Fairing by Finite Difference Methods

Leif Kobbelt

Abstract. We propose an efficient and flexible scheme to fairly interpolate or approximate the vertices of a given triangular mesh. Instead of generating a piecewise polynomial representation, our output will be a refined mesh with vertices lying densely on a surface with minimum bending energy. To obtain those, we generalize the finite differences technique to parametric meshes. The use of local parameterizations (charts) makes it possible to cast the minimization of non-linear geometric functionals into solving a sparse linear system. Efficient multi-grid solvers can be applied which leads to fast algorithms that generate surfaces of high quality.

§1. Introduction

Fairing schemes which construct a surface by solving a constrained optimization problem, are traditionally based on piecewise polynomial representations [2,9,16,18]. The major difficulty in this approach is that on one hand efficient (linear) schemes are in general dependent on the specific parameterization and hence fail to be proper models of the physical or geometric intent [7]. On the other hand more sophisticated non-linear optimization is computationally involved and often unstable [15].

The basic idea of the variational design approach is to measure the quality of a surface in terms of its bending energy. The most common functional is the *total curvature*

$$\boldsymbol{E}(\mathcal{S}) \;:=\; \int_{\mathcal{S}} \kappa_1^2 + \kappa_2^2 \;\, d\mathcal{S} \tag{1}$$

which approximates the bending energy of a thin plate. However, since the principal curvatures and the area element depend non-linearly on the surface $\mathcal{S}$, this functional is difficult to minimize. For practical fairing schemes, the total curvature is therefore replaced by the so-called *thin-plate functional*

$$\boldsymbol{E}(F) \;:=\; \int_{\Omega} F_{uu}^2 + 2\,F_{uv}^2 + F_{vv}^2 \;\, du\,dv \tag{2}$$

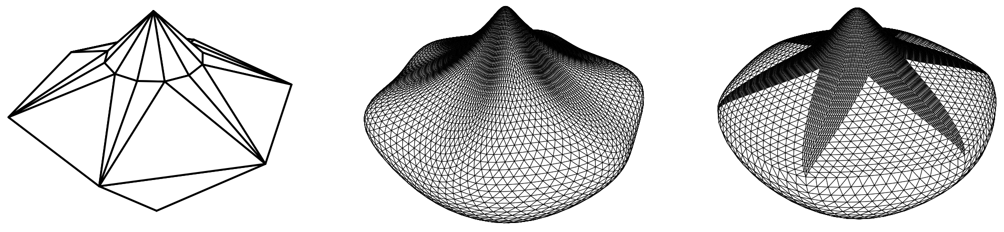

Fig. 1. The triangular mesh on the left is interpolated by surfaces minimizing the thin-plate energy. In the center all patches are parameterized over equilateral triangles; on the right the parameterization is approximately isometric. The shape of the right surface looks 'better' since the true total curvature functional is approximated closer in this case.

which turns out to be identical to the total curvature if the parameterization $F : \Omega \in \mathbb{R}^2 \to \mathcal{S}$ is isometric. Alas, this assumption is far from being satisfied in general. Minimizing (2) instead of (1) hence changes the mathematical model for fairness significantly and the geometric justification for the approach is no longer valid (cf. Fig 1).

Opposed to the *exact* minimization of the *approximate* energy functional is another class of fairing schemes which *approximate* the minimum of the *true* functional by non-linear optimization techniques. Those schemes are mainly based on point evaluation of the integrand function $\kappa_1^2 + \kappa_2^2$ and apply a quadrature formula with respect to an estimated area element [15].

A third way of dealing with the intrinsic non-linearity in the problem of generating fair surfaces is to hide it in the parameterization. In [7] a promising approach to achieve this goal is proposed where the parameterization of the surface $\mathcal{S}$ is defined over a non-planar domain. However, the concept of *data-dependent functionals* still lacks the necessary flexibility to construct surfaces of arbitrary topology since G^k boundary conditions between individual polynomial patches have to be observed.

All the above mentioned approaches are focused on the generation of *spline surfaces.* In this paper I propose a practical (i.e. simple, fast and robust) scheme to compute refined *triangular meshes* with vertices lying densely on a fair surface. There are no topological restrictions as long as the mesh is locally isometric to a disc. The scheme has linear complexity in the number of generated triangles and works completely automatical.

§2. Classical Fairing

In the classical fairing setting an optimal surface is sought in a space spanned by a finite element basis $\{\Phi_i\}$ while maintaining smoothness conditions across the boundaries between elements (e.g., a spline space). There are two major difficulties in this approach:

• A consistent C^k-parameterization of closed surfaces is not possible in general and hence *geometric* continuity conditions are introduced. By this, however, we loose the linear structure of the search space which makes the optimization much more difficult.

• Approximating geometric curvatures by a combination of second order partial derivatives is a rather bad mathematical model (cf. Fig 1). A parameter correcting optimization algorithm could reduce the fairness energy without actually modifying the geometric shape of the surface.

The conditions for interpolation or approximation and for the smooth connection between adjacent patches can be imposed either by directly eliminating degrees of freedom or by Lagrangian multipliers or by some penalty method that includes an additional term which measures the approximation error and the non-smoothness between patches, into the energy functional.

Once a proper basis $\{\Phi_i\}$ for the search space is chosen, the minimization of (2) is a rather simple task: We formally compute the partial derivatives of the objective functional with respect to the coefficients $\boldsymbol{c}_i$ in the expansion

$$\boldsymbol{E}(F) \ := \ \boldsymbol{E}\Big(\sum_i \boldsymbol{c}_i \, \Phi_i\Big).$$

and set them to zero. The solvability of the optimization problem is guaranteed if the kernel of the energy functional (2) and the kernel of the approximation constraint (i.e., the space of functions having the value zero at all approximation sites) have a trivial intersection.

§3. Non-classical Fairing

When generating globally fair surfaces, the goal to achieve well distributed curvature with as few oscillations as possible is much more important than infinitessimal differentiability. For instance the penalty methods [15] exploit the fact that in practical applications, tangent plane continuity is established if the jumps of the normal vectors fall below some ε-threshold.

Non-classical fairing schemes use this observation as a justification for no longer requiring C^k smoothness. Surfaces are approximated by piecewise linear C^0 polyhedra and the shape of these polygonal meshes is optimized. The big advantage provided by those *mesh-smoothing* algorithms is their flexibility with respect to the topology of the surface to be modeled: The C^0 conditions in a triangular mesh are trivially guaranteed.

There are several approaches that apply smoothing operators to meshes in order to improve their fairness. Linear operators have the advantage of being fast and easy to implement [17] but non-linear operators are able to additionally preserve important geometric properties [6,14]. Some schemes increase the fairness of a mesh by changing the positions of the vertices only, others also allow topological changes of the mesh [3,10].

Most of the non-classic schemes are designed to operate on fine meshes where, e.g., noise has to be removed. The task of interpolating a given coarse mesh by a finer one can apparently be reduced to the first problem by taking the original mesh, subdividing all the triangles uniformly to introduce degrees of freedom for the optimization and then applying a smoothing operator. Most of the proposed schemes, however, will perform very poorly on this type of data. The reason for this is the typical low-pass characteristics of local

$$\begin{tikzcd}
 & \int_{\mathcal{S}} \kappa_1^2 + \kappa_2^2 \; d\mathcal{S} \arrow[ddl, "\boxed{\textit{discretize}}"'] \arrow[dr, "\boxed{\textit{linearize}}"] & \\
 & & \int_{\Omega} F_{uu}^2 + 2\,F_{uv}^2 + F_{vv}^2 \; dudv \arrow[dd, "\boxed{\textit{Galerkin}}"] \\
\sum_i \omega_i \,\|\Gamma(\boldsymbol{p}_i)\|^2 \arrow[ddr, "\boxed{\textit{optimize}}"'] & & \\
 & & F :\in \mathrm{span}\{\Phi_i\} \arrow[dl, "\boxed{\textit{sample}}"] \\
 & \triangle - \text{mesh} &
\end{tikzcd}$$

Fig. 2. Different paths lead from the original mathematical model of *fairness* (top) to an approximative solution represented by a triangular mesh (bottom).

smoothing operators. Local high frequency noise is filtered out very quickly but changes of the global shape are propagated very slowly.

§4. The Discrete Fairing Approach

An algorithmic approach to free form surface design are *subdivision surfaces* [5]. Here, a surface is not defined by a map from the parameter domain into $\mathbb{R}^3$ but by a rule how to compute points on [12] or close to [1,4] that surface. Generalizing the concept of knot insertion for B-splines, a subdivision surface is defined by a coarse mesh roughly describing its shape and a rule how to refine the mesh. By applying this rule recursively, we obtain a sequence of finer and finer meshes which converge to a smooth limit surface. In practice the refinement is only repeated until the resulting mesh approximates the final surface up to a prescribed tolerance.

In the discrete fairing algorithm [13] we use the topological aspect of this iterative surface generation paradigm to define a nested sequence of meshes. The actual position of the new vertices when refining a given mesh will be determined by the minimization of a bending energy functional.

Algorithmically we are exactly in the situation described at the end of the last section: We refine a given mesh topologically and then apply a smoothing scheme to improve the fairness. However, we can exploit the fact that iterative refinement generates a sequence of meshes which match the requirements for multi-grid solvers [8]. This accelerates our scheme considerably.

Fig. 2 compares the different approaches to fairing. The crucial step in the discrete fairing approach is to discretize the original continuous functional.

The flexibility of this approach stems form the fact that the requirements for the discretization are much less demanding than the set-up for the Galerkin projection based approaches.

For the discretization step we only need two ingredients. First, we have to replace the surface integral by a quadrature formula, i.e., by a weighted sum of samples of the integrand function

$$\sum_i \omega_i \left(\|\Gamma_{i,uu}(\boldsymbol{p}_i)\|^2 \,+\, 2\,\|\Gamma_{i,uv}(\boldsymbol{p}_i)\|^2 \,+\, \|\Gamma_{i,vv}(\boldsymbol{p}_i)\|^2 \right) \;\approx\; \int_{\mathcal{S}} \kappa_1^2 + \kappa_2^2 \;\, d\mathcal{S}.$$

A natural choice for the sampling sites are the vertices $\boldsymbol{p}_i$ of the mesh (it is here where the discrete curvature is actually located). The weight coefficients ω_i have to reflect the local area element, i.e., the triangles' relative size.

To sample the integrand function at each vertex, we have to compute second order partial derivatives. We do this by applying divided difference operators $\Gamma_{i,*}$ which are constructed by finding a quadratic least squares approximant to the direct neighbors of each vertex with respect to a local parameterization

$$\mu_i \;:\; \boldsymbol{p}_j - \boldsymbol{p}_i \;\mapsto\; (u_j, v_j) \in \mathbb{R}^2.$$

Since we have to evaluate the derivatives at isolated points only, we can find locally isometric parameterizations and exploit the fact that geometric curvatures coincide with a combination of second order derivatives in this case. Constructing a new set of divided difference operators $\Gamma_{i,*}$ for every vertex $\boldsymbol{p}_i$ allows us to choose a different parameterization μ_i for each. Hence, we can compute derivatives (divided differences) always with respect to *local* parameterizations (*charts*). To define those we estimate the tangent plane at each vertex and get the parameter values for the neighbors by orthogonal projection. Another possibility is to approximate the exponential map by assigning parameter values according to the length of the edges and the angles between them [13,19]. Notice that the integrand function in (2) is rotational invariant.

It is tempting to re-estimate the local charts in every step of the iterative optimization. This makes sense given that the current mesh is always the best available approximation to the optimum. However, it turns out that this additional freedom makes the problem severely unstable. In [19] such a smoothing scheme is proposed but the authors have to introduce additional topological operations in order to balance the instability.

Following [13] we want to cast the fairing problem into a simple quadratic optimization problem and we want to find the "best" discrete approximation to the original objective functional. The way to satisfy both goals is to estimate the local charts, i.e., the local metric of the final surface by looking at the given data only. In [13], a weighted average between the discrete exponential map and the uniform parameterization is proposed. This initial estimate is kept fixed during the whole optimization process making the fairing energy functional quadratic with respect to the variables (= vertices).

For special definitions of the local charts observing the subdivision connectivity of iteratively refined meshes, [13] shows that the solution of the optimization problem is uniquely defined, i.e., the fairing functional is strictly positive definite.

We can understand this approach as being *dual piecewise polynomial*. Instead of assigning polynomial patches to the *faces* of a given mesh, we assign them to the *vertices*. This is implicitly done when solving the Vandermonde-system to construct the divided difference operators. For the evaluation of the discrete functional we only have to know these polynomials (and their derivatives) at the vertices but we can assume the virtual existence of a continuous surface consisting of patches around each vertex without having to care about their smooth connections.

§5. Results

The discrete fairing scheme has been implemented based on a modification of a V-cycle multi-grid scheme [8]. Since appropriate stationary subdivision schemes provide smooth (but not necessarily fair) meshes, we don't have to pre-smooth the mesh before going to a coarser level. In fact, is it enough to compute the back-leg of the V-cycle, i.e., we alternate topological subdivision and iterative smoothing.

The special eigenstructure of the Gauß-Seidel iteration matrix in our case implies fast convergence in high frequency sub-spaces but very slow convergence at low frequencies. As a consequence, the mesh is flattened very quickly but converges slowly to the true solution. This effect can sometimes be observed at the original vertices where interpolation constraints are imposed: Occasionally visible cusps remain and are smoothed out rather slowly.

Since exact interpolation in the *interior* of the surface is an artificial constraint anyway, we avoided this problem by letting the original vertices move by some ε in a post-processing step. This removed the "pimples" but did not have significant effect on the global shape.

An important issue for practical applications is the control of the boundaries. As suggested in [12], we use a univariate scheme [11] to generate boundary curves that are independent of the interior of the mesh. This allows not only to join two surfaces along a common C^0 feature line but it also gives the possibility to model sharp feature lines within an otherwise smooth surface. Fig. 3 shows an example of a surface generated by the discrete fairing scheme.

§6. Conclusions and Future Work

I presented a very efficient and robust scheme to generate triangular meshes that smoothly interpolate the vertices of a given mesh. There are no serious restrictions on the topology since no global surface parameterization has to be constructed. All we need are estimates of the local metric at each vertex. Upon this we can construct divided difference operators that allow the approximation of partial derivatives with respect to an isometric parameterization. Together with an estimation of the relative area element we can define a discrete version of the continuous objective function.

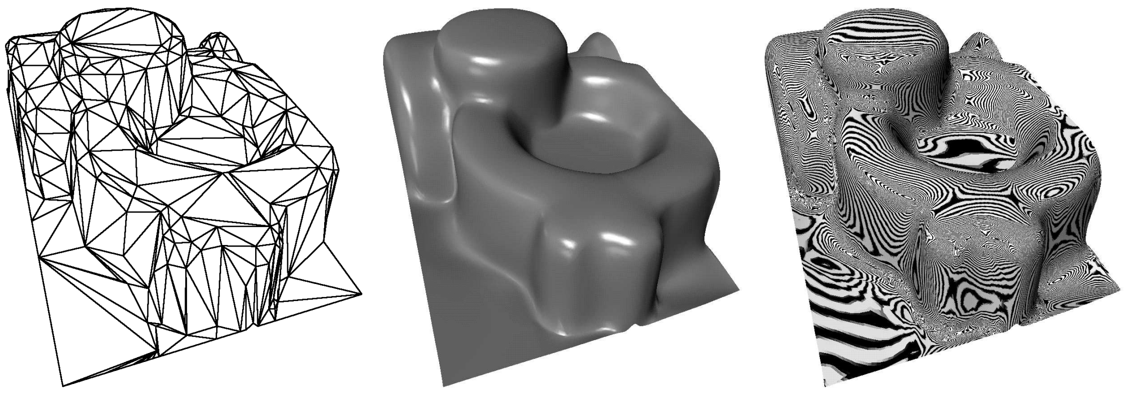

Fig. 3. A fair surface generated by the discrete fairing scheme. The flexibility of the algorithm allows to interpolate rather complex data by high quality surfaces. The process is completely automatical and took about 10 sec to compute the refined mesh with 50K triangles.

Due to the multi-resolution structure of meshes with subdivision connectivity, we can apply fast multi-grid solvers to compute the positions of the vertices in the optimal mesh. Right now, we generate moderately complex models within a few seconds — in the future interactive modeling should become possible.

Future research in this field should address the investigation of higher order fairing functionals since those lead to even better results in some cases. However, higher order partial derivatives require local parameterizations that cover a larger neighborhood around each vertex. This increases the number of topological special cases that have to be considered.

Unlike classical fairing schemes, the discrete scheme does not generate spline surfaces. However, if compatibility to some standard CAD data exchange format matters, the obtained meshes can be converted into a spline representation. This can be achieved by any simple least squares fitting scheme with parameter correction since the duty to establish fairness has already been taken care of by the discrete fairing scheme. The philosophy behind this procedure is to do the fairing by some flexible discrete (non-classic) method and once a sufficient number of points on the optimal surface is computed and the fair shape is recovered, we can use standard spline surfaces for the concluding fitting step.

If the original approximation constraints are considered as local forces that pull or push the surface into the wanted position then the discrete fairing allows to generate many, more densely distributed mini-forces that pull the spline surface more accurately into the optimal position.

References

1. Catmull E. and J. Clark, Recursively generated B-spline surfaces on arbitrary topological meshes, CAD 10 (1978), 350–355

2. Celniker G. and D. Gossard, Deformable curve and surface finite elements for free-form shape design, ACM Computer Graphics 25 (1991), 257–265.
3. van Damme R. and L. Alboul, Tight triangulations, Math. Meth. for Curves and Surfaces, Vanderbilt University Press (1995), 517–526
4. Doo D. and M. Sabin, Behaviour of recursive division surfaces near extraordinary points, CAD 10 (1978), 356–360
5. Dyn N., Subdivision schemes in CAGD, Adv. Num. Anal. II, W.A. Light (ed.), Oxford University Press 1991, 36–104.
6. Dyn N. and D. Levin and D. Liu, Interpolatory convexity-preserving subdivision schemes for curves and surfaces, CAD 24 (1992), 221–216
7. Greiner G., Variational design and fairing of spline surfaces, Computer Graphics Forum 13 (1994), 143–154.
8. Hackbusch W., Multi-Grid Methods and Applications, Springer Verlag 1985, Berlin.
9. Hagen H. and G. Schulze, Automatic smoothing with geometric surface patches, CAGD 4 (1987), 231–235.
10. Hoppe, H. and T. de Rose and T. Duchamp et. al., Mesh optimization, ACM Computer Graphics 27 (1993), 19–26
11. Kobbelt L., A variational approach to subdivision, CAGD 13 (1996), 743–761.
12. Kobbelt L., Interpolatory subdivision on open quadrilateral nets with arbitrary topology, Comp. Graph. Forum 15 (1996), 409–420.
13. Kobbelt L., Discrete fairing, Proceedings of the Seventh IMA Conference on the Mathematics of Surfaces, 1996.
14. le Méhauté A. and F. Utreras, Convexity-preserving interpolatory subdivision, CAGD 11 (1994), 17–37
15. Moreton H. and C. Séquin, Functional optimization for fair surface design, ACM Computer Graphics 26 (1992), 167–176.
16. Sapidis N. (ed.), Designing Fair Curves and Surfaces, SIAM 1994
17. Taubin G., A signal processing approach to fair surface design, ACM Computer Graphics 29 (1995), 351–358
18. Welch W. and A. Witkin, Variational surface modeling, ACM Computer Graphics 26 (1992), 157–166
19. Welch W. and A. Witkin, Free-form shape design using triangulated surfaces, ACM Computer Graphics 28 (1994), 247–256

Leif Kobbelt
Computer Graphics Group
University of Erlangen–Nürnberg
Am Weichselgarten 9
91058 Erlangen, GERMANY
`kobbelt@informatik.uni-erlangen.de`