf{d}i}\mathbf{x}_{i-1}, \mathbf{p}_2 = \frac{1}{n}\sum_{i=1}^{n}\mathbf{P}_{\mathbf{d}i}\mathbf{x}_{i+1}$. The algorithm flow is given in Alg. 2. A detailed description with equation derivation can be found in [16].

4 Fairing of Surfaces with Boundary

In our approach we are able to cope also with surfaces with boundary. Vertices of such surfaces cannot be treated equally, however. Surface vertices are either boundary vertices (which belong to at least one boundary edge) or inner surface vertices. For each boundary, the fairing algorithm for curves Alg. 1 is applied (Sec. 2.3). For inner surface vertices, the fairing algorithm for surfaces Alg. 2 is used (Sec. 3.3) with a single difference: when evaluating $\bar{H}^k$, only curvatures of direct neighbours which are *not* boundary vertices are used. Note that inner surface does not affect the boundary, while boundary affects the inner surface. Stopping criteria controls the same properties as for individual curve or surface fairing.

Algorithm 2 Surface Fairing

Set $k = 1$.

1: Set $i = 1$. If the current fairing criterion is irregularity, go to step 6.
2: Evaluate $\mathbf{H}^k = \frac{3}{2} \cdot \frac{\hat{\mathbf{x}}^k}{(\hat{\mathbf{x}}^k)^T \mathbf{B}^{-1} \hat{\mathbf{x}}^k + d}$, $H^k = \|\mathbf{H}^k\|$.
3: Compute $H^{k+1} = (1-\lambda)H^k + \lambda \bar{H}^k$, where $\bar{H}^k = \frac{1}{n} \sum_{i=1}^{n} s_i^k \cdot H_i^k$ and s_i^k are signs with which the curvature vectors $\mathbf{H}_i^k$ are transfered to $\mathbf{x}^k$. They are positive, if the curvatures are on the same side of the surface as $\mathbf{H}^k$, negative otherwise.
4: Update $\mathbf{x}^{k+1} \leftarrow \mathbf{x}^k + \alpha^k \mathbf{B}^{-1} \hat{\mathbf{x}}^k$, α^k is the solution of quadratic equation, $P = \frac{(\hat{\mathbf{x}}^k)^T \mathbf{B}^{-1} \hat{\mathbf{x}}^k}{(\hat{\mathbf{x}}^k)^T \mathbf{B}^{-1} \hat{\mathbf{x}}^k + d}$:
$$(\alpha^k)^2 (H^{k+1} \cdot P) + \alpha^k (2H^{k+1} \cdot P - H^k) + H^{k+1} - H^k = 0.$$
Accepted $\alpha_i^k \in (-1 - \sqrt{\frac{d}{\hat{\mathbf{x}}^k)^T \mathbf{B}^{-1} \hat{\mathbf{x}}^k}}, -1 + \sqrt{\frac{d}{\hat{\mathbf{x}}^k)^T \mathbf{B}^{-1} \hat{\mathbf{x}}^k}})$.
5: Set $i = i + 1$. If $i \leq n$, go to step 2. Otherwise, go to step 10.
6: Evaluate $\mathbf{U}^k = \frac{1}{2} \cdot (-\mathbf{P}\mathbf{x}^k + \mathbf{p}_1 + \mathbf{p}_2)$, $U^k = \|\mathbf{U}^k\|$.
7: Compute $U^{k+1} = (1 - \lambda_u) U^k$.
8: Update $\mathbf{x}^{k+1} \leftarrow \mathbf{x}^k + \lambda_u (-\mathbf{P}\mathbf{x}^k + \mathbf{p}_1 + \mathbf{p}_2)$.
9: Set $i = i + 1$. If $i \leq n$, go to step 6. Otherwise, go to step 10.
10: If average vertex move is smaller than a threshold θ_{mov} or $k > k_{max}$, terminate.
11: Set $k = k + 1$, go to step 1.

5 Experiments

We tested the algorithm stability and robustness on synthetic data, to be comparable with other fairing algorithms on standard datasets.[1] To the data, we have added random noise with $\mu = 0$ and σ given in each test. The noise was generated independently for each vertex and coordinate. Finally, on real datasets obtained from 3D scene reconstruction, we show our algorithm usefulness for this kind of application and demonstrate its ability to cope with datasets of up to 1.7M triangles. Parameters were set to: $\lambda = 0.1, \lambda_u = 0.01$ for all tests, iterations are given in each experiment.

Synthetic Data The first experiment, Fig. 5, studies algorithm robustness to even extreme noise (triangles even intersect each other). The plot on the right shows decreasing of average criteria H_{avg}, U_{avg} demonstrating surface improvement: After 150 iterations, it is already well-faired, more iterations do not improve neither the shape nor the criteria, but still the qualitative shape is preserved. The second experiment, Fig. 6, demonstrates both robustness and stability. To an original object, noise with $\sigma = E_{avg}$ (the average edge length) was added, which has been faired out after 100 iterations. Then, noise with $\sigma = \frac{1}{2} E_{avg}$ was added again, which has been faired after 50 iterations. The final shape is very similar to the one before noise adding demonstrating the stability. We see the qualitative shape (convex/concave/hyperbolic/elliptic) has been preserved.

Standard Datasets We have selected widely-used datasets: Bunny (34.8k vertices), Dragon (437.6k vertices) and Horse (48.5k vertices). The experiment with Bunny, Fig. 7, shows the whole fairing process: to the original mesh, noise with $\sigma = \frac{1}{5} E_{avg}$ was added. After 10 iterations, most of the noise is already faired out. After 20 iterations it corresponds almost exactly to the original one, after 1000 iterations surface details, such as bumps, disappeared, but still the qualitative shape is preserved even after so many

[1] by visual comparison with published results

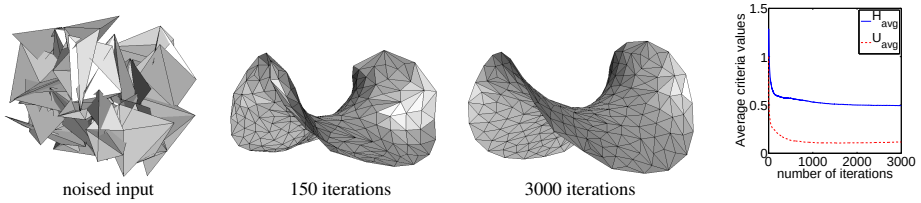


Fig. 5. Saddle fairing, added noise with $\sigma = 2E_{avg}$. Plot shows decreasing criteria with increasing iteration, qualitative shape preserved despite extreme noise and many iterations

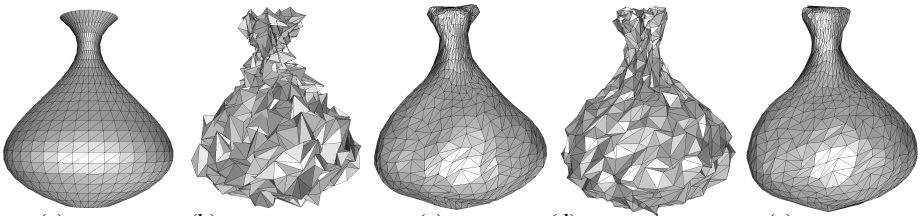
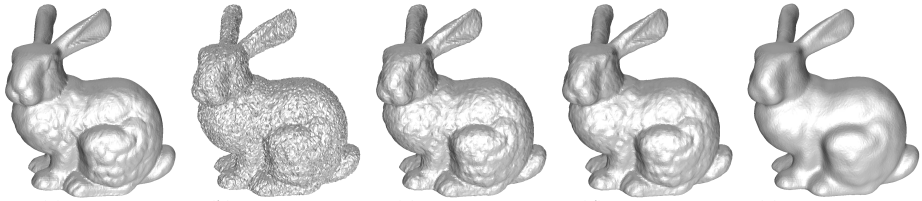


Fig. 6. Vase fairing: to original solid, extreme noise was added which has been faired out. Then again we added noise and fairing ended in very similar shape to that after the first fairing

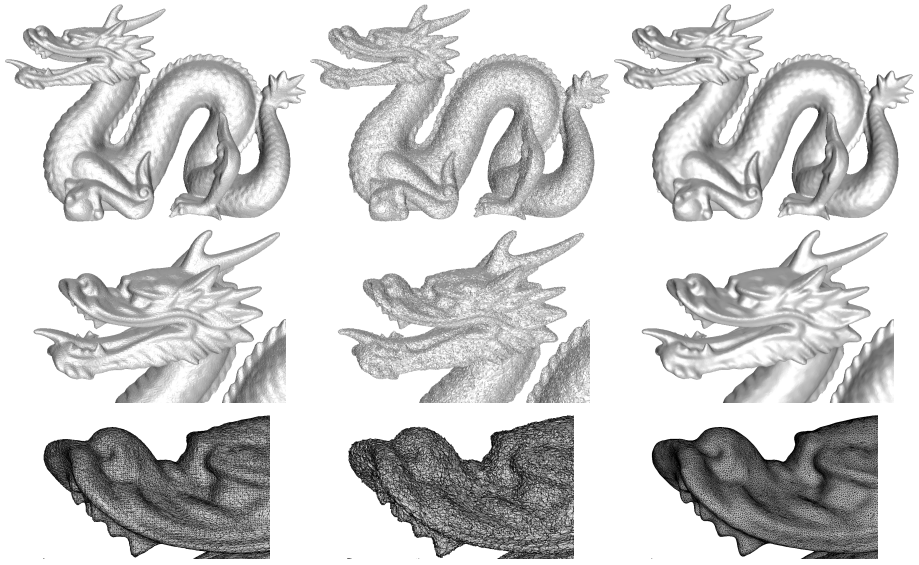
iterations. Running-time of one iteration was about 0.85s on CPU Intel C2 2.83GHz. In comparison to [9, 12], we are able to better control the fairing process and thus not to over-smooth the result. The second experiment, Fig. 8, shows our ability to preserve even fine features: To the original mesh, noise of $\sigma = \frac{1}{5}E_{avg}$ was added and faired out. The result corresponds to the original model very well. One iteration lasted 2min. Unlike in [14], we were able to keep not just sharp edges, but also finer features, e.g. scales. With more iterations, we are able to smooth these features too. Closeups show improvement in both mesh smoothness and regularity as compared to the original model. The third test, Fig. 9, demonstrates both curvature and irregularity fairing: To the original mesh, bigger noise with $\sigma = \frac{1}{4}E_{avg}$ was added. The resulting object has improved curvature (it corresponds to the original one) as well as irregularity, which is better than in the input (note the poor triangulation on the neck and hind leg), and even than in [15] which moreover cannot deal with noise. One iteration took about 1s.

Real Data Finally, performance on real data obtained from our stereo 3D reconstruction pipe-line [3] is shown. The first model, Fig. 10, is of 26194 vertices and 51879 triangles. Clearly, both smoothness and mesh regularity need fairing. Only 20 iterations sufficed to achieve an acceptable result. The last experiment (Fig. 11) demonstrates our ability to cope with moderate-size data (860571 vertices and 1718800 triangles). The overall view is the faired model, the closeups show effectivity of our algorithm.



(a) original (b) noised input (c) 10 iterations (d) 20 iterations (e) 1000 iterations

Fig. 7. Bunny: noise $\sigma = \frac{1}{5} E_{avg}$ (original mesh courtesy of Stanford University Computer Graphics Laboratory)



original noised input 50 iterations

Fig. 8. Dragon: noise $\sigma = \frac{1}{5} E_{avg}$ (original mesh courtesy of Stanford University Computer Graphics Laboratory)

6 Conclusion

We have proposed a new fairing algorithm for triangulated meshes. Two criteria, curvature and irregularity, are defined on discrete surface and jointly optimised by discrete diffusion of a scalar function. Performed experiments proved the method is robust, stable, able to cope with extreme noise, preserves qualitative shape, does not shrink the object and is well applicable to real data of 0.8M vertices and 1.7M triangles.

Despite using diffusion, which is in principle a slow process, our algorithm is efficient, since only a few iterations suffice to achieve a faired surface. Fairing may remove some surface structures. This behaviour can be reduced by a vertex weighting scheme which makes individual vertices more or less rigid. This rigidity can be controlled by input image gradient, for instance.

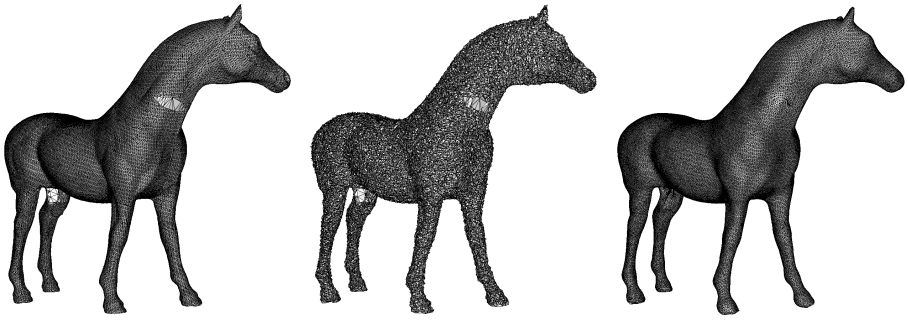


Fig. 9. Horse: noise $\sigma = \frac{1}{4}E_{avg}$, best visible electronically (original data courtesy of Cyberware, Inc.)

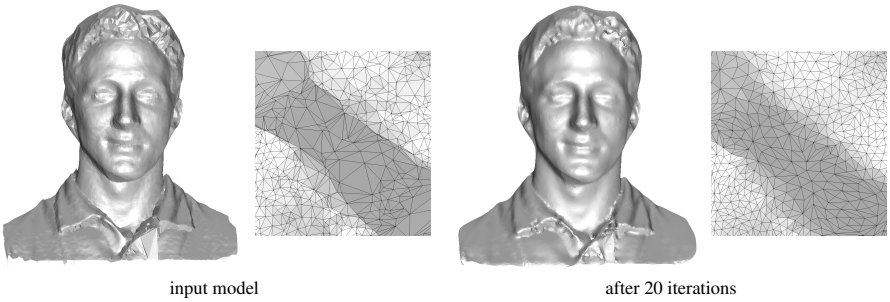


Fig. 10. Face fairing: details demonstrate high improvement in mesh quality

It seems parameter tuning is not necessary to achieve good results for individual datasets, as confirmed by our experiments, where we used the same setting, except for the number of iterations (which, in fact, is not a parameter since it is specified by the user depending on the required degree of fairing). Our algorithm is not applicable to fairing only, for instance it can be used for surface blending without any change.

Acknowledgements We thank anonymous reviewers for their helpful comments and suggestions contributing to final paper quality. This work has been supported by Czech Academy of Sciences project 1ET101210406, by Czech Ministry of Education project MSM6840770012 and by STINT Foundation project Dur IG2003-2 062.

References

1. Nistér, D.: Automatic dense reconstruction from uncalibrated video sequences. PhD thesis, Royal Institute of Technology KTH, Stockholm, Sweden (2001)
2. Strecha, C., Tuytelaars, T., van Gool, L.: Dense matching of multiple wide-baseline views. In: Proceedings International Conference on Computer Vision. (2003) 1194–1201
3. Kamberov, G., et al.: 3D geometry from uncalibrated images. In: Proceedings of the ISVC '06. Volume 4292 of LNCS., Springer (2006) 802–813

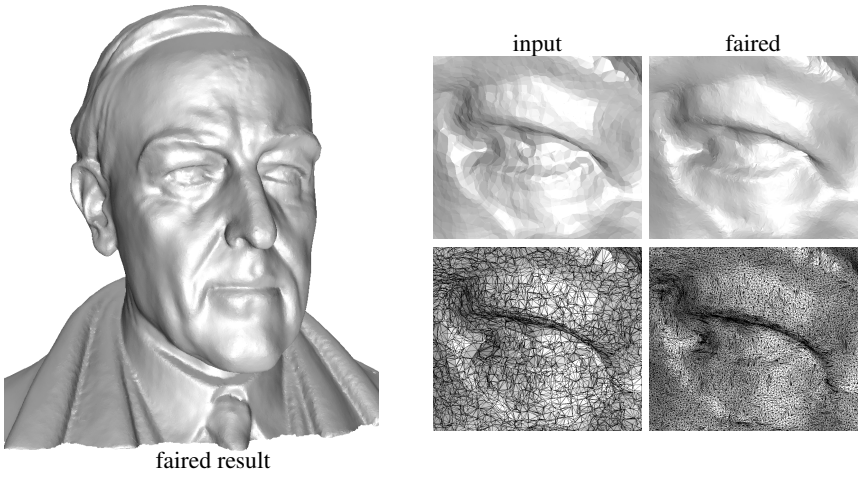


Fig. 11. W.Wilson 3D model fairing, best visible electronically (digitised plaster model ©APF)

4. Duan, Y., Qin, H.: A novel modeling algorithm for shape recovery of unknown topology. In: Proceedings International Conference on Computer Vision (ICCV01). (2001) 402–411
5. Dyn, N., Hormann, K., Kim, S., Levin, D.: Optimizing 3D triangulations using discrete curvature analysis. In: Mathematical Methods for Curves and Surfaces. (2001) 135–146
6. Kobbelt, L.P.: Discrete fairing and variational subdivision for freeform surface design. The Visual Computer **16** (2000) 142–158
7. Taubin, G.: A signal processing approach to fair surface design. In: Proceedings of the SIGGRAPH'95. (1995) 351–358
8. Schneider, R., Kobbelt, L.: Geometric fairing of irregular meshes for free-form surface design. Computer Aided Geometric Design **18** (2001) 359–379
9. Desbrun, M., Meyer, M., Schröder, P., Barr, A.H.: Implicit fairing of irregular meshes using diffusion and curvature flow. In: Proceedings of the SIGGRAPH '99. (1999) 317–324
10. Surazhsky, T., Magid, E., Soldea, O., Elber, G., Rivlin, E.: A comparison of gaussian and mean curvatures estimation methods on triangular meshes. In: IEEE International Conference on Robotics and Automation 2003 (ICRA '03). (2003) 1021–1026
11. Delingette, H.: Simplex meshes: a general representation for 3D shape reconstruction. Research Report 2214, INRIA, Sophia Antipolis (1994)
12. Yagou, H., Ohtake, Y., Belyaev, A.: Mesh smoothing via mean and median filtering applied to face normals. In: Geometric Modeling and Processing, 2002. (2002) 124–131
13. Mashiko, T., Yagou, H., Wei, D., Ding, Y., Wu, G.: 3D triangle mesh smoothing via adaptive MMSE filtering. In: Int. Conf. on Computer and Information Technology. (2004) 734–740
14. Jones, T.R., Durand, F., Desbrun, M.: Non-iterative, feature-preserving mesh smoothing. In: Proceedings of the SIGGRAPH '03. (2003) 943–949
15. Escobar, J.M., Montero, G., Montenegro, R., Rodrguez, E.: An algebraic method for smoothing surface triangulations on a local parametric space. International Journal for Numerical Methods in Engineering **66** (2006) 740–760
16. Kostlivá, J., Šára, R., Matýsková, M.: Inflection point preserving fairing of discrete surfaces with boundary. Research Report CTU–CMP–2008–19, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic (2008)