Third Hungarian Conference on Computer Graphics and Geometry, Budapest, 2005

Local Fairing of Freeform Curves and Surfaces

Péter Salvi[1,2] and Tamás Várady[2]

[1] Loránd Eötvös Science University, Budapest
[2] Geomagic Hungary, Budapest

Abstract

After reviewing different approaches, a new algorithm is presented for fairing B-spline curves and surfaces. It is based on a special target curvature*, computed from the not-yet-faired curve or surface. The method is parameter invariant and local. It moves a single control point at a time, so to find a global optimum iterative methods with appropriate heuristics need to be applied. The results are illustrated by a few examples.*

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling

1. Introduction

Digital Shape Reconstruction (formerly called Reverse Engineering) is a fast growing area in Computer Aided Geometric Design, which deals with the creation of geometric models using measured data. In many practical applications of DSR, surface fairness is a crucial matter, in particular in the automobile industry. Although fairness does not have an exact mathematical definition, researchers agree that it is inherent to pleasing aesthetics, and that the curvature of fair surfaces must be distributed evenly. A wide range of graphical *interrogation methods* (e.g. curvature maps, isophote and reflection lines) has been developed to detect small surface artifacts, but even with these, today fairing is a laborious manual process that needs a lot of skill. This is why there is a natural need for (semi) automatic fairing algorithms.

There are various approaches that can make a surface more fair. (i) Variational methods integrate fairing into the surface approximation process. (ii) Postprocessing methods usually define some *fairness measure*, and try to minimize it by changing the existing surface geometry, while maintaining some constraints, such as the maximum deviation from the original surface. Here we will only deal with the latter approach that seems to be more useful in the DSR context. It is important to note, that though the faired surfaces should be smooth, the highly curved *features* of the original shapes must be preserved.

Since research on surface fairing methods inevitably involves research on two-dimensional curve (spline) fairing, here we will summarize the previous work on fairing both curves and surfaces, introducing and comparing the most important fairness measures and algorithms.

In Section 2 different concepts of fairness will be presented. Section 3 gives an overview on previous work in the literature. Section 4 presents our proposed algorithm, followed by test results in Section 5.

2. Fairness

Fairness may have different meanings in different applications. Depending on the requirements, even the same surface can be qualified as fair or unfair, however, there are general guidelines that can be applied to most of the cases.

One widely used criterion of fairness is the smoothness and smooth distribution of reflection lines. If a fair object was placed into a room lit by parallel lights, the reflections of the light source should bend smoothly and evenly over the surface. This effect can be simulated using computer graphics — the most modern modelling systems offer these kinds of rendering options.

Isophote lines are very similar to reflection lines, since their smoothness depends on the change of the first derivative of the surface. An isophote line is a set of surface points where the angle between the normal vector and the viewing direction is the same (within a given tolerance). Since this map is much simpler than the previous, but reveals just about the same flaws, it is more often seen in practice. Another variant is the highlight band,[2] which can also be computed very fast.

While these maps simulate something that is visible to the eye under special conditions, there are others that only have a mathematical meaning, but still proved to be crucial in examining fairness. These are the curvature maps and the curvature combs, which depend on the second derivatives. Curvature maps colour-code the curvature values and can have various types (Gaussian, mean, minimum, maximum, etc.), curvature combs display the values as orthogonal straight line segments along a curve.

These methods are also called visual interrogation tools, because they help the user to find minor discontinuities or wiggles on the surface. Since these are found by looking at the *changes* of the map, they show flaws of one degree higher than the derivatives the tool depends on.

In order to create a fairing algorithm, it is common to define a *fairness measure*, i.e. a functional that represents the fairness of the surface. In other words, we can say that a surface S is fair, if

$$\mathcal{F}(S) < \tau$$

applies, where τ is a user-defined tolerance. One "classical" definition of fairness by Farin and Sapidis is as follows:

> A curve is fair if its curvature plot is continuous and consists of only a few monotone pieces.[4]

In the next section, we will review what sort of other alternatives exist.

3. Previous Work

In this section we introduce the most well-known or prominent measures and algorithms in the literature. We will also present curve-fairing methods, since most results for surfaces can be easily generalized from them.

As Roulier and Rando point out, we cannot hope to have a universal fairness measure or algorithm, but we should strive to create new ones, in order to give designers the freedom of choosing the most suitable algorithms for their tasks.[14]

3.1. Fairness measures

A natural measure for curve fairness is the strain energy, that is based on a drawing technique used in ship design, from the 18th century until today. To create a smooth curve, metal weights were placed at the interpolation points and a flexible spline was spanned between them. The resulting curve **c** minimizes the strain energy, yielding the measure

$$E = \int (\kappa(s))^2 \mathrm{d}s, \tag{1}$$

where $\kappa(s)$ is the curvature of the curve as a function of the arc length. This minimizes the mean curvature, while giving a penalty to the extreme values by squaring.[14]

Computing the curvature can be difficult, so it is often replaced by a simpler, parameter-dependent formula:

$$\hat{E} = \int (\mathbf{c}''(t))^2 \mathrm{d}t. \tag{2}$$

The third degree interpolating spline minimizes this value, but has the drawback that in cases where the parameterization substantially differs from the arc-length parameterization, fairness is not guaranteed, and unexpected results may occur.

Since neither (1) nor (2) penalizes the sign change of the curvature, curves faired by these measures may preserve unwanted inflections. Different variations of the interpolating spline were devised to avoid this, e.g. the spline in tension or the ν-spline.[4]

Moreton and Séquin introduced another measure, called Minimum Variation Curve, optimizing the variation of the curvature:[12]

$$E_{MVC} = \int (\kappa'(s))^2 \mathrm{d}s. \tag{3}$$

This has the advantage that it does not create unnecessary inflection points due to its convexity-preserving property.

The curve-fairing measures introduced so far all have their surface-fairing equivalents. Similar to the strain energy, in the surface case we can minimize the thin plate energy

$$\Pi_P = \iint_S a(\kappa_1^2 + \kappa_2^2) + 2(1-b)\,\kappa_1\kappa_2\,\mathrm{d}S, \tag{4}$$

where κ_1 and κ_2 are the principal curvatures of the surface S and a, b are material-specific constants that usually take the values $a = 1$ and $b = 0$ or $b = 1$.[7]

This also has a simpler variant

$$\Pi = \iint_A S_{uu}^2 + 2S_{uv}^2 + S_{vv}^2\,\mathrm{d}u\,\mathrm{d}v, \tag{5}$$

which is parameter dependent, so it can only be used safely for isometric parameterization.

Moreton and Séquin suggested an alternative measure based on the variation of the curvature:

$$\Pi_{MVS} = \iint_S \left(\frac{\partial\kappa_1}{\partial e_1}\right)^2 + \left(\frac{\partial\kappa_2}{\partial e_2}\right)^2 \mathrm{d}S, \tag{6}$$

that vanishes on spheres, cones and tori. Although this measure gives excellent results, it requires very complex computations.

3.2. Fairing algorithms

One of the simplest, widely used curve-fairing method is *knot removal and reinsertion* (KRR), originally conceived by Kjellander and later made local by Sapidis and Farin.[3]

If we define an order k B-spline as

$$\mathbf{x}(t) = \sum_{i=0}^{n} \mathbf{d}_i N_{i,k}(t), \quad t \in [t_{k-1}, t_{n+1}], \tag{7}$$

where $\mathbf{d}_i$ are the control points, $N_{i,k}$ are the B-spline bases and $T = (t_j)_{j=0}^{n+k}$ are the knots, then it is at most C^{k-2}-continuous at the knot points.

We can add knots to a B-spline in a way that its shape does not change. This can be unambiguously done by multiplying the control points with a matrix. On the other hand, if we take out a knot, we can only preserve the shape if the curve was originally C^{k-1}-continuous at that knot point.

So the problem is locating the control points of

$$\tilde{\mathbf{x}} = \sum_{i=0}^{n-1} \tilde{\mathbf{d}}_i N_{i,k,\tilde{T}} \qquad (\tilde{T} \subset T) \tag{8}$$

in such a way that $\mathbf{d} = A\tilde{\mathbf{d}}$ applies, where A is the knot insertion matrix.[8] This breaks down to an overdefined equation that can have several approximate solutions. Farin gives the most local solution for third degree B-splines, ensuring C^3 continuity at the knot by repositioning only one control point.[4]

This gives the idea of the KRR algorithm, i.e. to find, remove and then reinsert the knot where the third derivative has the largest discontinuity. The process may be iterated until a suitable end condition is met. Finding such a condition is not a trivial task. A vast range of heuristics can be applied, including best-first-search[8] and simulated annealing.[13]

Eck and Hadenfeld fix all but one control points and locally minimize the fairness measure

$$E_l = \int_{t_{k-1}}^{t_{n+1}} \left(\tilde{\mathbf{x}}^{(l)}(t)\right)^2 \quad l = 2,3 \tag{9}$$

while keeping the distance from the original curve under a δ tolerance:[3]

$$\max\{|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)| \mid t \in [t_{k-1}, t_{n+1}]\} \leq \delta. \tag{10}$$

Because of the convex hull property, this is easily done by preserving the $|\mathbf{d}_r - \tilde{\mathbf{d}}_r| \leq \delta$ inequality:

$$\tilde{\mathbf{d}}_r^* = \mathbf{d}_r + \delta \cdot \frac{\tilde{\mathbf{d}}_r - \mathbf{d}_r}{|\tilde{\mathbf{d}}_r - \mathbf{d}_r|}. \tag{11}$$

Both of these methods have equivalents in surface fairing. The main disadvantage of the KRR algorithm is that removing a knot changes a whole line of control points in the other parametric direction, e.g. if we have a surface

$$X(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} \mathbf{d}_{ij} N_{i,k,U}(u) N_{j,l,V}(v), \quad (u,v) \in [u_{k-1}, u_{n+1}] \times [v_{l-1}, v_{m+1}], \tag{12}$$

where $U = (u_i)_{i=0}^{n+k}$ and $V = (v_j)_{j=0}^{m+l}$ represent the knots, removing a knot v_s means removing a knot from all of the B-splines $\mathbf{x}_i = \sum_{j=0}^{m} \mathbf{d}_{ij} N_{j,l,V}(t)$, where $i = k, \ldots, n$. Furthermore, the generalized KRR only ensures C^3 continuity in one parametric direction.

Hahmann proves that it is sufficient to remove and reinsert a knot in only three rows or columns of B-splines, thus the algorithm can be made local for surfaces.[8] However, C^3 continuity in only one direction is not satisfactory in real-life applications.

Hadenfeld proposed a fairing method using the measure (5), as above only one control point is moved at a time and the largest deviation is constrained from the original.[7]

In a recent publication[1] fairing was performed through the optimization of knot vectors; in our research, however, we preserve the original knots.

4. The New Algorithm

In this section we first sketch a fairing algorithm for curves, then we generalize it for surfaces. We can expect that the curvature comb of a fair curve is smooth, without any jumps or sudden changes. Therefore we can smooth the curve defined by the curvature comb's endpoints, which is practically the same as the evolute, due the $\kappa = 1/\rho$ equality. We will call the smoothed curve the *target evolute*.

Now we want to find a curve that is close to the original, but whose evolute is the target evolute. This also defines a fairness measure: the closer the evolute is to the target evolute, the fairer is the curve. Let $\mathbf{n}$ denote the normal and $\mathbf{e}$ the target evolute, then our fairness measure is

$$E = \int \|(\mathbf{c}(t) + \rho\mathbf{n}(t)) - \mathbf{e}(t)\|^2 \mathrm{d}t, \tag{13}$$

assuming that the two curves have a common parameterization. The algorithm for finding the minimum of this functional will be presented later. Controlling the deviation from the original curve can be managed in the same manner as written in the previous section.

Since the evolute (and the curvature comb) may be self-intersecting, we use directly the curvatures instead, so our measure becomes

$$\hat{E} = \sum_i |\kappa(t_i) - g(t_i)|^2, \tag{14}$$

where g is the smoothed (target) curvature.

In the surface case the single curvature need to be replaced by the two principal curvatures. Let g_1 and g_2 be the target curvatures based on κ_1 and κ_2, then

$$\hat{\Pi} = \sum_i \sum_j (|\kappa_1(u_i,v_j) - g_1(u_i,v_j)|^2 + |\kappa_2(u_i,v_j) - g_2(u_i,v_j)|^2) \tag{15}$$

is a meaningful fairness measure.

4.1. Determining the target curvature

Any simple and fast smoothing method can be effectively used for defining the target curvature, for example averaging the consecutive sample points of the evolute. Smoothness of the target curvature is much more important than to be close to the evolute of the original curve, therefore we should use a loose sampling rate. Global averaging can remove parts of the curvature that represent features, so the user should be allowed to restrict the smoothing or edit the target curvature manually.

Another possibility is to fit a NURBS curve over the sampled points. For surfaces this leads to the solution of a system of linear equations that minimizes the functional

$$F(\mathbf{g}_q) = \sum_i \sum_j \|\mathbf{g}_q(u_i, v_j) - \kappa_q^0(u_i, v_j)\|^2 \quad (q = 1, 2), \quad (16)$$

where κ_q^0 is a principal curvature of the original surface (here the curvatures are interpreted as surfaces over the parameter domain). To get smooth results, we should also minimize the curvature of the fitted surface:

$$\hat{F}(\mathbf{g}_q) = \sum_i \sum_j \|\mathbf{g}_q(u_i, v_j) - \kappa_q^0(u_i, v_j)\|^2 + \int_u \int_v \hat{\kappa}_q(u, v)\, \mathrm{d}u\, \mathrm{d}v, \quad (17)$$

where $\hat{\kappa}_g$ is the curvature of $\mathbf{g}_q$.

Having the target curvatures for our curves and surfaces, the next step is to modify the current entities in such a way, that their curvature gradually gets closer to the target.

4.2. Finding the optimal solution

For simplicity's sake here we present an iterative method that moves only one control point in every iteration. This has the advantage of locality in exchange for speed, but this drawback is countered by our choice of minimization algorithm — the downhill simplex method, which is simple and very fast.[13] An iteration consists of the following steps:

1. Select the next control point to move from a priority queue.
2. Minimize the fairness measure by moving the selected control point.
3. Calculate a new control point position if it falls too far from the original (see (11)).

Selection of the next control point has great influence on the quality of fairness. Selecting the control point where the largest deviation of the target curvature occurs is a natural choice. However, this can lead to a deadlock, if the same control point is chosen over and over again. A list of the recently moved control points may be kept in order to avoid this. Also, boundary control points should not be selected for most applications.

Other minimization procedures to find the global optimum can also be used, such as simulated annealing.

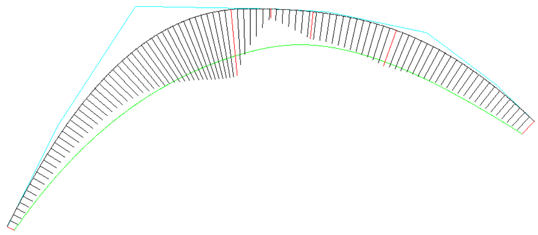

Figure 1: *The initial curve with its target curvature in green.*

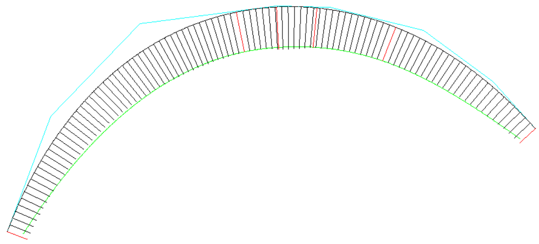

Figure 2: *The faired curve.*

5. Results

Figure 1 shows a curve before fairing. The original curvature comb and the original control polygon are also shown. The green line is a smooth version of the given curvature comb, this is the target curvature we want to approximate. Figure 2 shows the curve after fairing, the final curvature distribution is clearly much better, though it is not necessarily identical to the target function due to tolerance constraints and the final number of steps. Figures 3–4 highlight the effect of fairing through extrusion surfaces.

Figure 5 shows the actual and desired target curvature distributions for a car body surface element. Table 1 summarizes the numerical results. As we can see, there is minimal

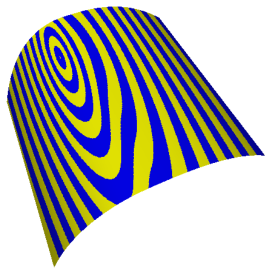

Figure 3: *Extrusion surface of the initial curve.*

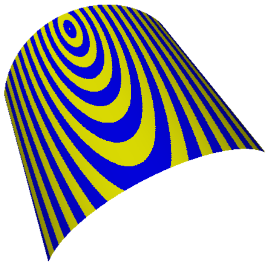

Figure 4: *Extrusion surface of the faired curve.*

Iteration	Fairness measure	Max. distance (mm)
0	0.901531	0.00000
200	0.244919	2.91302
400	0.223582	4.40624
600	0.215052	5.02049
800	0.201543	5.14342
1000	0.198280	5.85641

Table 1: *Fairing phases.*

improvement after 400 iterations, which justifies to stop iterating. Figure 6 shows the isophote lines of the original surface before and after fairing. Figure 7 is a deviation map to show where the largest positional modifications took place in order to get the final faired surface.

6. Conclusions

Fairing curves and surfaces is a complex problem. Unfortunately, the goal of generating perfectly fair shapes cannot be unambiguously formulated with mathematical terms,

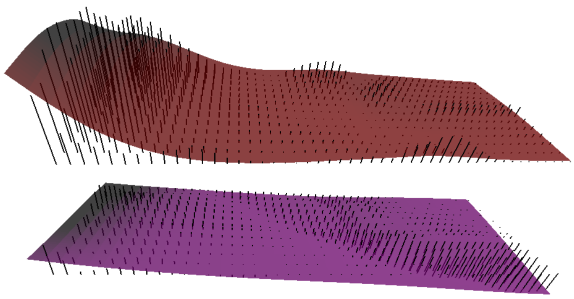

Figure 5: κ_1 *and* κ_2 *principal curvatures (teeth) and the target curvature functions.*

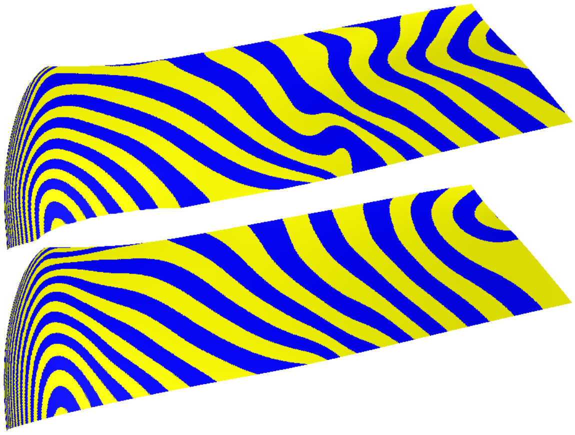

Figure 6: *Isophote map of the initial and final surface.*

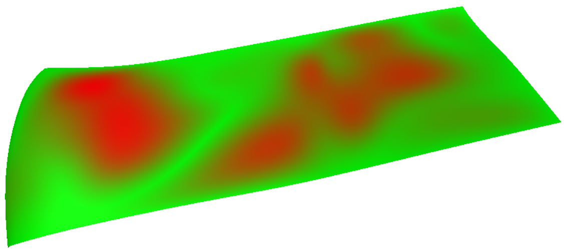

Figure 7: *Distance map showing deviations from the original surface.*

and there are many alternatives. Authors propose a method where a smoothed target curvature function is approximated step by step. The algorithm modifies a single control point at a time. Applying an iterative strategy, the global shape is optimized until the magnitude of the improvement becomes negligible. Our future research is going to replace the current iterative methods by direct methods, that can efficiently lead to fair curves and surfaces.

Acknowledgements

This research has been conducted within Geomagic Hungary, Ltd., Budapest. The first author is a PhD student at the Information Faculty of the Loránd Eötvös Science University.

References

1. B. Aszódi, Sz. Czuczor, L. Szirmay-Kalos, *NURBS Fairing by Knot Vector Optimization*. Journal of WSCG, Volume 10, pp. 19–26, 2004.
2. K-P. Beier, Y. Chen, *The Highlight Band, a Simplified*

Reflection Model for Interactive Smoothness Evaluation. In: N. S. Sapidis (Ed.), Designing Fair Curves and Surfaces, pp. 213–230, SIAM, ISBN 0-898-71332-3, 1994.

3. M. Eck, J. Hadenfeld, *Local Energy Fairing of B-Spline Curves*. Computing Supplement 10, pp. 129–147, 1995.

4. G. Farin, *Curves and Surfaces for Computer Aided Geometric Design. A Practical Guide*. Academic Press, 5th edition, 2002.

5. G. Farin, J. Hoschek, M.-S. Kim (Eds.), *Handbook of Computer Aided Geometric Design*. North-Holland, 2002.

6. J. Hadenfeld, *Fairing of B-Spline Curves and Surfaces*. In: J. Hoschek, P. Kaklis (Eds.), Advanced Course on FAIRSHAPE, pp. 59–75, Teubner Stuttgart, ISBN 3-519-02634-1, 1996.

7. J. Hadenfeld, *Local Energy Fairing of B-Spline Surfaces*. In: M. Daehlen, T. Lyche, L. L. Schumaker (Eds.), Mathematical Methods for Curves and Surfaces, pp. 203–212, Vanderbilt University Press, ISBN 0-826-51268-2, 1995.

8. S. Hahmann, S. Konz, *Knot-Removal Surface Fairing using Search Strategies*. Computer Aided Design 30, pp. 131–138, 1998.

9. S. Hahmann, *Shape improvement of surfaces*. Computing Supplement 13, pp. 135–152, 1998.

10. M. Hoffmann, *Geometric and Solid Modeling. An Introduction*. Morgan Kaufmann Publishers, 1989.

11. H. P. Moreton, C. H. Sequin, *Functional Optimization for Fair Surface Design*. ACM SIGGRAPH Computer Graphics, Volume 26, Issue 2, pp. 167–176, 1992.

12. H. P. Moreton, C. H. Sequin, *Minimum Variation Curves and Surfaces for Computer-Aided Geometric Design*. In: N. S. Sapidis (Ed.), Designing Fair Curves and Surfaces, pp. 123–159, SIAM, ISBN 0-898-71332-3, 1994.

13. W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery, *Numerical Recipes in C*. Cambridge University Press, 2nd Edition, ISBN 0-521-43108-5, 1992.

14. J. Roulier, T. Rando, *Measures of Fairness for Curves and Surfaces*. In: N. S. Sapidis (Ed.), Designing Fair Curves and Surfaces, pp. 75–122, SIAM, ISBN 0-898-71332-3, 1994.