

# Fairing Bicubic B-Spline Surfaces using Simulated Annealing

Stefanie Hahmann and Stefan Konz

**Abstract.** In this paper we present an automatic fairing algorithm for bicubic B-spline surfaces. The fairing method consists of a knot removal and knot reinsertion step which locally smoothes the surface. The simulated-annealing search strategy is used to search for the global minimum of the fairing measure.

## §1. Introduction

Free form surfaces are an indispensable part of powerful CAD-systems. Bicubic tensor product B-splines, which are the subject of this paper, are often used in geometric modelling due to their well known advantages which result basically from the local support of the basis splines and their dependence on the knot vector. However, the designer is not always satisfied with the fairness or smoothness of the resulting surface obtained from interpolation or approximation of some data sets. A fair surface can be obtained by two different ways:

- modelling surfaces with constraints (see e.g. [10,2,16,7,18,8]),
- post-processing surface fairing (see e.g. [4,14,9]).

In the present paper we introduce a new *post-processing* fairing method for bicubic tensor product B-spline surfaces based on the knot-removal reinsertion step [6] combined with a search strategy, called *simulated annealing*. This method will be local in contrast to Kjellander's method for bicubic surfaces [14]. The principal reason for incorporating a search strategy into the fairing process is to overcome the drawback of other iterative algorithms which stop when the fairing measure achieves its first local minimum.

Section 2 will recall an algorithm of Farin/Sapidis for curves [20]. A fairing algorithm for B-spline surfaces based on a special search strategy is presented in Section 3. The heuristic *simulated-annealing algorithm*, depending on a probability function and some other parameters searches for a *global minimum* of the fairness measure. It results in an optimal surface with respect to its fairness. Some examples are shown at the end.

## §2. Knot Removal for B-spline Curves and Surfaces

Given the positive integers  $n, m$  and  $k, l$  and  $\mathbf{u} = (u_i)_{i=0}^{n+k}$ ,  $\mathbf{v} = (v_j)_{j=0}^{m+l}$  two sequences of real numbers with  $u_i < u_{i+k}$  and  $v_j < v_{j+l}$  ( $i, j = 0, \dots, n, m$ ), the B-spline basis functions associated with the *knot vector*  $\mathbf{u}$  are denoted by  $(N_{i,k,u})_{i=0}^n$  (or simply by  $N_{i,k}$ ) and are assumed to be normalized to sum to one. The same holds for  $(N_{j,l,v})_{j=0}^m$ . A parametric *tensor product B-spline surface*  $\mathbf{X}$  in  $\mathbb{R}^3$  of order  $(k, l)$  is then defined by

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

where the coefficients  $\mathbf{d}_{ij} \in \mathbb{R}^3$  form the control net. Note that the B-spline  $\mathbf{X}(t)$  is only defined over  $[u_{k-1}, u_{n+1}] \times [v_{l-1}, v_{m+1}]$ .  $u_k, \dots, u_n, v_l, \dots, v_m$  are called *inner knots*. For more details about B-splines, see [3,21].

The knot removal step consists of approximating a given spline  $\mathbf{x} = \sum_0^n \mathbf{d}_i N_{i,k,t}$  by a spline  $\tilde{\mathbf{x}} = \sum_0^{n-1} \tilde{\mathbf{d}}_i N_{i,k,\tilde{t}}$  on a knot vector  $\tilde{\mathbf{t}}$  which is a subsequence of  $\mathbf{t}$  with one knot less. While *knot insertion* doesn't change the shape of the curve, the "inverse" process of knot removal cannot be carried out in general without changing the shape, except the case where the curve is  $C^{k-1}$  at  $t_j$ , i.e. its continuity order is higher than it should be according to its multiplicity). In the future we will use knot removal for bicubic surfaces, therefore we restrict the following considerations to the cubic case.

Different possibilities to determine approximated solutions of the *knot removal problem* exist: From the approximation point of view, knot removal can be solved by  $\min \|\tilde{\mathbf{x}} - \mathbf{x}\| : \{\tilde{\mathbf{x}} \in \mathcal{S}_{k,\tilde{t}}\}$  with respect to some appropriate norm  $\|\cdot\|$ . For more details see [17]. In general, all control points are involved in that knot removal procedure, which is therefore a global one.

We want our fairing step to work as local as possible. This means, that after reinsertion of the removed knot  $t_j$ , a minimum number of control points  $\tilde{\mathbf{d}}_i$  of the curve  $\tilde{\mathbf{x}}$  should differ from the original control points  $\mathbf{d}_i$ . The knot removal problem can locally be solved by calculating an approximate solution of an over-determined (3,2)-system of linear equations. Detailed description of these knot removal algorithms can be found in [6,5].

Farin/Sapidis' fairing algorithm for cubic splines is based on the fact, that after a knot-removal-reinsertion step of the knot  $t_j$  the new curve  $\tilde{\mathbf{x}}(t)$  is  $C^3$  at  $t_j$ . Hence, the *local fairness measure*  $z_j = |\kappa'(t_j^+) - \kappa'(t_j^-)|$  decreases to zero. By repeating this procedure several times always at the most offending knot (i.e.  $t_k$  with  $z_k = \max(z_i) : \{i \in (4, \dots, n)\}$ ), they could expect, that the sum of the local measures

$$\xi = \sum_{i=4}^n z_i \quad (2)$$

decreases also. The algorithm stops if there is no more improvement of the fairness measure  $\xi$ . This is an algorithm according to the following fairness criterion:

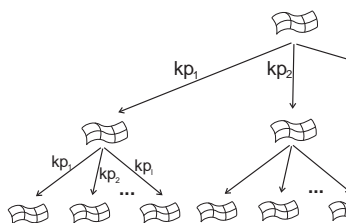
- (C) A  $C^2$ -cubic B-spline curve is fair if the curvature function is continuous, has the appropriate sign, and is as close as possible to a piecewise monotone function with as few as possible monotone pieces [20].

We want to close this section with the conclusion that a knot-removal-reinsertion step has a local fairing effect, and can therefore be used in a fairing method for the entire curve, or tensor product surface. The main problem is to find a sequence of knots such that repeated knot-removal-reinsertion at these knots leads to a (global) minimum of the global fairness measure  $\xi$ .

### §3. Surface Fairing Method

The curve fairing method can easily be extended to tensor product surfaces. Of course, one can simply apply the curve scheme to each row and each column of control points of the surface control net. This procedure has two main disadvantages: the fairness measures are not adapted to the surface geometry and therefore don't justify the selection of a knot as the most offending one. And the "stop"-criterion for the algorithm doesn't guarantee to find the global minimum of the fairness measure  $\xi$  (2). The algorithm stops when the first local minimum of that function is reached. And this is probably not the lowest one. Now a surface fairing algorithm is presented which is based on knot-removal-reinsertion (which is a very fast fairing step) and which uses simulated annealing to *optimize the fairing process*.

When searching for a global minimum, one can do it by spanning a so called *search tree* (see Figure 1).



**Figure 1.** Search tree of the iterative fairing method.

The search tree has as its root the given surface to be faired. Each vertex of the tree represents a B-spline surface that results from a fairing step (knot removal and reinsertion) at one knot of the parent vertex. The edges of the tree are marked with the knot pair  $kp_i$  which is treated by the knot operations. The search tree lists systematically all possible steps of the fairing algorithm. Each path from the root to a leaf of the tree describes the history of one algorithm call.

At least one of the leaves represents a surface with a global minimum of the fairing measure. The global minimum (after  $k$  iterations) can only be found when performing an exponential number  $O((n \cdot m)^k)$  of fairing steps. The aim of a search strategy is to find this path without constructing the

whole tree because of the exponential number of vertices. The five main characteristics of our fairing method are now presented:

### 3.1. Knot operation

We use *Knot-Removal-Reinsertion* (KRR) as a local fairing step for the surface. To do so, the knot operations (Section 2) have to be adapted to the tensor product surface description. It's an inherent property of tensor products that a lot of algorithms on surfaces (e.g. de Boor algorithm, degree elevation, etc.) can be reduced to the univariate curve algorithms with respect to the two parameters  $u$  and  $v$ . Different strategies can be adopted for KRR of a B-spline surface:

- KKR in  $u$ -direction: for  $j = 4$  to  $m$  do KKR of  $u_i$  at  $(u_i, v_j)$ ,
- KKR in  $v$ -direction: for  $i = 4$  to  $n$  do KKR of  $v_j$  at  $(u_i, v_j)$ ,
- KRR in both directions, first in  $u$  and then in  $v$  direction, or the other way round.

Notice, that there is no symmetry in the third KRR-step. Changing  $u$  and  $v$  leads to different results, because the second step depends on the first.

### 3.2. Fairing measure

The *local measure*  $(z_{ij}^u + z_{ij}^v)$  of a knot pair  $(u_i, v_j)$  from a surface  $\mathbf{X}(u, v)$  which evaluates the fairness of the knot pair is defined by the partial derivation of a curvature function  $g(u, v)$  (e.g. Gaussian curvature  $\kappa_{min} \cdot \kappa_{max}$ , or  $\kappa_{min}^2 + \kappa_{max}^2$ , etc.) along one of the two parameters of the surface:

$$z_{ij}^u = \frac{|\frac{\partial g}{\partial u}(u_i, v_j)|}{\|\frac{\partial \mathbf{X}(u, v)}{\partial u}\|}, \quad z_{ij}^v = \frac{|\frac{\partial g}{\partial v}(u_i, v_j)|}{\|\frac{\partial \mathbf{X}(u, v)}{\partial v}\|}. \quad (3)$$

According to the selected curvature function the automatic fairing algorithm will evaluate a knot pair in a special manner.

Finally the sum of the local measures of all knot pairs will define the *global fairness measure*  $\xi$  which will be minimized by the fairing algorithm:

$$\xi = \sum_{i,j} z_{i,j}^u + z_{i,j}^v. \quad (4)$$

**Remark:** One can look for the local measures  $z_{ij}^u, z_{ij}^v$  independent of each other, and apply KRR only in  $u$  or  $v$ -direction.

### 3.3. Visualization utility

An appropriate technique for emphasizing visually the fairing effect is a light reflection method. The fairing method reduces surface irregularities and provides a *more pleasing shape*. Those aesthetic aspects are well captured by isophotes, reflection line or highlight lines [19,15,1,11], because they are very sensitive to changes in the surface normals. We chose the isophote method.

### 3.4. Scheduler and termination condition

The selection of the knot pair and the search for the global minimum of the fairing measure can be done in the following way.

#### Simulated Annealing

Simulated annealing is a heuristic method to solve large optimization problems. It ensures “*with high probability*” to find a global minimum of the function of interest, where iterative methods often get stuck at a local minimum. In 1983, IBM researchers [12,13] found an interesting analogy in material physics, where the heuristic process of an optimal annealing of metal is a quit difficult task. That’s where the algorithm got it’s name.

Simulated annealing is a stepwise algorithm which is allowed to make *good steps* (i.e. steps which improve some measure of quality) and to make *bad steps* (those that do not). Those bad steps are necessary in order to leave a valley of a local minimum. The probability that bad steps are accepted decreases while the algorithm proceeds.

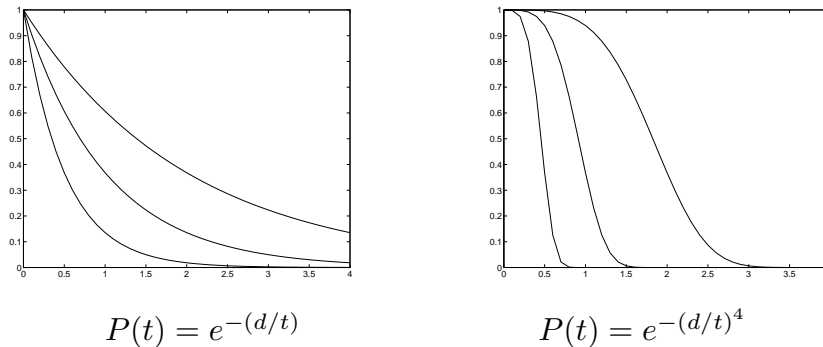
This algorithm can be compared with the *hill climbing* model, see Figure 2. Searching in this case the maximum of a function one can imagine that a hill climber is searching for the highest top. Depending of the departure he has to walk on tops (good steps) and valleys (bad steps) in order to reach the highest top. Therefore he must not stop walking when he arrives at the first top. In this case he would only find a local maximum. Sometimes the climber ought to take a descending path. This helps to evade from a local maximum. In general the topography of the problem isn’t known in advance. Hence it is natural to use a probability for the acceptance of bad steps and in order to ensure that the algorithm stops.



**Figure 2.** Topography of an optimization process.

In our fairing context, the measure of quality is the global fairness measure  $\xi$  (4), which has to be minimized by a sequence of fairing steps, the knot-removal-reinsertion (KRR)-steps. At each stage of the algorithm a knot pair  $(u_i, v_j)$  of the current surface is chosen either randomly or by a ranking list with respect to the local fairness measure  $z_{ij}^u + z_{ij}^v$  (3) at each knot pair. While the KRR-step increases  $\xi$ , the steps are accepted as good steps. Otherwise, bad steps are accepted with a certain probability. As the probability to make bad steps gets smaller during the algorithm it gets less and less possible to leave the current valley of the fairness measure and the algorithm stops there.

Let  $max\_kp$  be the number of inner knot pairs. If the number of stages of the algorithm is controlled by the integer  $max\_step$ , the probability function  $P$  at each stage  $t_k$  ( $k = 1, \dots, max\_step$ ) should be a monotonously decreasing function. Some examples are shown in Figure 3.



**Figure 3.** Probability of accepting bad steps ( $d = 0.5, 1, 2$ ).

**Algorithm:** (*Simulated annealing for fairing surfaces*)

```

for  $k = 1 \dots, end\_temp$  do
  do
    for  $r = 1 \dots, max\_step$  do
      choose randomly a knot pair  $kp$ 
      Let  $\tilde{X} := KRR(X)$  be the faired surface after a KRR-step
      if  $\xi_X > \xi_{\tilde{X}}$ 
        then  $X := \tilde{X}$ ; b_count = 1                                /* good step */
      else choose a random number  $0 \leq \nu \leq 1$ 
        if  $\nu < P(t_k)$ 
          then  $X := \tilde{X}$ ; b_count = 1                                /* bad step */
          else b_count += 1                                          /* bad step refused */
        end for
      end for
    until (b_count  $\geq blimit$ )                                     /* no more improvement */
  end for

```

The decreasing sequence of real numbers  $t_1 > t_2 > \dots > t_{max\_step}$  is called *annealing schedule* and controls the probability that bad steps will be done. The  $t_k$  correspond to temperatures in the annealing of metal. More they get smaller, less a bad step will be probable. Once the annealing schedule is fixed, for example as a linear schedule  $t_k = \frac{max\_kp}{k}$ , an appropriate factor  $d$  in the probability function  $P$  has to be chosen (see Figure 3).

At each stage  $k$  the number of good steps is limited by  $max\_step$ , in order to avoid to make too many good steps at the beginning and to get stuck at a local minimum in a false valley. A general suggestion we can make for the choice of the parameter  $max\_step$  is to take it half or less of  $max\_kp$ .

There are two possibilities for the algorithm to stop. In general the algorithm should stop when the making of a bad step was refused  $blimit$  times in succession (due to the decreasing probability of making bad steps). In this case there is no more way out of the current valley. The actual minimum is

accepted as the optimal one. When the algorithm seems to be in an infinite loop, it stops after a maximal number of stages *end\_temp* (final temperature). This should be an emergency stop, and *end\_temp* has to be chosen big enough to allow a regular stop.

The simulated annealing algorithm doesn't guarantee to find a global minimum, but it's well known that it's an effective way to search for one. This algorithm has already been used in CAGD by L. Schumaker [22] for computing optimal triangulations.

#### §4. Results

One fairing step consists of KRR applied at one knot pair. Under all KRR algorithms (Section 2) we favorite the

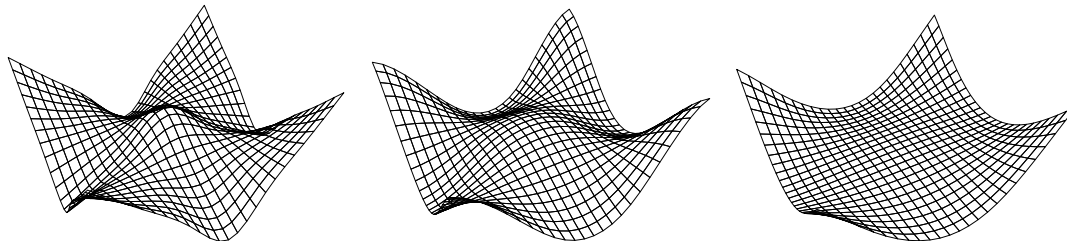
minimal region algorithm (mr-KRR),

where the distance of the old control net from the new one can be relatively high, because only one control point is changed to ensure the  $C^3$ -continuity (in  $u$  or  $v$ -direction) at the selected knot. This leads to big changes in the fairness measure but also in the shape of the surface, and the

least squares algorithm (ls-KRR),

which minimizes the squared distance between the old and new control net by moving three control points. Therefore the shape of the surface doesn't change very hard but the fairing effect isn't as high as with the first knot removal method.

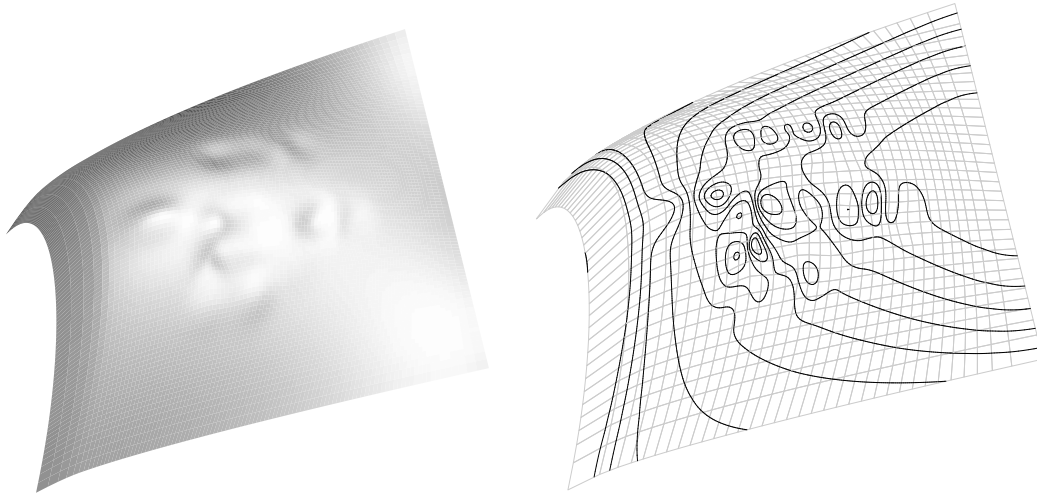
*The user has to decide which aspect he prefers: preservation of the surface shape or maximal fairing effect.*



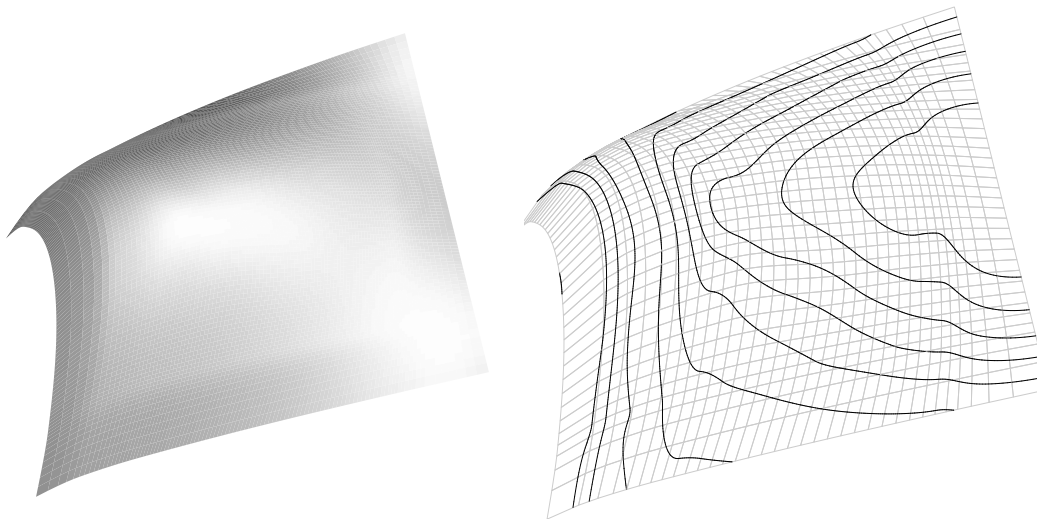
**Figure 4.** Fairing effect of the two different KRR-steps:  
**(a)** unfair surface   **(b)** least-squares KRR   **(c)** minimal-region KRR

The first example illustrate the different fairing effects of these two knot-removal algorithms for surfaces, see Figure 4. The unfair surface (Fig. 4a) has only 25 inner knots. One expects from a fairing method that the high variations of curvature, which occur in the middle of the patch and on the boundary, become smaller or disappear. We then run the simulated annealing fairing (parameters:  $g(u, v) := \kappa_{max}^2 + \kappa_{min}^2$  (3),  $max\_step = blimit = 10$ ) once with the least-squares KRR (32 steps) (Figure 4b) and once with the minimal-region KRR (71 steps) (Figure 4c). As it was assumed above, the

ls-KRR is better adapted for shape preserving fairing, while a more flattening fairing effect is achieved with the mr-KRR.



**Figure 5.** *Unfair surface with isophote analysis.*



**Figure 6.** *Faired surface with isophote analysis.*

If all inner knots are involved in the fairing algorithm, only the four surface corner control points are not modified. But it is also possible to apply the fairing method to a subset of knots. This can be necessary if the boundary curves and tangent planes are not allowed to move, because the surface needs to fit into a whole patchwork of surfaces. Figures 5 and 6 show an example of locally fairing a surface (i.e. the boundary curves will not be affected). The given surface has 256 inner knots. The control points near the border are kept fixed. The simulated annealing with least-squares fairing (ls-KRR) is then applied to only 121 knot in the area, where the surfaces needs to be faired (see the isophote analysis in Figure 5). It can be observed that several runs of the simulated annealing algorithm always stops after  $\sim 400$  steps with

nearly the same result. i.e. a faired surface, as in Figure 6.

For comparison, notice that a systematic search for the global minimum of the fairness measure (after only 10 KRR-steps+ 10 search tree levels) would have needed  $121^{10}$ !!! KRR-steps.

### References

1. Beier, K.-P., Highlight-line algorithm for realtime surface quality assessment, *Computer Aided Design* **26** (1994), 268–277.
2. Bonneau, G.-P., and H. Hagen, Variational design of rational Bézier curves and surfaces, in *Curves and Surfaces in Geometric Design*, P.-J. Laurent, A. Le Méhauté, and L. L. Schumaker (eds.), A. K. Peters, Wellesley MA, 1994, 51–58.
3. De Boor, C., *A Practical Guide to Splines*, Springer, New York, 1978.
4. Brunet, P., Increasing the smoothness of bicubic spline surfaces, *Computer Aided Geometric Design* **2** (1985), 157–164.
5. Eck, M., and J. Hadenfeld, Knot removal for B-spline curves, *Computer Aided Geometric Design* **12** (1995), 259–282.
6. Farin, G., G. Rein, N. Sapidis, and A. J. Worsey, Fairing cubic B-spline curves, *Computer Aided Geometric Design* **4** (1987), 91–103.
7. Ferguson, D. R., P. D. Frank, and K. Jones, Surface shape control using constrained optimization on the B-spline representation, *Computer Aided Geometric Design* **5** (1988), 87–103.
8. Greiner, G., Variational design and fairing of spline surfaces, *Proc. Eurographics 1994*, 143–154.
9. Hadenfeld, J., Local energy fairing of B-spline surfaces, in *Mathematical Methods for Curves and Surfaces*, Morten Dæhlen, Tom Lyche, Larry L. Schumaker (eds.), Vanderbilt University Press, Nashville & London, 1995, 203–212.
10. Hagen, H., and P. Santarelli, Variational design of smooth B-spline surfaces, in *Topics in Surface Modelling*, H. Hagen (ed.), SIAM, 1992, 85–92.
11. Hahmann, S., Visualization techniques for surface analysis, to be published in *Data Visualization Techniques*, C. Bajaj (ed.), John Wiley, 1996.
12. Kirkpatrick S., C. D. Gelatt, M. P. Vecchi, *Science* **220** (1983), 671–677.
13. Kirkpatrick S., Optimization by simulated annealing, *J. Stat. Phys.* **34** (5/6), (1984), 975–987.
14. Kjellander, J. A., Smoothing of bicubic parametric surfaces, *Computer Aided Design* **15** (1983), 289–293.
15. Klass, R., Correction of local irregularities using reflection lines, *Computer Aided Design* **12** (1980), 73–77.
16. Lott, N. J., and D. I. Pullin, Method for fairing B-spline surfaces, *Computer Aided Design* **20** (1988), 597–604.

17. Lyche, T., and K. Morken, Knot removal for parametric B-spline curves and surfaces, *Computer Aided Geometric Design* **4** (1987), 217–230.
18. Moreton, H. P., and C. H. Séquin, Functional optimization for fair surface design, *Comp. Graph.* **26** (1992), 167–176.
19. Poeschl, T., Detecting surface irregularities using isophotes, *Computer Aided Geometric Design* **1** (1984), 163–168.
20. Sapidis, N., and G. Farin, Automatic fairing algorithm for B-spline curves, *Computer Aided Design* **22** (1990), 121–129.
21. Schumaker, L. L., *Spline Functions: Basic Theory*, Wiley, New York, 1981.
22. Schumaker, L. L., Computing optimal triangulations using simulated annealing, *Computer Aided Geometric Design* **10** (1993), 329–345.

Stefanie Hahmann  
Laboratoire LMC-IMAG  
Université INP de Grenoble  
F-38041 Grenoble cedex 9, FRANCE  
[Stefanie.Hahmann@imag.fr](mailto:Stefanie.Hahmann@imag.fr)

Stefan Konz  
Universität Kaiserslautern  
Fachbereich Informatik  
D-67653 Kaiserslautern, GERMANY  
[ko@nemetschek.de](mailto:ko@nemetschek.de)