Surface Reconstruction Based Upon Minimum Norm Networks

Andreas Kolb, Helmut Pottmann and Hans-Peter Seidel

Abstract. In this paper we present a method for surface reconstruction using an extension of the *Minimum Norm Network (MNN)* designed for interpolating (functional) scattered data to the parametric case. Given a polyhedron with triangular faces, we obtain a smooth surface interpolating the vertices of the polyhedron and preserving its topology. As for functional MNN, a curve network is constructed satisfying certain conditions at the vertices from where the curves emanate, and having minimal norm with respect to a certain functional. The G^1-MNN can then be extended to a smooth surface using, for example, methods described by Shirman/Séquin, Peters and Mann. Additionally we give a construction for a G^2 MNN and describe its extension to a smooth surface.

§1. Introduction

In this paper we address the problem of fitting a smooth surface to a given polyhedron with triangular faces. The resulting surface has to interpolate the vertices of the polyhedron and preserve its topology. This problem is well known and it has been studied by many authors within recent years (see [1,3,8,9,10,11,13,15,16,18]). Applications from the automotive industry and other CAD/CAM based technologies that relate to this problem are numerous. Methods that are capable of solving the following problem are of great practical interest:

> **Problem:** Given a polyhedron in 3-space with vertices $\mathbf{V}_i$, $i = 1, \ldots, N$, edges $\mathbf{E}_{ij}$ and triangular faces. The goal is to find a smooth surface that interpolates the $\mathbf{V}_i$'s, preserves the topology of the polyhedron[1] and consists of triangular (polynomial or rational) Bézier patches.

We prefer to use polynomial Bézier patches since they are widely used in CAD systems.

Mathematical Methods in CAGD III
M. Dæhlen, T. Lyche, and L. L. Schumaker (eds.), pp. 1–10.

ISBN xxx.

This is especially relevant when digitized models have to be converted into a CAD format or when known CAD models are locally deformed (e.g. at the points of a regular grid). The basic approach of our method is often used and consists of the following two steps:

1. For each edge $\mathbf{E}_{ij}$, one space curve $\mathbf{C}_{ij}$ is defined such that curves with a common end point have the same tangent plane (or, additionally, agree with a *second fundamental form*; see [6]).
2. The constructed curve network is filled with triangular Bézier patches interpolating the network of curves, such that neighboring patches meet smoothly.

§2. Previous Work

In the following we give an overview of some of the existing methods using the same construction steps. For a more complete survey see [9,10,15].

Some of the methods we present in this section use more than just the polyhedral data as input. Often there is a pre-processing step required in order to estimate the surface normal and/or the second fundamental form at the vertices. The normal is estimated by computing the weighted sum of the polygon normals of all polygons meeting at the considered vertex. The weights are either the sizes of the polygons (or their reciprocal values) or their angles at the vertex. The algorithm in [13] as well as the algorithm presented in this paper may modify these normals during the process of optimization of the curve network. In [11], Moreton and Séquin present an estimator for the second fundamental form using only position data.

2.1. Curve Network Construction

Shirman and Séquin [18]: A surface normal is estimated for each vertex. The curves are defined as Hermite interpolants, where the tangents are calculated by projecting the related edge onto the tangent plane and scaling these by the length of the edge, i.e. by $\|\mathbf{E}_{ij}\|$.

DeBoor, Höllig and Sabin [2]: In a first step surface normals and second fundamental forms are estimated for each vertex. The curves of the curve network are then constructed trying to interpolate position, tangent and curvature information at both end points of a curve $\mathbf{C}_{ij}$ with a cubic polynomial. De Rose and Mann [3] use this method for the construction of a curve network.

Moreton and Séquin [11]: Moreton and Séquin determine the curve network in the following manner: Each curve $\mathbf{C}_{ij}$ leaving the vertex $\mathbf{V}_i$ has to comply with a, yet unknown, surface normal and second fundamental form. These two surface parameters plus any leftover curve parameters not determined by these constraints, are optimized using a functional describing the

[1] more precisely: the final surface can be obtained from the input polyhedron by applying a continuous distortion

variation of curvature of the curves $\mathbf{C}_{ij}$. This optimization process is *non-linear* and computationally very expensive.

Minimum Norm Network [12]: The MNN method is designed to solve the problem in the functional situation, i.e., there exists a plane such that the parallel projection of the polyhedron onto this plane is bijective. The resulting triangulation in the plane is used as global parameterization of the surface. Using this parameterization yields a representation of the curve network by means of the (unknown) partial derivatives of the surface at the vertices. The representation is linearly dependent on these unknowns, thus applying a quadratic functional leads to a linear optimization.

Our intension is to expand the MNN-method to the general problem as described above. We especially want to define a method for linear optimization of curve networks.

2.2. Surface Construction

Shirman and Séquin [18]: For each triangular hole of the defined cubic G^1 network, three total-degree quartic are used to interpolate both the network curves and a cross-boundary vector field computed by the method of Chiyokura and Kimura ([1]). The internal boundaries are constructed by Farin's method in order to guarantee G^1 joints ([4]).

Peters [16]: Peters studies the problem of fitting one Bézier patch to one (triangular or quadrilateral) hole of a given cubic G^1 curve network. The input curves have to be admissible, i.e., the well-known *Vertex Enclosure Problem (VEP)* (see [16]) has to be solvable for each vertex. Careful analysis of Peters' method reveals that the chosen degrees of the scalar weight-polynomials for the smooth surface-surface joints are not high enough to decouple the VEP. Thus, either the weight-polynomials have to be degree-reduced, which results in a surface that is not G^1, or a global linear system has to be solved. In [8], Liu and Sun present a way to overcome this problem in the case of a regular, i.e., a tensor-product like, curve network.

Moreton and Séquin [11]: Based on the above described curve network construction surfaces are fitted to this curve network by optimizing both the variation of curvature of the surface patch and a fairness functional describing the error of smoothness for the surface-surface joints. This optimization is again non-linear. However, the results of these methods are in general very appealing.

DeRose and Mann [3]: This method was originally designed for the approximation of known surfaces, taking the position, surface normal and second fundamental form as input. The curve network is then constructed using the DeBoor, Höllig and Sabin scheme. A cubic triangular Bézier patch is fitted to each hole of the curve network interpolating the curvature at all three related vertices. The resulting surface is, in general, not G^1. DeRose and Mann describe a refinement method to reduce the maximum error of the G^1 surface-surface match.

§3. Optimized Cubic G^1 Network

Converting the principles of MNN construction to the parametric case immediately identifies a major problem of not having a global parameterization. In [13], Nielson suggests a method using local coordinate frames for each vertex. Nielson's representation of the curve network yields 6 unknowns at each vertex for the optimization. Our representation of the curve network makes use of the fact that each regular surface $\mathbf{S}$ can be locally represented in a frame of three orthonormal vectors $\mathcal{F} = \{\hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\mathbf{w}}\}$ with help of a bivariate *scalar* valued function s defined as (see Figure 1)

$$\mathbf{S}(u,v) = u\hat{\mathbf{u}} + v\hat{\mathbf{v}} + s(u,v)\hat{\mathbf{w}}.$$

This results in a representation of the curve network having only 2 unknown parameter at each vertex. An additional benefit is that this representation can be extended to curve networks of higher order.

Let us examine the representation of a curve $\mathbf{C}$ on $\mathbf{S}$ in this local frame $\mathcal{F}$. Locally we can always find a function $\phi : [-d,d] \to \mathbb{R}^2$, $d > 0$ with $\phi(0) = (0,0)$ and $\mathbf{C}(t) = \mathbf{S} \circ \phi(t)$ in a neighborhood of $t = 0$. Thus $\mathbf{C}(0) = V$ and with $\phi'(0) = (a,b)$ we obtain:

$$\mathbf{C}'(0) = a\hat{\mathbf{u}} + b\hat{\mathbf{v}} + (as_u(0,0) + bs_v(0,0))\,\hat{\mathbf{w}}. \tag{2}$$

In our situation, the surface $\mathbf{S}$ is not known. But since the topology of our resulting surface is given by the polyhedron, we can set up local frames $\mathcal{F}_i$ at each vertex $\mathbf{V}_i$ simply by estimating the vector $\hat{\mathbf{w}}_i$ as described at the beginning of section 2. We shall refer to the plane defined by $\hat{\mathbf{w}}_i$ as the *local parameter plane*.

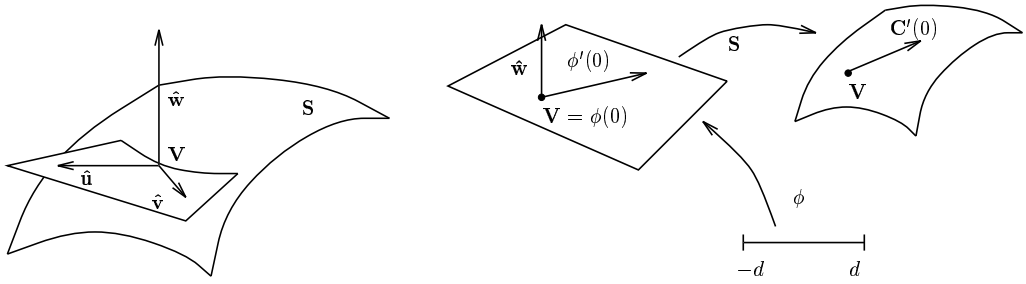

Fig. 1. A local coordinate frame and a curve in this frame.

We now define the cubic curves $\mathbf{C}_{ij}$ of our G^1 curve network as being the Hermite interpolants to the constraints

$$\mathbf{C}_{ij}(0) = \mathbf{V}_i \quad \text{and} \quad \mathbf{C}'_{ij}(0) = \vec{\mathbf{t}}_{ij} + (a_{ij}(s_i)_u + b_{ij}(s_i)_v)\hat{\mathbf{w}}_i,$$

where the vector $\vec{\mathbf{t}}_{ij} = a_{ij}\hat{\mathbf{u}}_i + b_{ij}\hat{\mathbf{v}}_i$ is a *local curve parameter*. Analogous constraints are set up for $t = 1$. After determining the local curve parameters $\vec{\mathbf{t}}_{ij}$ and $\vec{\mathbf{t}}_{ji}$, the curve $\mathbf{C}_{ij}$ depends linearly on the surface data $(s_i)_u, \ldots, (s_j)_v$. Thus, this representation of the curve network is again suitable for linear optimization using a quadratic functional.

3.1. Determination of the Local Curve Parameter

We still have to answer the question of how the local curve parameters $\vec{\mathbf{t}}_{ij}$ must be set. Obviously the setting of these parameters has a great influence on the resulting curve network. The determination of the parameters is split into two steps: first we determine the direction of $\vec{\mathbf{t}}_{ij}$, then we set the length of the vector.

To compute the direction of $\vec{\mathbf{t}}_{ij}$ we use one of the following methods:

D1. Normal Projection: Projection of the edge $\mathbf{E}_{ij}$ onto the plane defined by $\hat{\mathbf{w}}_i$.

D2. Plane Curves: We define a plane containing the edge $\mathbf{E}_{ij}$ and intersect this plane with the local parameter plane at $\mathbf{V}_i$ (see Figure 2).

To set the length of $\vec{\mathbf{t}}_{ij}$ methods can be used as follows:

L1. Chord Length: $\left\|\vec{\mathbf{t}}_{ij}\right\| = \|\mathbf{E}_{ij}\|$.

L2. Circle Segment: The positions $\mathbf{V}_i$, $\mathbf{V}_j$ and the direction $\vec{\mathbf{t}}_{ij}$ define a unique circular segment s; we set $\left\|\vec{\mathbf{t}}_{ij}\right\| = arc(s)$ (see Figure 2).

L3. DeBoor-Höllig-Sabin: Using $\hat{\mathbf{w}}_i$ as normal and estimating the curvature with the Moreton-Séquin estimator (see [11]) a cubic curve $\tilde{\mathbf{C}}$ is fitted to the position, tangent-direction and curvature at the end points; we set $\left\|\vec{\mathbf{t}}_{ij}\right\| = \left\|\tilde{\mathbf{C}}'(0)\right\|$.

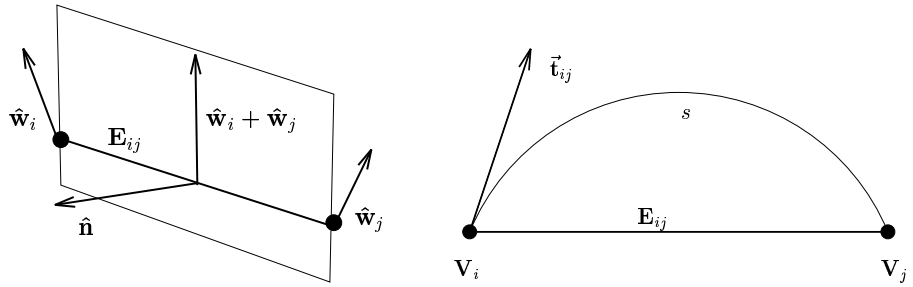

Fig. 2. Curve in a plane and circle segment.

Thus, all local parameter can be determined by any combination of the above described methods.

In the test we made the methods D2 and L2 generally gave the best results.

3.2. Optimizing the Network

As already stated, we want to have a linear optimization for our network, thus only quadratic functionals can be used. Let us first consider a single curve $\mathbf{C}$. A widely used functional is defined as

$$\sigma(\mathbf{C}) = \int_0^1 \|\mathbf{C}''(t)\|^2 \, dt.$$

We refer to this functional as a *simple functional*. The second functional we use is based on the *Laplace Beltrami operator* (see [6])

$$\Delta_{\mathbf{C}_0}(\mathbf{C}) = \frac{\mathbf{C}''}{\langle \mathbf{C}_0' | \mathbf{C}_0' \rangle} - \frac{\langle \mathbf{C}_0' | \mathbf{C}_0'' \rangle \mathbf{C}'}{\langle \mathbf{C}_0' | \mathbf{C}_0' \rangle^2}$$

This operator describes the second derivative of the curve $\mathbf{C}$ with respect to the arc-length of the reference curve $\mathbf{C}_0$. It can be shown that, for $\mathbf{C}_0 = \mathbf{C}$, the curvature vector $\hat{\mathbf{k}}$ of the curve $\mathbf{C}$ can be expressed as $\hat{\mathbf{k}} = \Delta_{\mathbf{C}_0}(\mathbf{C})$. Thus for $\mathbf{C}_0 \approx \mathbf{C}$ the functional

$$\sigma(\mathbf{C}) = \int_0^1 \|\Delta_{\mathbf{C}_0}(\mathbf{C})\|^2 \, ds,$$

where ds denotes integration by arc-length, is a good approximation of the integral over the squared curvature of the curve $\mathbf{C}$. Greiner [5] gives a good overview and more details on the use of quadratic functionals in the area of surface fairing.

Due to the representation of the curve network, applying a quadratic functional has rather the effect of equally distributing the curvature of the curve network than changing the total curvature. Thus the results of applying the simple and the Laplace-Beltrami functional do not, in general, differ very much. In our examples we always use the simple functional.

The functional used to optimize the network is simply the summation of the functionals for each curve. Optimization is then performed by solving a $2N \times 2N$ linear system for the unknown derivatives $(s_i)_u, (s_i)_v$, $i = 0, \ldots, n$. This process can be iterated by using the normals calculated in the optimization as the input normals for the next iteration step. Practice shows that this iteration converges within a small number of iterations.

§4. Filling the G^1 Network

In this section we discuss briefly the methods used to fill the constructed G^1 network with triangular Bézier patches.

One would prefer to have one polynomial patch per facet, but as already mentioned in section 2, a G^1 network can not, in general, be extended to a overall G^1 surface due to the VEP. Thus if a G^1 surface is required, we have to use split schemes. However, for many applications a small error in the G^1 surface-surface joint is acceptable.

We use either Peters' method or the Shirman-Séquin split scheme to fill our G^1 network (see section 2). Peters' method yields one quintic Bézier patch per facet with, in general, some error in the G^1 surface-surface match.

Additionally, we use a modification of the Shirman-Sequin scheme. This scheme fits three total degree quartics to each hole of the curve network. Considering the vertex $\mathbf{V}_i$, the first control point $\mathbf{S}_i$ of the inner *cubic* boundary curve is initially set to the centroid of $\mathbf{S}_i$ and the corresponding control points

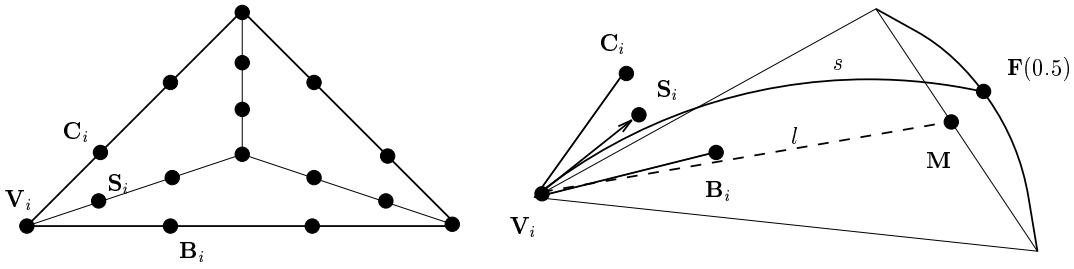

Fig. 3. Setting of the first inner control point.

$\mathbf{B}_i$ and $\mathbf{C}_i$ of the curve network (see Figure 3). This setting works fine in situations where the curve network imposes a nearly plane surface. In cases of strongly bending curves the resulting surface tends to be flat in the center. We correct this in the following way: Let $\mathbf{M}$ denote the mid-point of the triangle edge and $\mathbf{F}$ the curve of the network which are both opposite to vertex $\mathbf{V}_i$. We scale the vector $\mathbf{S}_i - \mathbf{V}_i$ by the fraction $\|s\| \,/\, \|l\|$, where l is the line connecting $\mathbf{V}_i$ and $\mathbf{M}$ and s is the circular segment defined by the positional data $\mathbf{V}_i$, $\mathbf{F}(0.5)$ and the direction $\mathbf{S}_i - \mathbf{V}_i$.

§5. Optimized Quintic G^2 Network

For functional MNN there are extensions that construct C^2 networks (see [7], [17]). These C^2 networks and the resulting interpolants exhibit good quality. Furthermore, Pottmann [17] introduces cross-boundary derivatives into the optimization of the network which leads to a more surface-like network description. This, plus the fact that a G^2 curve network can always be filled with one triangular Bézier patch per facet in order to obtain an overall G^1 surface, motivates the investigations of G^2 networks in the parametric case.

The basic idea of the extension of a G^1 network to a G^2 network is very much the same as for the construction of the G^1 network. Similarly to the case of the first derivative of the curve $\mathbf{C}$, we derive a description for the second derivative by means of the local coordinate frame $\mathcal{F}$:

$$\mathbf{C}''(0) = c\hat{\mathbf{u}} + d\hat{\mathbf{v}} + \left(cs_u + ds_v + (a,b) \begin{pmatrix} s_{uu} & s_{uv} \\ s_{uv} & s_{vv} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \right) \hat{\mathbf{w}},$$

where $\phi''(0) = (c,d)$ and $\phi'(0) = (a,b)$. As was the case for the G^1 network, the local curve parameter $\vec{\mathbf{t}} = a\hat{\mathbf{u}} + b\hat{\mathbf{v}}$ can be determined using the method of section 3.1. The additional curve parameter $\vec{\mathbf{s}} = c\hat{\mathbf{u}} + d\hat{\mathbf{v}}$ for the second derivative can either be set to $\mathbf{0}$, or is computed in parallel with $\vec{\mathbf{t}}$ when using the method L3 or L4. In the case of (L4) we also vary the direction of $\vec{\mathbf{s}}$. Note that in the case of $c = d = 0$, $\hat{\mathbf{w}}$ lies in the osculating plane of the curve at $t = 0$. This implies that, for $s_u = s_v = 0$, $\mathbf{C}$ is locally geodetic, i.e., the resulting surface normal lies in the osculating plane of $\mathbf{C}$.

To introduce a cross-boundary vector field, we observe that, for a tangent vector $\vec{\mathbf{h}} = \overline{a}\hat{\mathbf{u}} + \overline{b}\hat{\mathbf{v}} + \left(\overline{a}s_u + \overline{b}s_v\right)\hat{\mathbf{w}}$ at $\mathbf{V}$, the $\vec{\mathbf{w}}$-component of the derivative

of a cross-boundary vector-field $\vec{\mathbf{d}}(t)$ with $\vec{\mathbf{d}}(0) = \vec{\mathbf{h}}$ in direction of $\vec{\mathbf{t}}$ is given by

$$\left\langle \frac{\partial \vec{\mathbf{d}}}{\partial \vec{\mathbf{t}}}(0) \mid \hat{\mathbf{w}} \right\rangle = (a, b) \begin{pmatrix} s_{uu} & s_{uv} \\ s_{uv} & s_{vv} \end{pmatrix} \begin{pmatrix} \overline{a} \\ \overline{b} \end{pmatrix},$$

if the $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ components of $\partial \vec{\mathbf{d}} / \partial \vec{\mathbf{t}}(0)$ are set to 0.

We fix $\vec{\mathbf{h}}$ either orthogonal to $\vec{\mathbf{t}}$ or as the intersection between the local parameter plane and the plane perpendicular to the corresponding edge $\mathbf{E}$. The cross-boundary vector-field $\vec{\mathbf{d}}$ is now defined as cubic Hermite interpolant to the end constraints $\vec{\mathbf{h}}$, to $\partial \vec{\mathbf{d}} / \partial \vec{\mathbf{t}}$ at this vertex and to the analogous set of data at the opposite vertex of edge $\mathbf{E}$. This description is again linearly dependent on the surface data $s_u, \ldots, s_{vv}$.

The resulting network is optimized using the quadratic functionals described in section 3.2. As a result of this representation, the optimization can again be realized in a reasonable amount of time by solving a $5N \times 5N$ linear system in the unknown first and second derivatives at the vertices.

§6. Filling the G^2 Network

In this situation not only the curve network is given but also the optimized cross-boundary vector field which we must interpolate. One of the following methods may be employed:

Modified Shirman-Séquin: This simple variation of the Shirman-Séquin method uses three degree six triangular Bézier patches per facet. The cubic cross-boundary vector field is interpolated with a method similar to the Chiyokura-Kimura scheme (see [1]).

Single Patch: As was mentioned before, a G^2 curve network can always be extended to an overall G^1 surface using one Bézier patch per facet. The degree of a patch interpolating the curve network and the cross-boundary vector field has to be seven. We remark that this single patch approach may fail in certain extreme situations, i.e., the resulting patch may have singular points at the boundary. In such a situation we use the modified Shirman-Séquin method to guarantee the regularity of the resulting surface.

§7. Examples and Applications

We give two simple examples and two applications based on realistic problems.

The Shifted Octahedron: A frequently used example is the reconstruction of a sphere from a regular octahedron. We tighten this example by moving one vertex out of position but preserve its radial distance from the origin (see Figure 4).

The Boundary Problem: For a simple curve network construction scheme, the normal estimation is the crucial point. All normal estimation techniques

exhibit problems at the boundaries of open polyhedrons, where only few polygon normals contribute to the normal estimation (see Figure 5).

The Car Part: This first application is based on the following problem: A part of a car is designed with a commercial CAD system. The part is further processed by using a local deformation. This deformation is simulated on a regular grid of points. To make the result of the simulation again available for the CAD system, we use some data reduction to get rid of redundancies and reconstruct the surface (see Figure 6).

The Machinery Part: Here we use our methods for design purposes. In this case, the input data consists of a set of curves that should be approximated by a surface. A simple pre-processing step is used to digitize the curves and build a set of polygonal meshes in order to define a polyhedron. In the curve network construction we add the constraint that all curves connecting the points of one input curve should result in a overall G^1 curve interpolating these points (see Figure 7a and 7b).

To illustrate the quality of the surfaces that are produced by the different surface reconstruction schemes, we use color curvature plots to display the *Gaussian* curvature (see [6]).

The color at a surface point is obtained by linearly mapping a predefined interval of curvature values to a range of colors going from red (minimal curvature) to blue (maximal curvature).

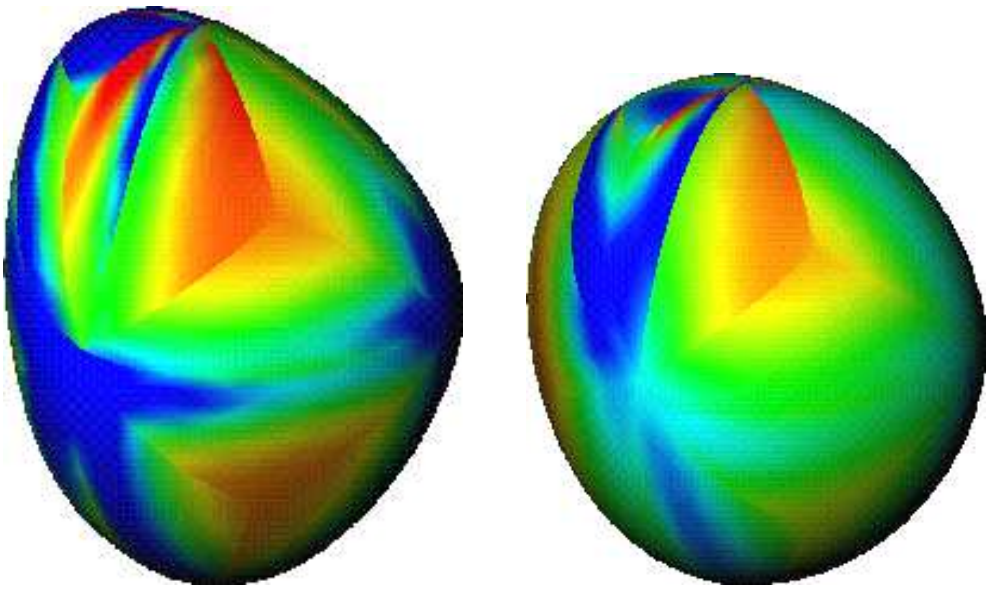

Fig. 4. The Shifted Octahedron: Surface reconstruction using the Shirman-Séquin method (left) and using the G^1-MNN based approach (right); the local parameters are set using methods D2 and L2 and the network is filled using the modified Shirman-Séquin split scheme.

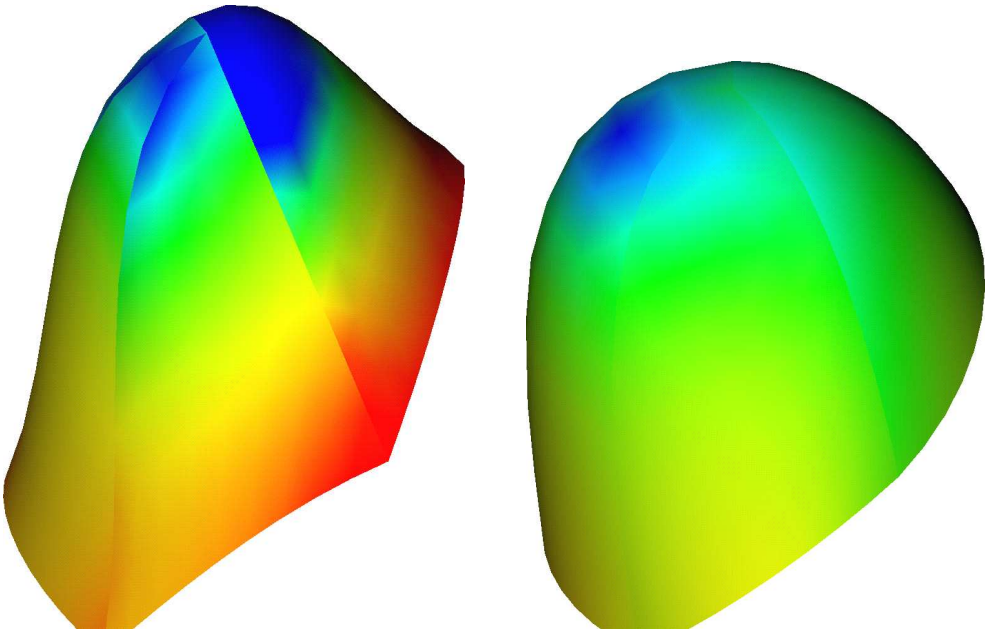

Fig. 5. The Boundary Problem: The curve network constructed using the Shirman-Séquin method (left) and the G^1 MNN scheme using methods D2, L2 (right); in both cases Peters' method is used to fill the network with G^1 continuity, since the VEP is solvable at the inner vertex.

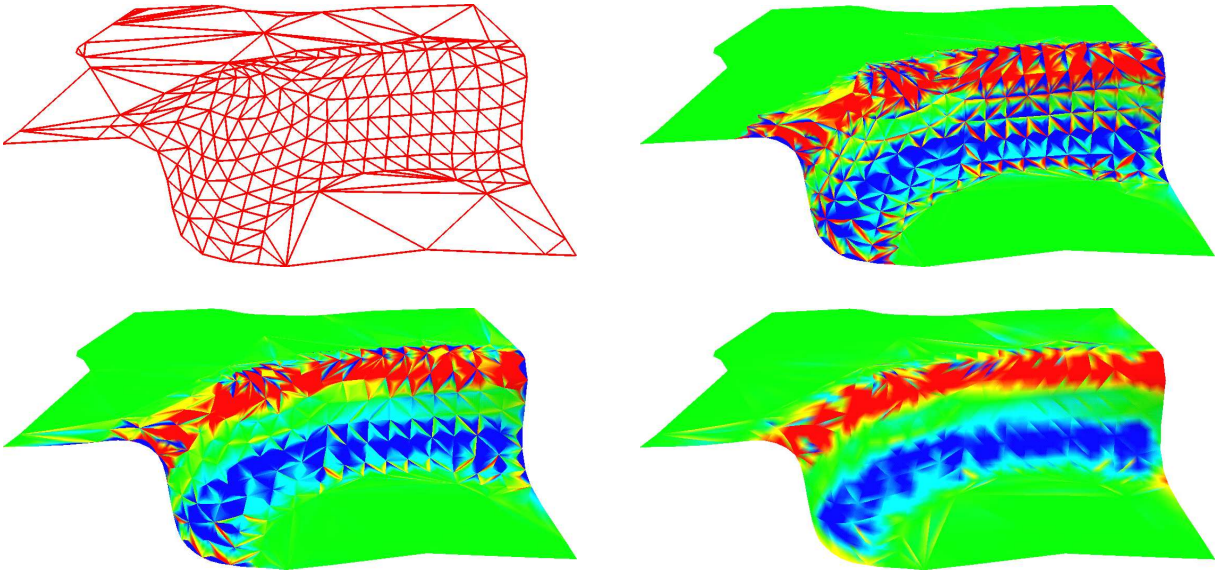

Fig. 6. The Car Part: The input polyhedron (upper left), the surface reconstructed using Shirman-Séquin's method (upper right) and using the G^1 and the G^2 MNN schemes (lower left and lower right, respectively); in both cases the D2, L2 methods are used to construct the MNN; the G^1 and G^2 curve networks are extended using the modified Shirman-Séquin method and the single patch scheme, respectively.

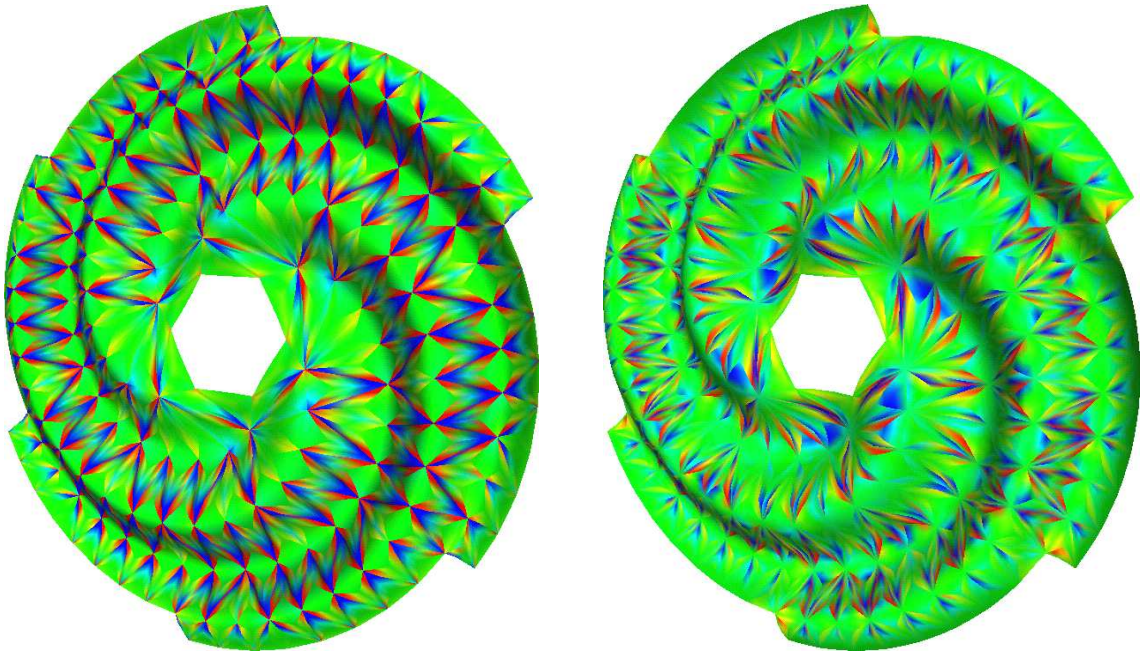

Fig. 7a. The Machinery Part: The machinery part designed using the Shirman-Séquin method (left) and using the G^2 MNN based surface reconstruction scheme using the D2, L2 methods to set the local curve parameters (right); the G^2 MNN is filled using the modified Shirman-Séquin split scheme.

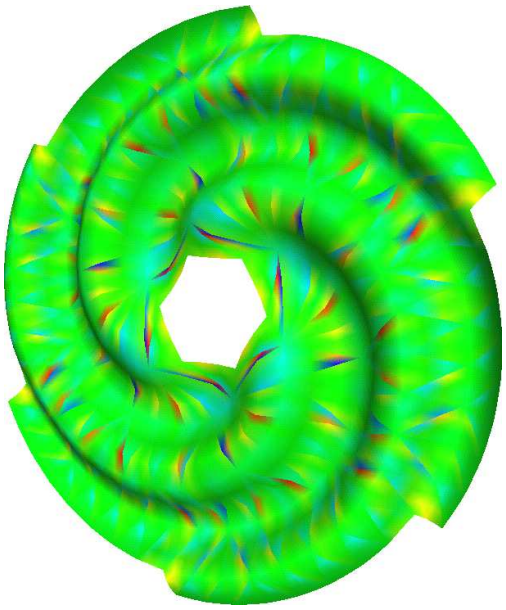

Fig. 7b. The Machinery Part: The machinery part designed using the G^2 MNN scheme and the D2, L2 method; the network is filled using the single patch scheme.

Even though we significantly improve the overall surface of the machinery part, the sequence of curvature plots reveals that an approach based on curve networks is not appropriate for this kind of problem.

References

1. Chiyokura, H., Localized Surface Interpolation Method for Irregular Meshes, *Advanced Computer Graphics*, (1986), 3–19.
2. De Boor,C., Höllig, K. and Sabin, M., High Accuracy Geometric Hermite Interpolation, *CAGD*, **4**, (1987), 269–278.
3. DeRose, T. and Mann, S., An approximatly G^1 cubic surface interpolant, *Math. Methods in CAGD II*, T. Lyche and L.L. Schumaker (eds), Academic Press, (1992).
4. Farin, G., A Construction for Visual Continuity of Polynomial Surface Patches, *Computer Graphics and Image Processing*, **20**, (1982), 272–282.
5. Greiner, G., Surface Construction Based upon Variational Principles, P.J. Laurent, A. LeMéhauté and L.L. Schumaker (eds), *Curves and Surfaces II*, (1994), AKPeters.
6. Klingenberg, W., A Course in Differential Geometry, Springer-Verlag, Berlin-Heidelberg, (1978).
7. Kolb, A. and Seidel, H.-P., Interpolating Scattered Data with C^2 Surfaces, *CAD*, to appear.
8. Liu, Q. and Sun, T.C., G^1 Interpolation of Mesh Curves, *CAD*, **26** (4), (1993), 259–267.
9. Lounsbery, M., Mann, S. and DeRose, T., Parametric surface interpolation, *IEEE Computer Graphics and Applications*, **12**, (1992), 45–52.
10. Mann, S., Surface approximation using geometric Hermite patches, PhD thesis, University of Washington, (1992).
11. Moreton, H.P. and Séquin, C.H., Functional Optimization for Fair Surface Design, *ACM Computer Graphics*, **26**, (1992), 167–176.
12. Nielson, G.M., A Method for Interpolating Scattered Data Based Upon a Minimum Norm Network, *Math. Comp.*, **40** (1983), 253–271.
13. Nielson, G.M., Interactive Surface Design using Triangular Network Splines, Proceedings 3rd Int. Conf. on Engineering Graphics and Descriptive Geometry, Vol. 2, Vienna, (1988), 70–77.
14. Nielson, G.M. and Franke, R., A Method for Construction of Surfaces Under Tension, *Rocky Mountain J. Math.*, **14**, (1984), 203–221.
15. Peters, J., Local Smooth Surface Interpolation: A Classification, *CAGD*, **7**, (1990), 191–195.
16. Peters, J., Smooth Interpolation of a Mesh of Curves, *Constr. Approx.*, **7**, (1991), 221–246.
17. Pottmann, H., Scattered Data Interpolating Based upon Generalized Minimum Norm Networks, *Constr. Approx.*, **7**, (1991), 247–256.
18. Shirman, L.A. and Séquin, C.H., Local Surface Interpolation with Bézier Patches, *CAGD*, **4**, (1987), 279–295.
19. Shirman, L.A. and Séquin, C.H., Local Surface Interpolation with Bézier Patches: Errata and Improvements, *CAGD*, **8**, (1991), 217–221.

Acknowledgements. The authors wish to thank Dr. H. Boerger (*com_posite consultants*) and S. Karbacher (Univ. of. Erlangen) who made the car-part data available. Furthermore R. Heimüller (Univ. of. Erlangen), the engineer and designer of the machinery part, contributed to our experiments.

Andreas Kolb, Hans-Peter Seidel
Computer Science Department
University of Erlangen
AM Weichselgarten 9
D-91058 Erlangen
{kolb,seidel}@informatik.uni-erlangen.de

Helmut Pottmann
Institut für Geometrie
Technische Universität Wien
Wiedner Hauptstr. 8-10
A-1040 Wien
pottmann@egmvs2.una.ac.at