Polynomial Surfaces Interpolating Arbitrary Triangulations

Stefanie Hahmann and Georges-Pierre Bonneau

Abstract—Triangular Bézier patches are an important tool for defining smooth surfaces over arbitrary triangular meshes. The previously introduced 4-split method interpolates the vertices of a 2-manifold triangle mesh by a set of tangent plane continuous triangular Bézier patches of degree five. The resulting surface has an explicit closed form representation and is defined locally. In this paper, we introduce a new method for visually smooth interpolation of arbitrary triangle meshes based on a regular 4-split of the domain triangles. Ensuring tangent plane continuity of the surface is not enough for producing an overall fair shape. Interpolation of irregular control-polygons, be that in 1D or in 2D, often yields unwanted undulations. Note that this undulation problem is not particular to parametric interpolation, but also occurs with interpolatory subdivision surfaces. Our new method avoids unwanted undulations by relaxing the constraint of the first derivatives at the input mesh vertices: The tangent directions of the boundary curves at the mesh vertices are now completely free. Irregular triangulations can be handled much better in the sense that unwanted undulations due to flat triangles in the mesh are now avoided.

Index Terms—Triangulation, irregular 3D meshes, arbitrary topology, modeling, surfaces, triangular patches, piecewise polynomial patches, interpolation, arbitrary tangent vectors, reconstruction.

1 Introduction

The easiest way of modeling free-form surfaces is to use tensor-product Bézier/BSpline or NURBS patches. Tensor-product Nurbs patches have long ago become a de facto standard in the CAD/CAM industry. But, tensor-product patches are able to model only a very restricted type of surface, those for which the topological type is the same as that of a square. Unfortunately, 2-manifold surfaces with arbitrary topological type are very common in everyday life. For example, a cup of coffee has the topological type of a torus. Modeling the skin of a cup of coffee as a single smooth surface is not an easy task. If you are restricted to tensor-product patches, most probably you would use two pieces: one for the container and one for the handle. You would then trim the container along the joint between the two pieces. You would carefully modify the control points of the handle in order to blend the two pieces as smoothly as possible. Trimming is the common method used for modeling surfaces of arbitrary topological type with tensor-product patches. But, dealing with trimmed models is rather cumbersome. These reasons explain why researchers have tried to develop mathematical models that can deal with control-polygons of arbitrary topological type, not only tensor-product control-polygons.

Two different research directions have been pursued. One is based on subdivision surfaces that recursively subdivide the control-mesh. The other direction consists of building a patchwork of smoothly joined parametric patches, with the same topology as the control-polygon. The present paper deals with this last kind of surface. Not surprisingly, the two directions have encountered the same main difficulty: dealing with the smoothness of the surface. Here, we ensure tangent-plane continuity of the resulting surface. Our method builds a network of triangular Bézier patches that *interpolates* a given 2-manifold triangular mesh with arbitrary topological type. Interpolation is a very useful and intuitive feature in modeling. But, unfortunately, it is also quite tricky to find smooth curves or surfaces that interpolate a given control-polygon. In particular, if the control-polygon has nonregular features, one small edge joined to a long edge in the 1D case, or one very flat triangle joined with a big triangle in the 2D case, then, most probably, the interpolating curve or surface will suffer from severe undulations in this area. Note that this problem is not particular to the parametric methods. The methods based on subdivision surfaces have exactly the same problem when it comes to interpolating control-polygons.

In this paper, we introduce a new interpolation method that avoids undulations, even when interpolating irregular triangulations. Each input triangle is regularly subdivided into four subtriangles and one degree 5 Bézier patch is associated to each of the subtriangles. These four Bézier patches are referred to as a macropatch. Inside a macropatch, the four Bézier patches are connected with C^1 continuity. The macropatches are themselves connected with G^1 continuity. The key difference with our previously introduced 4-split [12] method lies in the fact that our new interpolant allows *free choice of all first derivatives* at each input vertex, along each input edge. Whereas the previous method was restricted to having first derivatives that form an affine transformation of a regular n-gone, this new method does not impose any constraint on the first derivatives. This allows us to avoid any unwanted undulations when interpolating irregular triangulations.

- *S. Hahmann is with the Laboratoire LMC-IMAG, University of Grenoble, B.P. 53, F-38041 Grenoble cedex 9, France. E-mail: Stefanie.Hahmann@imag.fr.*
- *G.-P. Bonneau is with the Laboratoire iMAGIS-GRAVIR, INRIA Rhône-Alpes, 655, avenue de l'Europe, F-38330 Montbonnot, France. E-mail: Georges-Pierre.Bonneau@imag.fr.*

Manuscript received 8 June 2001; revised 14 Sept. 2001; accepted 19 Sept. 2001.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number 114326.

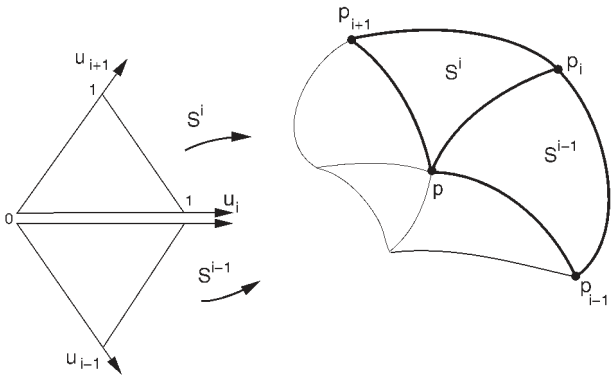

Fig. 1. Parameterization.

The paper is organized as follows: After some notations and some basics about tangent plane continuity, the presentation of previous related works will then outline the main difference to our method which we believe is able to increase the general quality of such interpolating spline surfaces. Then, the description of the algorithm starts (Section 3), which is mainly composed of three steps. The first two steps (Sections 3.2-3.4) are coupled. They consist of constructing a curve network interpolating the mesh vertices and of constructing the tangent ribbons along that curves in order to satisfy the G^1 conditions between adjacent patches. Step 3 (Section 3.5) finally counts the remaining Bézier control points of the patches and explains how to calculate them. We then go into detail of some design and computational aspects (Section 4) of that method. Section 5 will present results and compare them to earlier works. The concluding remarks finally indicate some directions for further research.

2 Basics and Related Works

2.1 Notations

The problem we address is to find a parametric polynomial surface which interpolates a given triangulated surface mesh $\mathcal{M}$ with tangent plane continuity. The triangle mesh can be of arbitrary topology, but should be 2-manifold. There is no restriction on the valence of its vertices. It can be an open or a closed mesh. The requirements on the surface are to:

- be polynomial of as low degree as possible,
- interpolate the mesh vertices,
- be defined locally, and
- be smooth in the sense of having a pleasing shape.

The smoothness requirement is a global one and it conflicts therefore with a local definition of the surface. We therefore try to get an overall well-shaped surface only by doing local operations.

The general procedure consists of constructing parametric patches in one-to-one correspondence to the mesh faces. Each patch boundary curve corresponds to an edge of $\mathcal{M}$ and interpolates both end points. Each patch is, in our case, a polynomial image of the unit domain triangle. The polynomial patches and curves are represented in Bernstein-Bézier basis. A curve of degree n with control-points $\boldsymbol{b}_0, \cdots, \boldsymbol{b}_n$ has the parametric equation $B(t) = \sum_{i=0}^{n} \boldsymbol{b}_i t^i (1-t)^{n-i}$. A triangular patch of degree n with control-points $\boldsymbol{b}_{(i,j,k),i+j+k=n}$ has the parametric equation

$$S(s,t) = \sum_{(i,j,k),i+j+k=n} \boldsymbol{b}_{(i,j,k)} s^i t^j (1-s-t)^k.$$

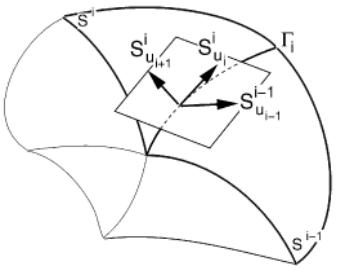

Fig. 2. Tangent plane continuity.

The patches of the surface we aim to construct are defined locally around the vertices which they interpolate. We adopt therefore the following parameterization: Let us consider a mesh vertex, $\boldsymbol{p} \in \mathbb{R}^3$, and its neighborhood points, $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n$, ordered in a trigonometric sense. The integer n is called the *valence of* $\boldsymbol{p}$. The patches around the vertex are numbered S^1 to S^n. Let $u_i \in [0,1]$ be the parameter corresponding to the curve between $\boldsymbol{p}$ and $\boldsymbol{p}_i$, the patch S^i is then parametrized as shown in Fig. 1. In the following, we denote $\Gamma_i(u_i)$ the *patch boundary curve* joining $\boldsymbol{p}$ to $\boldsymbol{p}_i$

$$\Gamma_i(u_i) \;:=\; S^i(u_i, 0) = S^{i-1}(0, u_i) \qquad\qquad u_i \in [0,1].$$

2.2 Tangent Plane Continuity

We require our interpolating surface to be tangent plane, e.g., G^1-continuous. Tangent plane continuity doesn't depend on parameterization as C^1 continuity does. Tangent plane continuity is the mostly used definition of first order continuity for free-form parametric surfaces in CAGD. In addition to being parameter-independent, it also allows more free parameters in comparison to the C^1 continuity.

The surface should be G^1 continuous which means to have continuously varying tangent planes between the patches. The G^1 conditions are at the origin of every step in the surface construction algorithm. They have to be satisfied between adjacent patches and at the mesh vertices where an arbitrary number of patches meet. They therefore have to be satisfied by the boundary curves at the mesh vertices and by the cross-boundary tangents.

Let S^i and S^{i-1} be two adjacent patches sharing a common boundary curve $\Gamma_i(u_i)$. They meet with tangent plane continuity if there exist three scalar functions Φ_i, μ_i, ν_i such that

$$\Phi_i(u_i) \frac{\partial S^i}{\partial u_i}(u_i, 0) = \mu_i(u_i) \frac{\partial S^i}{\partial u_{i+1}}(u_i, 0) + \nu_i(u_i) \frac{\partial S^{i-1}}{\partial u_{i-1}}(0, u_i), \tag{1}$$

where $\mu_i(u_i)\nu_i(u_i) > 0$ (preservation of orientation) and $\frac{\partial S^i}{\partial u_{i+1}}(u_i, 0) \times \frac{\partial S^{i-1}}{\partial u_{i-1}}(0, u_i) \neq 0$ (well-defined normal vectors). This formula means that the three partial derivatives along the boundary curve Γ_i are always coplanar, see Fig. 2.

The goal is now to determine, for each mesh vertex of valence n, the scalar valued functions Φ_i, μ_i, ν_i and the patches S^i, $i = 1, \ldots, n$. Of course, there are an infinite number of solutions. We require the patches to be polynomial with all the advantages they have with respect to rational patches. Also, we require them to be of minimal possible degree and to be defined explicitly and locally, meaning that we don't want to have to solve any global linear system in order to compute them.

2.3 Related Works

The earliest interpolation schemes are due to Piper [22], Shirman and Séquin [24], and some others [14], [17] which are all based on the same idea using a Clough-Tocher domain split. The initial work to that can be found in [6]. They all apply to arbitrary triangle meshes and result in three polynomial patches of degree four per mesh face. The problem of polynomial G^1 interpolation is that the surfaces have to be twist compatible at the vertices. The problem is solved by splitting the domain triangle Clough-Tocher-like into three triangles by introducing a new point at the interior. Due to the very low degree, only a few degrees of freedom are available for shape control. This can be the reason for what Mann et al. found out in their survey paper [17]. They compared polynomial and rational triangular interpolation schemes and conclude that all of them suffer from shape defects.

Then, Loop [15] developed another polynomial scheme which results in only one patch for a mesh face. Its low degree six is obtained by choosing a special setting for the scalar functions Φ_i, μ_i, ν_i in (1), namely:

$$\begin{aligned} Loop'94: \quad & \Phi_i \text{ function of degree } 2 \\ & \mu_i = \nu_i \equiv \tfrac{1}{2}, \end{aligned}$$

in order to solve the twist problem. This particular choice entirely fixes the patch boundary curves up to a scalar degree of freedom per vertex. He obtains, however, quite fair surfaces by relaxing the interpolation condition. It therefore does not really belong to the class of schemes we consider here, but the way he solved the twist problem motivated our work.

A third type of methods is the basic 4-split method [12].

$$\begin{aligned} Hahmann/Bonneau'00: \quad & \Phi_i \text{ piecewise function of degree } 1 \\ & \mu_i = \nu_i \equiv \tfrac{1}{2}. \end{aligned}$$

By taking Loop's low degree setting for the scalar functions, μ_i, ν_i, and by lowering the degree of Φ_i by one and by introducing piecewise polynomial boundary curves and cross-boundary tangents, we obtained enough degrees of freedom for the curve network construction. The four patches (due to the domain triangle 4-split) are of degree five and interpolate the mesh vertices.

It turns out that this particular setting of $\mu_i = \nu_i = const$ can lead to "misbehavior" of the surface, to unpleasant shapes if the input mesh contains irregularities. Irregularities mean, for example, that very short and very long edges meet at the same vertex or that small flat triangles are joint to more equiangular triangles. All these situations in an input mesh make it extremely difficult to find an interpolating smooth surface. The same problem also occurs for interpolatory subdivision surfaces; they are also not able to result in a fair interpolating surface in that case.

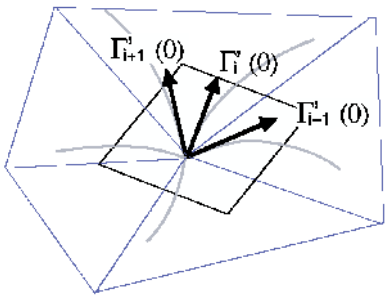

Fig. 3. Tangent plane continuity at mesh vertex.

One solution to that problem can consist of a kind of preprocessing step of the input mesh by trying to optimize the mesh with respect to one well-chosen cost function. However, in this case, interpolation doesn't make sense anymore.

What we want to develop in the present paper is a new mesh interpolation method which allows for *arbitrary values of the scalar functions* around a vertex, while maintaining, at the same time, the interpolation scheme polynomial of degree five. The functions will be linear. What the difference between using constant or arbitrary linear functions really is will be explained in detail in the following section.

Other triangular interpolants exist including the convex combination schemes [8], [9], [11], [19], boundary curve schemes [21], algebraic methods [2], singular parameterization [18], quasi G^1 surfaces [16], and methods for meshes with restricted vertex valences [25].

3 The G^1 Interpolation Scheme

The algorithm of the present method consists mainly of three constructive steps:

1. boundary curve network,
2. cross-boundary tangents,
3. fill-in patches.

The first two steps are linked together because of the G^1 conditions. First, we examine in detail the G^1 conditions at a mesh vertex. We get first and second derivatives of the patch boundary curves at the vertices. These data are then used to find a curve network interpolating these data and which is therefore G^1 compatible.

3.1 Tangent Plane Continuity at Mesh Vertices

The tangent plane continuity conditions at a mesh vertex of valence n consist of the n equations (1) between the n patches meeting at that vertex evaluated at the parameters $u_i = 0$:

$$\Phi_i(0)\,\Gamma'_i(0) \;=\; \mu_i(0)\,\Gamma'_{i+1}(0) \;+\; \nu_i(0)\,\Gamma'_{i-1}(0), \quad i = 1, \ldots, n. \tag{2}$$

The indices are taken modulo n. It turns out that, at the vertex, the three partial derivatives in (1) are identical to the first derivatives of the three boundary curves Γ_{i-1}, Γ_i, and Γ_{i+1}, see Fig. 3.

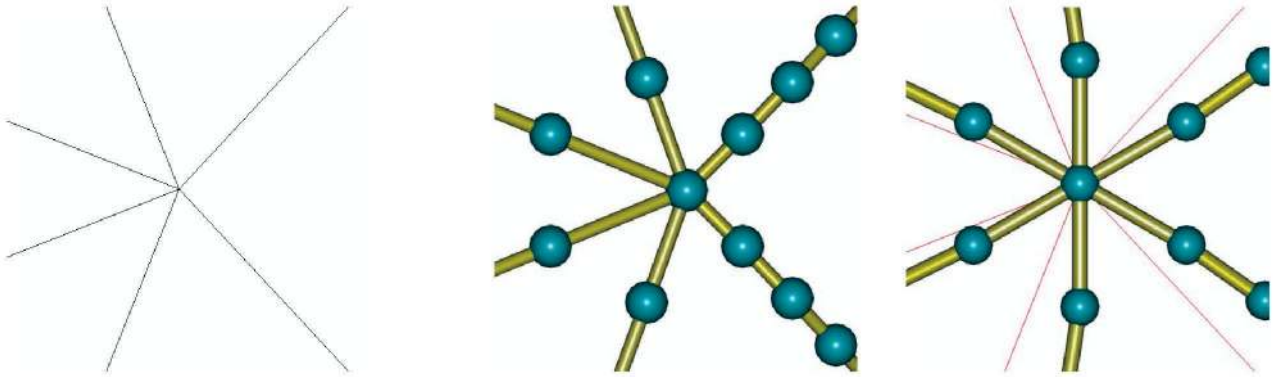

Fig. 4. Left: Zoom on a 3D vertex configuration with six common edges. Middle: Arbitrary tangents at mesh vertex. The vertex together with the curves' Bézier control points build the polygons. Right: Tangent Bézier control points corresponding to a constant value of the scalar functions, in particular, $\mu_i(0) = \nu_i(0) \equiv \frac{1}{2}$.

Equation (2) therefore relates the values of the scalar functions around a vertex to the first derivatives of the boundary curves. From the general G^1 condition, we already know that the degree of the patches is related to the degree of the scalar functions. If one wants to get patches of as low degree as possible, one has to keep the degree of these functions as low as possible. This was done in [15], [12], where $\mu_i = \nu_i = const$. It was shown in these papers that, in the case of $\mu_i = \nu_i = const$, the first derivatives in terms of Bézier control points always form an affine transformation of a regular planar n-gon. This leads to very restricted positions of the derivative vectors around a vertex. There is not enough freedom for choosing the derivatives of the curves at the vertex. Indeed, in order to get smooth surfaces it is very important first to build smooth boundary curves, as was pointed out by Mann et al. in [17], and, hence, to be able to choose the first derivatives with as most freedom as possible. Fig. 4 (left) shows a top-view zoom on the real example given later in Figs. 14, 15, and 16 (right), centered on the order six vertex. The tangents corresponding to $\mu_i = \nu_i \equiv const$ are shown in Fig. 4 (right). Because of the restriction on their choice, these tangents lead to boundary curves with undulations and, thus, to unpleasant shapes, as can been seen in Fig. 15 (right). On the contrary, our new method enables to choose arbitrary tangents, as shown in Fig. 4 (middle) and, thus, to remove boundary curve undulations, as shown in Fig. 16 (right).

Conclusion. The setting of $\mu_i = \nu_i \equiv const$ doesn't allow for an arbitrary choice of the boundary curve's derivatives at the mesh vertices. However, it works well if the input mesh is almost regular in the sense of almost equiangular triangles. A preprocessing mesh optimization would make sense if interpolation of the control mesh is not required.

3.2 Collecting Data Ensuring G^1 Continuity at Mesh Vertices

Tangents. For simplification, let us first introduce the following notation for the tangents (first derivatives) of the boundary curves at a mesh vertex:

$$\boldsymbol{d}_i^1 \;:=\; \Gamma_i'(0) \;=\; \frac{\partial S^i}{\partial u_i}(0,0).$$

Let us now explain how to determine the quantities which are related by (2), namely, $\mu_i(0)$, $\nu_i(0)$, $\Phi_i(0)$ and the curve's tangents $\boldsymbol{d}_i^1$, $i = 1, \ldots, n$.

If one multiplies (cross product) for each index i the G^1 equation $\Phi_i(0)\boldsymbol{d}_i^1 - \mu_i(0)\boldsymbol{d}_{i+1}^1 - \nu_i(0)\boldsymbol{d}_{i-1}^1 = 0$ by the vectors $\boldsymbol{d}_{i-1}^1, \boldsymbol{d}_i^1, \boldsymbol{d}_{i+1}^1$, resp., one gets three vector valued equations which are then multiplied (dot product) each by the vector $\boldsymbol{n}$ (normal to the $\boldsymbol{d}_i^1$), one gets a (3×3) linear system of equations of rank 2. Doing so for each index i, one gets the following formulas

$$\begin{aligned} \mu_i(0) \;&=\; \frac{|\boldsymbol{d}_i, \boldsymbol{d}_{i+1}, \boldsymbol{n}|}{|\boldsymbol{d}_{i-1}, \boldsymbol{d}_{i+1}, \boldsymbol{n}|}\;\Phi_i(0)\;, \\ \nu_i(0) \;&=\; \frac{|\boldsymbol{d}_{i-1}, \boldsymbol{d}_i, \boldsymbol{n}|}{|\boldsymbol{d}_{i-1}, \boldsymbol{d}_{i+1}, \boldsymbol{n}|}\Phi_i(0), \qquad i = 1, \ldots, n. \end{aligned} \tag{3}$$

The tangents of the boundary curves can therefore be chosen arbitrarily in length and direction as long as they belong to the same plane, namely the tangent plane. Then, the scalar values $\mu_i(0)$ and $\nu_i(0)$ are also fixed up to a scalar factor $\Phi_i(0)$ by (3). Similar formulas have been developed by Du and Schmitt in [4], but his G^1 surface is then composed of rational patches.

In practice, we choose the normalization factor $\Phi_i(0)$ such that $\mu_i(0) \cdot \nu_i(0) = \frac{1}{4}$. This choice is motivated by the need of a normalization that generalizes the regular case where $\mu_i(0) = \nu_i(0) = 1/2$. And, it will simplify the computation of the boundary curves where this product occurs??.

Fig. 4 (left) shows a zoom on a real example of a vertex with six common edges. An arbitrary choice of the curves' tangents leads to the Bézier control points of the boundary curves, as shown in Fig. 4 (middle). They are obtained by the new method which is described in Section 4. The tangent control points corresponding to constant scalar functions, $\mu_i(0) = \nu_i(0) \equiv \frac{1}{2}$, are shown in Fig. 4 (right). The edges in the background show the deviation in that case.

Geometrical Interpretation. Formula (3) has the following geometrical meaning: For each triple of tangents $\{\boldsymbol{d}_{i-1}^1, \boldsymbol{d}_i^1, \boldsymbol{d}_{i+1}^1\}$, the value of $\mu_i(0)$ is proportional to the area of the triangle $\Delta(\boldsymbol{p}, \boldsymbol{p} + \boldsymbol{d}_i^1, \boldsymbol{p} + \boldsymbol{d}_{i+1}^1)$ and $\nu_i(0)$ is proportional to the area of the triangle $\Delta(\boldsymbol{p}, \boldsymbol{p} + \boldsymbol{d}_{i-1}^1, \boldsymbol{p} + \boldsymbol{d}_i^1)$, see Fig. 5.

Scalar Functions. Once the values (3) are determined for each mesh vertex, the scalar functions can be taken

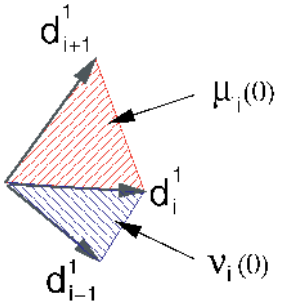

Fig. 5. Geometrical meaning of scalar functions. The values of $\mu_i(0), \nu_i(0)$ are proportional to the area of the shaded triangles.

(piecewise) linear along the domain triangle edges where they are defined on:

$$\text{(piecewise) } \textbf{linear functions :} \qquad \mu_i, \nu_i, \Phi_i : [0,1] \to I\!R.$$

Twists. The second derivatives of the patch boundary curves are also involved when establishing G^1 continuity at a mesh vertex. This can be seen when differentiating (1) with respect to u_i and evaluating it at $u_i = 0$, $i = 1, \ldots, n$. One obtains the following necessary system of conditions of G^1 continuity at the mesh vertices:

$$\begin{aligned}\Phi_i(0)\boldsymbol{d}_i^2 = \mu_i'(0)\boldsymbol{d}_{i-1}^1 - \Phi_i'(0)\boldsymbol{d}_i^1 + \nu_i'(0)\boldsymbol{d}_{i+1}^1 \\ + \mu_i(0)\boldsymbol{t}_{i-1} + \nu_i(0)\boldsymbol{t}_i, \quad i = 1, \ldots, n,\end{aligned} \tag{4}$$

where $\boldsymbol{d}_i^2 := \Gamma_i''(0) = \frac{\partial^2 S^i}{\partial u_i^2}(0,0)$ denote the second boundary curves derivatives at the vertex, and $\boldsymbol{t}_i := \frac{\partial^2 S^i}{\partial u_i \partial u_{i+1}}(0,0)$ denote the twist vector of patch S^i at the vertex. The twists and the second derivatives are the unknowns in these equations. The best way to proceed is to choose all the twists $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_n$ and then simply evaluate the equations separately from each other in order to get values for the second derivative $\boldsymbol{d}_i^2$. No linear system has to be solved for that. How the twists are chosen will be explained in Section 4.

Conclusion. At each mesh vertex, we now have fixed the data which is necessary to satisfy G^1 continuity there:

- the position (interpolation of mesh vertices),
- the tangents,
- the second derivatives of the patch boundary curves,
- the twists of the patches surrounding that vertex.

One is therefore tempted to believe that a curve network composed of quintic Hermite curves between two neighboring vertices which interpolates these data could be chosen. Unfortunately, this would generally lead to rational patches, as we will explain in the next section.

3.3 Curves and Tangent Ribbons Are Linked

What we call tangent ribbons are the cross-boundary derivatives along the macropatch boundary curves. They ensure the G^1 continuity between adjacent patches. An equivalent condition to (1) of G^1 continuity is used for their construction.

Tangent Ribbons. The patches S^i and S^{i-1} meet with G^1 continuity along the common boundary if there exist three scalar functions Φ_i, μ_i, ν_i and a vector valued function in $I\!R^3$ $\boldsymbol{V}_i$ such that the following two equations hold:

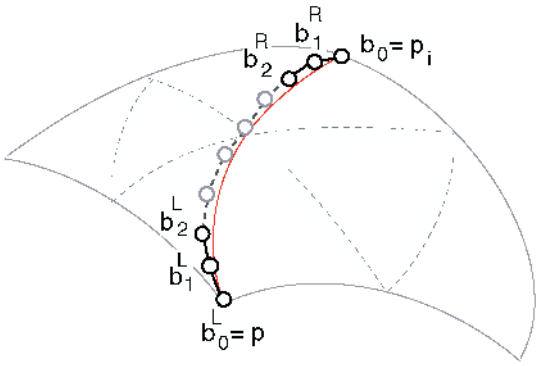

Fig. 6. Bézier control points of boundary curves.

$$2\,\mu_i(u_i)\,\frac{\partial S^i}{\partial u_{i+1}}(u_i,0) = \Phi_i(u_i)\,\Gamma_i'(u_i) + \boldsymbol{V}_i(u_i) \tag{5}$$

$$2\,\nu_i(u_i)\,\frac{\partial S^{i-1}}{\partial u_{i-1}}(0,u_i) = \Phi_i(u_i)\,\Gamma_i'(u_i) - \boldsymbol{V}_i(u_i). \tag{6}$$

The equivalence between (5), (6), and (1) can be seen by adding up (5), (6). From Section 3.2, it is known that μ_i and ν_i are linear functions in order to guarantee an arbitrary choice of the tangents at the vertices. This implies, however, that the tangent ribbons $\frac{\partial S^i}{\partial u_{i+1}}(u_i,0)$ and $\frac{\partial S^{i-1}}{\partial u_{i-1}}(0,u_i)$ defined by (5), (6) between S^i and S^{i-1} would be, in general, rational. Therefore, the patches would all be rational in contradiction to our main requirement: The interpolation surface must be polynomial.

Boundary Curves. In order to keep in the class of polynomial patches, the boundary curves Γ_i common to a mesh vertex are chosen such that they satisfy the following condition: Their derivatives $\Gamma_i'(u_i) = \frac{\partial S^i}{\partial u_i}(u_i,0)$ are defined as a product of the linear scalar functions μ_i, ν_i and a vector valued piecewise polynomial function $\boldsymbol{H}_i$, i.e.,

$$\Gamma_i'(u_i) := \mu_i(u_i) \cdot \nu_i(u_i) \cdot \boldsymbol{H}_i(u_i), \qquad i = 1, \ldots, n. \tag{7}$$

The choice (7) is the only possibility that yields a polynomial solution of (5) and (6). Now, an explicit formula of the boundary curves can be given by assembling the results about the tangents and second derivatives at each vertex from Section 3.2 subject to the polynomial condition (7) between two neighboring vertices. Each boundary curve is finally a uniquely defined piecewise G^1 **quintic Bézier curve** $\Gamma_i(u_i)$, $u_i \in [0,1]$, composed of two pieces. The two sets of control points $\{\boldsymbol{b}_k^L\}_{k=0}^5$ and $\{\boldsymbol{b}_{5-k}^R\}_{k=5}^0$ are shown in Fig. 6.

The control points $\boldsymbol{b}_0^L, \boldsymbol{b}_1^L, \boldsymbol{b}_2^L$ and $\boldsymbol{b}_0^R, \boldsymbol{b}_1^R, \boldsymbol{b}_2^R$ are determined by interpolating the position, tangent, and second derivative known at the end points $\boldsymbol{p}_k, \boldsymbol{p}_i$ (mesh vertices).

The function $\boldsymbol{H}_i$ defines the curve's derivative by (7). It can be chosen piecewise quadratic. Its Bézier control points are denoted by $\boldsymbol{h}_k^L$ and $\boldsymbol{h}_k^R$, $k = 0, 1, 2$. $\boldsymbol{h}_0^L, \boldsymbol{h}_1^L, \boldsymbol{h}_0^R, \boldsymbol{h}_1^R$ are known from $\boldsymbol{b}_0^L, \boldsymbol{b}_1^L, \boldsymbol{b}_2^L$ and $\boldsymbol{b}_0^R, \boldsymbol{b}_1^R, \boldsymbol{b}_2^R$ since the position and tangent of Γ_i' are already known. A comparison of the coefficients in (7) together with the two linear conditions expressing C^1 continuity of Γ_i at $u_i = 1/2$ determine exactly the remaining unknowns.

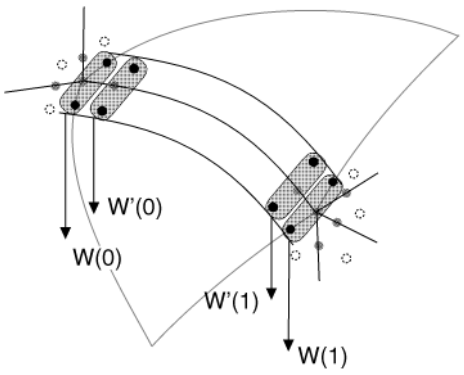

Fig. 7. G^1 compatible tangent ribbons at end points.

3.4 Compatible Tangent Ribbons

The tangent ribbons which guarantee tangent plane continuity between adjacent patches have to satisfy conditions (5) and (6). The requirements on the unknown function Vi are twice. V_i has to be compatible with the polynomial requirement. Therefore, the same multiplying factor as in the case of the curve derivative (see (7)) is necessary:

$$\boldsymbol{V}_i(u_i) := \mu_i(u_i) \cdot \nu_i(u_i) \cdot \boldsymbol{W}_i(u_i), \qquad u_i \in [0,1]. \quad (8)$$

Second, at the end points (mesh vertices), V_i has to be compatible with the boundary curve tangents computed earlier (Section 3.2) since the following relation holds:

$$\frac{\partial S^i}{\partial u_{i+1}}(0,0) = \boldsymbol{d}^1_{i+1}, \qquad i = 1, \ldots, n.$$

The same relations hold at the opposite vertex; they fix the values of V_i at the end points, i.e., $\boldsymbol{W}_i(0)$, $\boldsymbol{W}_i(1)$. Analogously, the G^1 compatible twists computed earlier (Section 3.2) at each vertex are related to the derivative of V_i at the end points, i.e., they fix the values of $\boldsymbol{W}'_i(0)$, $\boldsymbol{W}'_i(1)$.

All data we need for G^1 compatible tangent ribbons are fixed now. W_i can interpolate these data if we choose it as a piecewise C^0 quadratic or a cubic Hermite curve. V_i will therefore be either piecewise C^0 quartic or quintic polynomial. Finally, from (5), (6) together with (8), it follows that the tangent ribbons will be either a piecewise C^0 cubic or a quartic polynomial. Both possibilities don't increase the degree of the patches, which is equal to five because of the boundary curves. An explicit Bézier representation can be computed directly from the preceding considerations. See Fig. 7.

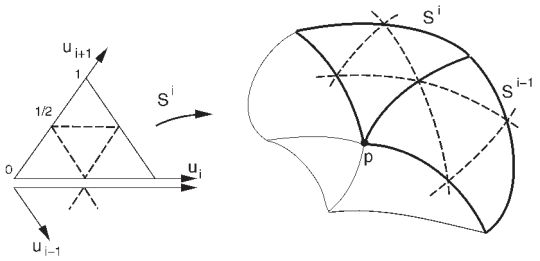

Fig. 8. Four-split parameterization of the macropatches.

3.5 Constructing Bézier Patches

The third and last step of the algorithm consists of the construction of the surface patches by determining the remaining control points. How to choose them and what kind of conditions they must fulfill are the two questions to be answered now. Recall that we call the group of four Bézier patches associated with an input mesh triangle a macropatch (see Section 1).

It turns out from Section 3.3 that the macropatch boundary curves are piecewise quintic curves consisting of two pieces joining with C^1 continuity at the parameter value $\frac{1}{2}$. The underlying parameterization for each macropatch is based on a 4-split of the domain triangle, as shown if Fig. 8.

The macropatches are finally composed of four quintic triangular Bézier patches. From Sections 3.3 and 3.4, we know the coordinates of all the boundary and the first inner row of control points, see black dots in Fig. 9 (left). These data guarantee a G^1 joint between all macropatches. However, the remaining 15 inner control points of each macropatch are not free because the joints between these four inner patches also need to be tangent plane continuous. In [7], it is shown that two triangular Bézier patches of the same degree join with C^1 continuity along their common boundary if all pairs of subtriangles (subsets of four control points) across the common boundary are coplanar and are an affine map of the domain triangles. In the present method, the patches are defined on the unit triangle, which implies that each pair of subtriangles of control points must be a parallelogram, see Fig. 9 (right). Since this linear parallelogram equation already holds at the boundary edge midpoints, the remaining 15 inner control points are finally related across the inner boundaries by nine linear parallelogram equations. Therefore, six inner control points remain free for shape control.

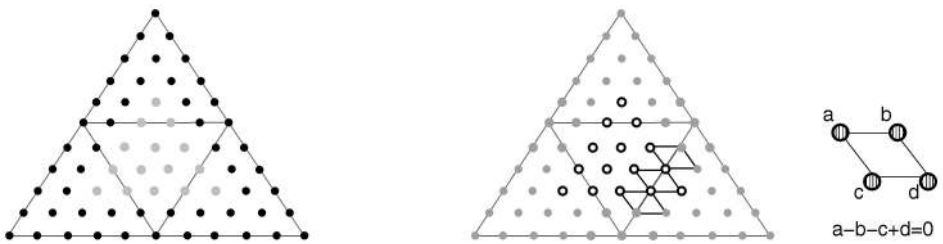

Fig. 9. Left: Schematic representation of macropatch control points that ensure the G^1 joint between all macropatches. Right: C^1 conditions for inner control points.

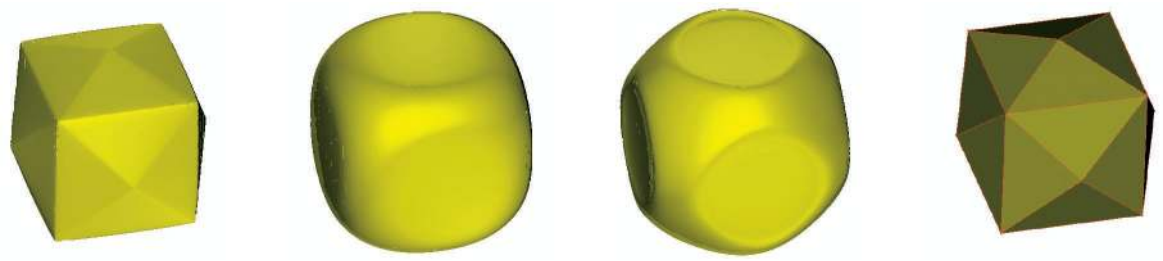

Fig. 10. Three interpolation surfaces—input "cube" mesh.

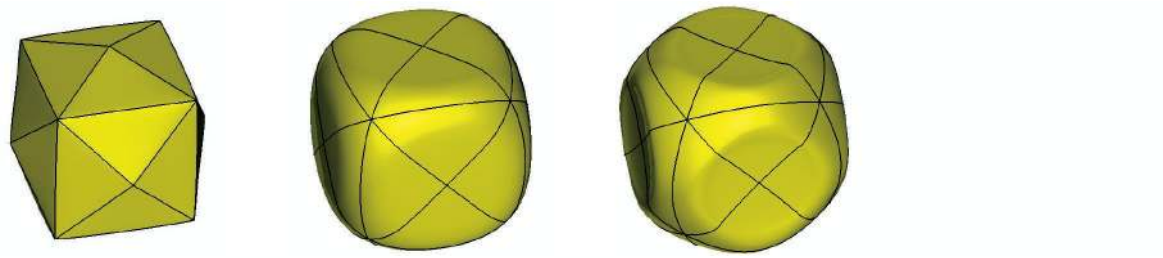

Fig. 11. Three interpolating surfaces with boundary curves.

4 Design Issues

Interpolation of arbitrary triangle meshes with the present method based on an arbitrary choice of the curve network offers a lot of degrees of freedom. They are welcome for shape control as well as for modeling complex shapes. Let us enumerate these degrees of freedom. For each mesh vertex of order n, there is one control point, there are n tangents, and n twists which are free. Then, for each macropatch corresponding to a mesh face, there are six control points free. These degrees of freedom have two main advantages: First, almost all of them offer an intuitive geometrical way to fix them. Second, they are numerous enough to allow construction of globally smooth surfaces.

Some Rules for Fixing the Degrees of Freedom. It is obvious that one input mesh can have quite different interpolating surfaces. Ideally, the surface can be seen as an enveloping skin of the mesh vertices which is more or less tight. This is equivalent to saying that one wants a globally smooth surface. From the survey of Mann et al. [17], it is known that the global smoothness is the most difficult challenge for all mesh interpolation schemes. Other particular design features, like flat points, sharp points, or corners, can, of course, be obtained by acting locally on the degrees of freedom. In Figs. 10 and 11, from left to right, three surfaces are shown which all interpolate the same input mesh (Figs. 10 (right)).

All the degrees of freedom listed above can be set up by using some heuristic, but geometrically-based, rules such that the resulting surface generally suffices the user.

- For each mesh vertex, there is a theoretical free **point**, corresponding to the patch corners and the boundary curve's end points. It is set up equal to the mesh vertex in order to make the surface interpolating the mesh. However, it is always possible to fix them otherwise.
- At each vertex of valence n, there are n **tangents** free to choose. Together with the second derivatives, they entirely determine the boundary curves coming in to that vertex. The curve network which is interpolated by the patches is the most important step toward a globally smooth and well-pleasing surface. Since the curves are constructed in correspondence to the mesh edges, we choose the following geometrically intuitive rule to fix the tangent: As tangent vector for the edge between the vertex $\boldsymbol{p}$ and $\boldsymbol{p}_i$ take the unit vector in the intersection of the tangent plane at $\boldsymbol{p}$ and the plane spanned by the edge and the normal vector at $\boldsymbol{p}_i$ and which is scaled by an appropriate factor. The default scaling factor for each tangent is taken as proposed in [22], i.e., $\frac{4}{9}\|\boldsymbol{p}-\boldsymbol{p}_i\|$ for a tangent of a cubic curve. This scaling factor can furthermore be used as an intuitive design handle, which can easily be modified in a 3D graphics program. It is geometrically very intuitive because it governs the flatness of the surface at the vertices. The surfaces in Figs. 10 and 11 from left to right are obtained by simply increasing this scalar factor simultaneously for all vertices.
- At each vertex, there are n **twists** free. They determine entirely the n second derivatives of the boundary curves around the vertices, see (4). The question of how to choose twists of polynomial patches has driven a lot of publications [1], [23], [20], [10], [3]. Zero twists are always an easy solution, but, in [7], it is shown that they never lead to satisfactory results. This can be confirmed in the present case. The lack of an intuitive heuristic rule for these values leads us to use a linear least squares minimization in order to choose them. More specifically, we minimize the following energy integral, a linearized version of a thin plate's bending energy: $\int S_{uu}+2S_{uv}+S_{vv}dudv$. All the degrees of freedom, which are fixed up to now, allow for the construction of uniquely defined patch boundary curves and G^1 compatible cross boundary tangents (Sections 3.2-3.4).

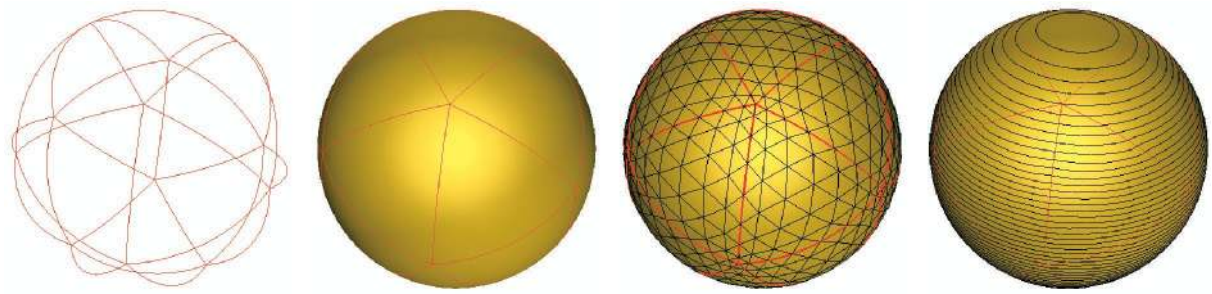

Fig. 12. Icosahedron example, from left to right: boundary curves, surface with boundary curves, surface with iso-parametric lines, surface with isophotos.

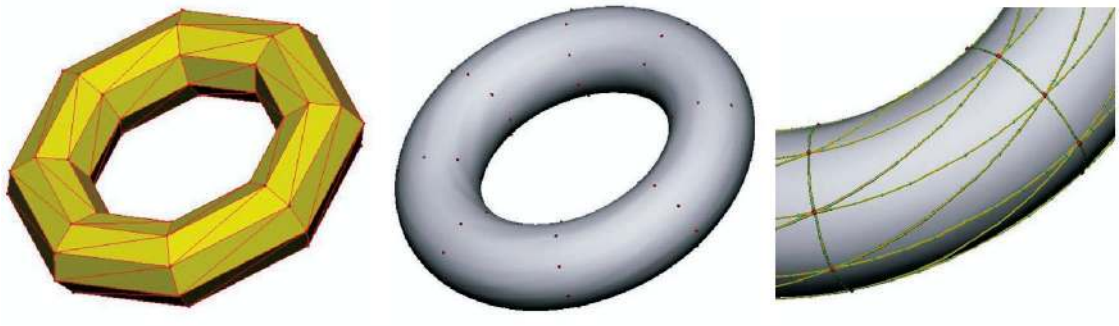

Fig. 13. Torus example. From left to right: input mesh, surface with interpolated vertices, zoom with boundary curve's control polygons.

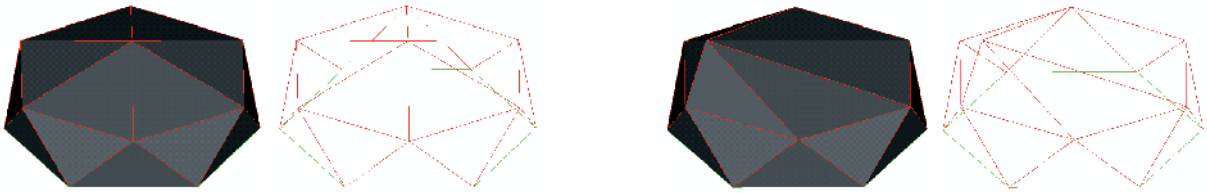

Fig. 14. Left: regular mesh, right: deformed mesh. Both meshes are shown with flat shading and wire frame; they are open meshes.

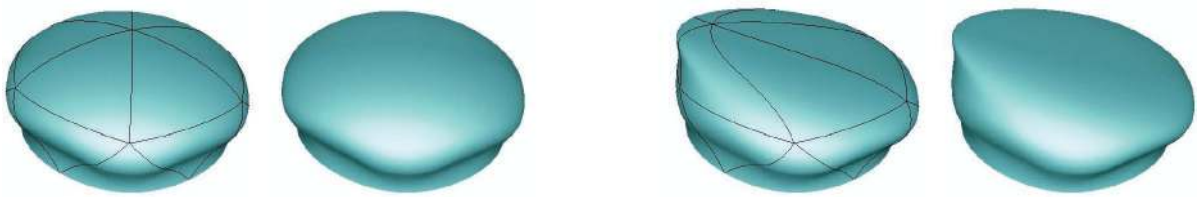

Fig. 15. Interpolation with affine map of tangents.

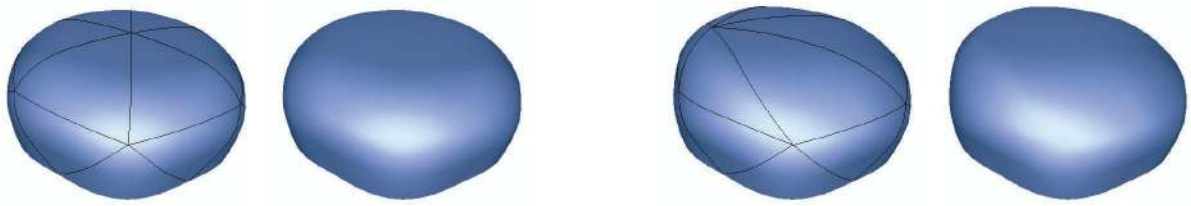

Fig. 16. Interpolation with arbitrary choice of tangents.

- At the interior of each macropatch, there are **six inner control points** free for shape control (Section 3.5). This doesn't differ from the previously developed method [12]. We generally use the same least square energy minimization as for the computation of the twists. This is explained in more detail in [13].

5 Results

We begin this section with examples of familiar shapes. These examples allow some comparison with earlier related works based on Clough-Tocher splitting because these works were tested in Mann et al.'s state-of-the-art paper [17] using similar familiar shapes. The methods using a Clough-Tocher split [22], [24], [14] have the advantage of yielding degree four polynomial patches and only three patches per

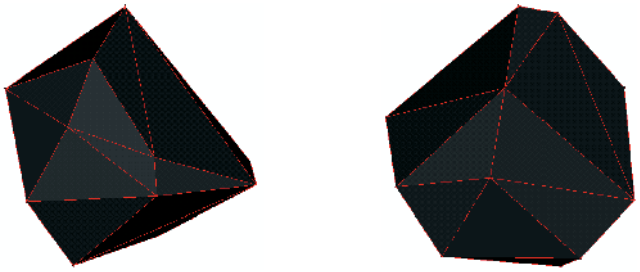

Fig. 17. Mesh with irregular vertices.

macropatch. Together with [21] they also allow for arbitrary tangents. However, the sphere and torus examples of Figs. 12 and 13, compared with the results shown in Mann et al.'s paper, clearly show that our method yields better results. We believe that this is mainly due to the parameterization of the surface, which is smoother because of the 4-split of the triangles. This improved behavior of the iso-parametric lines is illustrated in Fig. 12 in comparison with Fig. 8.8 in [17].

In the rest of this section, we show the results of our new interpolation method on several triangular meshes with irregular features, like small and long edges joining at a common vertex or "dirty" and equilateral triangles joining along a common edge. These features are known to cause unfair surfaces. For each mesh, we compare the results of our new method with those obtained by the earlier methods based on a regular n-gone choice of the tangents around the interpolated vertices.

Fig. 14 shows left a mesh with regular features: It is an open symmetric mesh with vertices of valence 4, 5, and 6, the edges common to the central vertex all have the same length, and all triangles are more or less equiangular. The right part of Fig. 14 shows the mesh resulting from the translation of the central vertex. Flat triangles appear and short edges join to long edges at the central vertex. Fig. 15 shows the interpolating surface of these two meshes with the previously introduced 4-split method [12]. Each surface is shown twice: with and without the macropatch boundary curves. While the surface obtained for the regular mesh is acceptable (Fig. 15 (left)), the result for the irregular mesh has an unpleasant shape (Fig. 15 (right)). Note, in particular, the overshooting undulations of the surface along the short edges which are joined with the central vertex (Fig. 15 (right)). This behavior is due to the fact that the lengths of all tangents at the central vertex are the same. They can't adapt to the length of the curves. Fig. 16 shows the results for the new method. For both meshes, the interpolating surface has a nice shape, unwanted undulations are avoided.

The next example is a closed mesh with artificially created heavy irregularities. Fig. 17 shows two different views we want to focus on. The resulting interpolating surfaces obtained with the early method, Fig. 18, are then compared with the results of the new method, Fig. 19. Again, unwanted undulations are avoided with the new method and globally fair surfaces are obtained. It is particularly remarkable how the macropatch boundary curves are able to smoothly follow the edges because of the free choice of the first derivatives and the ability to vary their lengths around a vertex. On the contrary, the regular n-gone choice of the first derivatives in the old method is clearly the reason why macropatch boundary curves—and therefore the surface itself—has unwanted undulations (see Fig. 18).

In comparison to these results, let us also show in Fig. 20 the result obtained with one of the two triangular interpolatory subdivision surface schemes: the butterfly subdivision scheme [5]. Again, the shape of the surface is clearly not acceptable for this input mesh.

Figs. 21 and 22 show the result of our interpolating scheme on a more complex data set, the mannequin data set (courtesy of the University of Washington). The right part of Fig. 21 shows the control-nets of the quintic Bézier patches. The central patch of each macropatch is colored in red. Fig. 22 shows two details of the mannequin interpolating surface, without and with the control-polygon of the

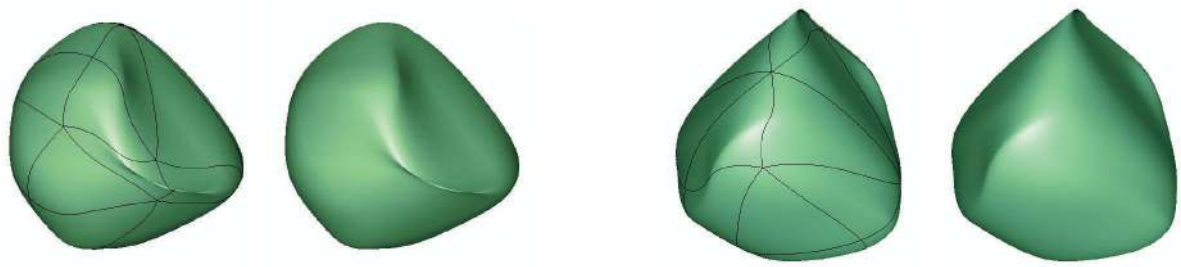

Fig. 18. Interpolation with affine map of tangents.

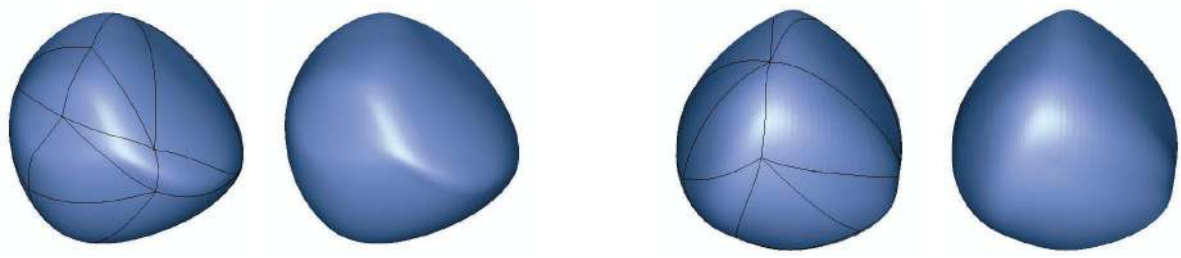

Fig. 19. Interpolation with arbitrary choice of tangents.

Fig. 20. Interpolation with the butterfly subdivision scheme.

boundary curves. Note again the overall smoothness and how the boundary curves smoothly follow the edges of the input mesh. This is possible only because of the free choice of the first derivatives.

6 Conclusion

We have introduced a new method for interpolating 2-manifold triangular meshes with a parametric surface composed of Bézier patches of degree 5.

While previous similar schemes enforced a regular n-gone choice of the first derivatives at the interpolated vertices, this new method allows a completely free choice of these tangents vectors. We have shown how to derive first and second derivative information at the interpolated vertices and across the boundary curves between interpolated vertices such that a polynomial interpolant of low degree can be found. In comparison with previous similar schemes, this new method allows us to find pleasing shapes, without unwanted undulations, even if the interpolated mesh has nonregular features, e.g., short and long edges joining at a common vertex or flat and big triangles joining along a common edge.

Future work will include, in particular, the refinement of this interpolation scheme. Based on the fact that our method utilized a regular 4-split of the input triangles, we will show that our interpolation scheme is refinable: We will prove that applying our interpolation scheme on a carefully subdivided triangulation yields the same interpolating surface, in other words, the interpolation scheme is invariant under subdivision. We will build a multiresolution modeling scheme for the design and edition of complex shapes at different levels of detail.

Acknowledgments

The authors would like to thank the *TVCG* reviewers for their helpful comments. This work was partially supported by the European Community 5-th framework program, with the Research Training Network MINGLE (Multiresolution IN Geometric modELing, HPRN-1999-00117).

References

[1] R. Barnhill, J. Brown, and I. Klucewicz, "A New Twist in CAGD," *Computer Graphics and Image Processing,* 1978.
[2] C. Bajaj, "Smoothing Polyhedra Using Implicit Algebraic Splines," *Computer Graphics,* vol. 26, no. 2, pp. 79-88, 1992.
[3] P. Brunet, "Increasing the Smoothness of Bicubic Spline Surfaces," *Proc. Surfaces in CAGD '84,* R. Barnhill and W. Böhm, eds., 1985.

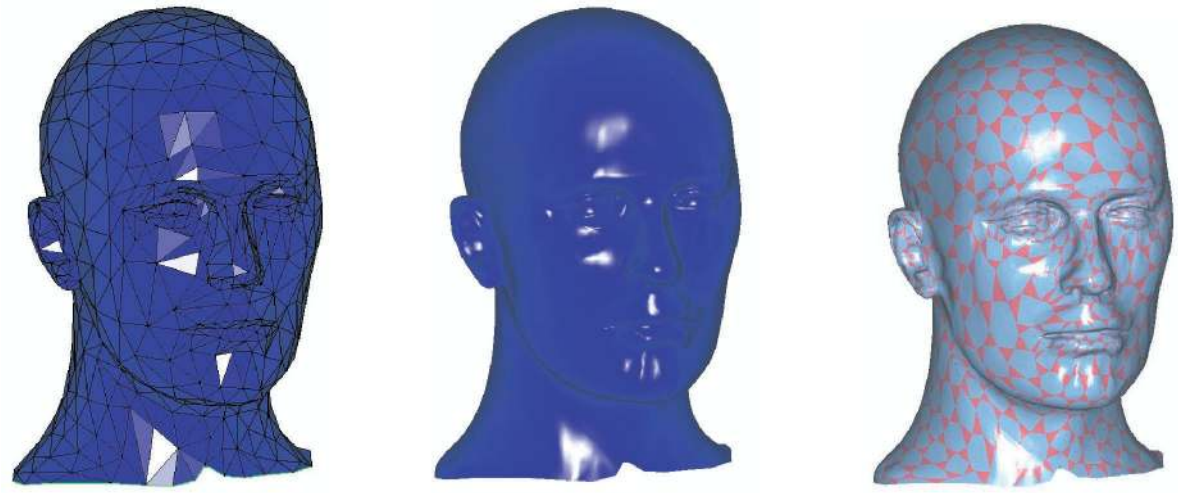

Fig. 21. Mannequin, mesh left, G^1 surface middle, control-polygons right.

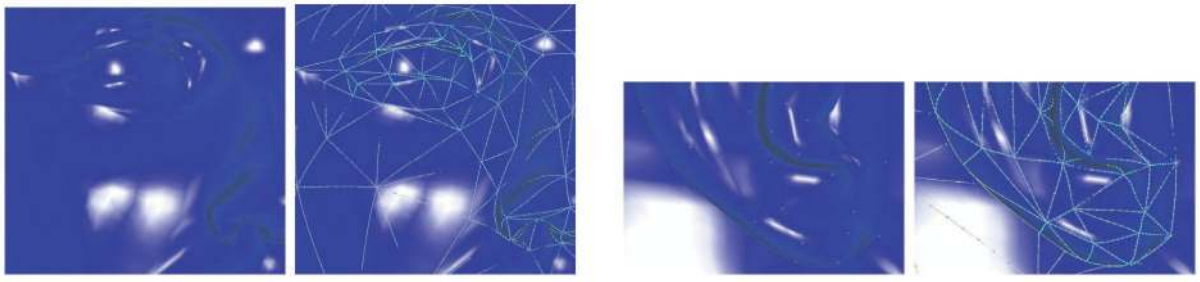

Fig. 22. Zoom of the mannequin, without and with control-polygons of boundary curves.

[4] H.W. Du and F. Schmitt, *CAD,* vol. 22, no. 9, pp. 556-573, 1990.
[5] N. Dyn, D.F. Levin, and J.A. Gregory, "A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control," *ACM Trans. Graphics,* vol. 9, pp. 160-169, 1990.
[6] G. Farin, "A Construction for Visual C^1 Continuity of Polynomial Surface Patches," *Computer Graphics and Image Processing,* vol. 20, pp. 272-282, 1982.
[7] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design,* fourth ed. New York: Academic Press, 1996.
[8] J.A. Gregory, "N-Sided Surface Patches," *The Math. of Surfaces,* J. Gregory, ed., pp. 217-232, Oxford: Clarendon Press, 1986.
[9] H. Hagen, "Geometric Surface Patches without Twist Constraints," *Computer Aided Geometric Design,* vol. 3, pp. 179-184, 1986.
[10] H. Hagen and G. Schulze, "Automatis Smoothing with Geometric Surface Patches," *Computer Aided Geometric Design,* vol. 4, pp. 231-236, 1987.
[11] H. Hagen and H. Pottmann, "Curvature Continuous Triangular Interpolants," *Math. Methods in Computer Aided Geometric Design,* T. Lyche and L.L. Schumaker, eds., pp. 373-384, New York: Academic Press, 1989.
[12] S. Hahmann and G.-P. Bonneau, "Triangular G^1 Interpolation by 4-Splitting Domain Triangles," *Computer Aided Geometric Design,* vol. 17, pp. 731-757, 2000.
[13] S. Hahmann, G.-P. Bonneau, and R. Taleb, "Smooth Irregular Mesh Interpolation," *Curve and Surface Fitting: Saint-Malo 1999,* A. Cohen, C. Rabut, and L.L. Schumaker, eds., pp. 237-246, Nashville: Vanderbilt Univ. Press, 2000.
[14] T. Jensen, "Assembling Triangular and Rectangular Patches and Multivariate Splines," *Geometric Modeling: Algorithms and New Trends,* G. Farin, ed., pp. 203-220, SIAM, 1987.
[15] C. Loop, "A G^1 Triangular Spline Surface of Arbitrary Topological Type," *Computer Aided Geometric Design,* vol. 11, pp. 303-330, 1994.
[16] S. Mann, "Surface Approximation Using Geometric Hermite Patches," PhD dissertation, Univ. of Washington, 1992.
[17] S. Mann, C. Loop, M. Lounsbery, D. Meyers, J. Painter, T. DeRose, and K. Sloan, "A Survey of Parametric Scattered Data Fitting Using Triangular Interpolants," *Curve and Surface Design,* H. Hagen, ed., pp. 145-172, SIAM, 1992.
[18] M. Neamtu and P. Pluger, "Degenerate Polynomial Patches of Degree 4 and 5 Used for Geometrically Smooth Interpolation in $I\!R^3$," *Computer Aided Geometric Design,* vol. 11, pp. 451-474, 1994.
[19] G. Nielson, "A Transfinite, Visually Continuous, Triangular Interpolant," *Geometric Modeling: Algorithms and New Trends,* G. Farin, ed., pp. 235-246, SIAM, 1987.
[20] H. Nowacki and D. Reese, "Design and Fairing of Ship Surfaces," *Computer Aided Geometric Design,* R. Barnhill and W. Böhm, eds., pp. 121-134, North-Holland, 1982.
[21] J. Peters, "Smooth Interpolation of a Mesh of Curves," *Constructive Approximation,* vol. 7, pp. 221-246, 1991.
[22] B.R. Piper, "Visually Smooth Interpolation with Triangular Bézier Patches," *Geometric Modeling: Algorithms and New Trends,* G. Farin, ed., pp. 221-233, SIAM, 1987.
[23] S. Selesnick, "Local Invariants and Twist Vectors in CAGD," *Computer Graphics and Image Processing,* vol. 17, pp. 145-160, 1981.
[24] L.A. Shirman and C.H. Séquin, "Local Surface Interpolation with Bézier Patches," *Computer Aided Geometric Design,* vol. 4, pp. 279-295, 1987.
[25] J.J. Van Wijk, "Bicubic Patches for Approximating Non-Rectangular Control Meshes," *Computer Aided Geometric Design,* vol. 3, pp. 1-13, 1986.

Stefanie Hahmann received the DEA in numerical analysis from the University of Paris VI, France, the MS degree in mathematics from the University of Braunschweig, Germany, and the PhD degree in computer science from the University of Kaiserslautern, Germany, in 1994. She is an associate professor at the Institut National Polytechnique de Grenoble (INPG-ENSIMAG) and a member of the research lab LMC-IMAG, France. Her research centers on geometric modeling, multiresolution methods, and scientific visualization.

Georges-Pierre Bonneau received the MS degree in mathematics from the Ecole Normale Supérieure de Cachan, France, the DEA in numerical analysis from the University of Paris VI, France, and the PhD degree in computer science from the University of Kaiserslautern, Germany, in 1993. He is a professor of computer science at the University Joseph Fourier, Grenoble, France. He is a member of the research team iMAGIS/INRIA, within the research lab GRAVIR/IMAG. His research centers on visualization, multiresolution methods, and CAGD. He was coorganizor, with Frits Post and Greg Nielson, of the Dagstuhl seminar on "Scientific Visualization" in May 2000 and was a member of several conference and program committees, including IEEE Visualization and Eurographics VisSym.