ELSEVIER

Computer Aided Geometric Design 17 (2000) 731–757

COMPUTER AIDED GEOMETRIC DESIGN

www.elsevier.com/locate/comaid

Triangular G^1 interpolation by 4-splitting domain triangles

Stefanie Hahmann *, Georges-Pierre Bonneau

University of Grenoble, Laboratoire LMC-IMAG, B.P. 53, F-38041 Grenoble cedex 9, France

Received September 1999; revised March 2000

Abstract

A piecewise quintic G^1 spline surface interpolating the vertices of a triangular surface mesh of arbitrary topological type is presented. The surface has an explicit triangular Bézier representation, is affine invariant and has local support. The twist compatibility problem which arises when joining an even number of polynomial patches G^1 continuously around a common vertex is solved by constructing C^2-consistent boundary curves. Piecewise C^1 boundary curves and a regular 4-split of the domain triangle make shape parameters available for controlling locally the boundary curves. A small number of free inner control points can be chosen for some additional local shape effects.

Keywords: G^1 continuity; Triangular Bézier surfaces; Twist compatibility; Interpolation; Triangle 4-split

1. Introduction

Defining surfaces from a set of points, which control in an intuitive way the form of a surface due to Bernstein–Bézier or B-spline basis functions has been becoming one of the most popular methods for modeling free form surfaces. The surface hereby is defined as a regular polynomial (possibly rational) map of a planar domain, tessellated into a regular grid of rectangles or triangles, resulting in a collection of tensor product or triangular patches. Such surface definitions generally don't allow the representation of surfaces of arbitrary topological type. n-sided patches can fill in n-sided holes in rectangular patch configurations and offer therefore the possibility to represent general closed surfaces or surfaces with handles. Nevertheless if one wants to model entire surfaces with n-sided patches, restrictions on the control net must be accepted.

* Corresponding author. E-mail: hahmann@imag.fr.

0167-8396/00/$ – see front matter
PII: S0167-8396(00)00021-2

A widely accepted and popular way in defining surfaces without any limit of topologies is the use of smoothly joined triangular patches, where each patch is defined over the unit triangle. They have the advantage to provide a uniform description for all possible topologies without any restriction on the number of faces that meet at a vertex, or on the number of edges of the faces.

The paper is concerned with defining a geometrical smooth surface by interpolating a triangulated set of points in $\mathbb{R}^3$. Such a triangulated point set which we call *surface mesh* should be 2-manifold and is allowed to represent surfaces of arbitrary topological type. There is no restriction on the order of the mesh vertices (i.e., the number of faces that meet at a vertex). Furthermore the surface mesh furnishes topology information, which is a data structure generating adjacency informations relating vertices, edges and faces. We assume that the surfaces mesh is already given.

Local smooth *triangular interpolants* of an arbitrary surface mesh have been developed by many. These previous works are the most directly related to the results found here in the sense that they interpolate a control net and not only approximate it. They can be divided into different groups depending on how they solve the "vertex consistency problem", which occurs when joining with G^1 continuity an even number of C^2-patches around a vertex. The earliest of these schemes are Clough–Tocher-like domain splitting methods (Farin, 1982; Piper, 1987; Shirman and Séquin, 1987; Jensen, 1987). Since the surface mesh triangles are divided into sub-triangles, we refer to them as *macro-triangles*. Three quartic G^1 patches per macro-triangle interpolating positions and normals are produced. One problem is how to employ the free parameters in order to get pleasing shapes. Convex combination schemes (Nielson, 1987; Hagen, 1986; Hagen and Pottmann, 1989; Gregory, 1986), blend side-side or side-vertex operators in order to interpolate transfinite position, tangent or curvature data of the boundary curves. They are rational patches without consistently defined twists at the vertices. The use of singular parameterizations (Neamtu and Pluger, 1994) is another possibility but seems to have problems in defining pleasing shapes. The boundary curve schemes (Peters, 1991a; Loop, 1994) first create C^2-consistent boundary curves and then fill in the patches polynomial. Furthermore some special interpolation methods can be found in (Herron, 1985; Van Wijk, 1986; Sarraga, 1987). They all make either restrictions on the mesh topology or on the input data, and are therefore not general enough in order to be compared with the methods listed above. An overview and comparison of most of these methods can be found in (Mann, 1992; Mann et al., 1992).

The surface interpolation scheme of an arbitrary mesh of points in $\mathbb{R}^3$, which is presented in this paper will satisfy the following requirements. They are desirable for the reasons that will be explained below:

- the surface *interpolates the vertices* of the given surface mesh. And if desired, the interpolation condition can be relaxed in order to only approximate the mesh,
- the surface is G^1 *continuous* for visual smoothness,
- the surface is *piecewise triangular* and the definition domain of the surface is the input surface mesh itself,
- the surface can be of arbitrary topological type,
- the surface results of a local interpolation method, where only a few data of the

corresponding mesh triangle and its neighbors is used. Global interpolation schemes generally result in a big system of equations where all input data influences the shape of every patch,

– an explicit *closed form polynomial and low degree* parameterization is given for each patch. Fast surface evaluations and calculus on the surfaces, such as derivatives and curvature, are important for rendering and interrogation purposes,
– the surface is *affine invariant* and *shape parameters* are available for local shape control.

An interesting triangular G^1 surface spline, which motivated this work, was recently given by Loop, and consists of triangular Bézier patches of degree six, one per macro-triangle. All requirements are satisfied except one: interpolation is theoretically possible, but leads to unwanted surface undulations in practice (Loop, 1994). These undulations are due to severe constraints on the second derivatives along the boundary curves, at each end-point. The surface mesh therefore only acts as a control mesh which is approximated and not interpolated.

In this paper we present an interpolating quintic G^1 triangular spline surface, which is a generalization of Loop's scheme. All requirements are fulfilled. Four Bézier patches per *macro-triangle* are created by a local scheme. The basic idea, which allows to perform interpolation without undulations, is to use a regular 4-split of the domain triangles. As a consequence of the 4-split, the constraints between derivatives at each end-point of the boundary curves are relaxed, and an interpolating curve network, without unwanted undulations, can be built. This approach has never been used before for parametric G^1 interpolation of triangulated surface meshes in $\mathbb{R}^3$. The advantage over the Clough–Tocher-split is that tiny triangles are avoided, the sub-triangles are more regular. Mesh vertices of arbitrary order are allowed. The vertex consistency problem is solved by constructing C^2-consistent boundary curves. The 4-split doesn't solve the vertex consistency problem, like the Clough–Tocher-split does, but it introduces enough degrees of freedom enabling to produce this new quintic surface spline. It will furthermore been shown in this paper that the additional vertices of order six which are introduced by the 4-split don't present the vertex consistency problem and that the four patches per macro-triangle join C^1-continuously to each other.

The paper is organized as follows. Section 2 reviews the G^1 conditions when a pair of parametric surfaces meet and when a collection of parametric patches meet at a corner. The "vertex consistency problem" which arises when an even number of patches meet at a corner is discussed. Section 3 briefly recalls the results of Loop, and shows an example where unwanted oscillations occur when interpolating meshes with this method. Section 4 gives some general remarks on the 4-split of the macro-patches. The following Sections 5–8 concentrate on the different steps of the surface construction resulting in an explicit representation of the four Bézier patches which interpolate the corners of a mesh triangle. Examples illustrating different meshes interpolation are given in Section 9. Eventually, Section 10 offers some concluding remarks and directions for future work.

2. Notations and G^1-conditions

2.1. Surface mesh

Let $\mathcal{M}$ denote the input *surface mesh*. It consists of a list of vertices and a list of edges. Together they describe a 2-manifold mesh in $\mathbb{R}^3$ whose faces are triangles. The number of faces/edges incident in one vertex is referred as *order of a vertex*.

We aim to construct a piecewise triangular surface $\mathcal{S}$ that interpolates the given vertices V. The spline surface is composed of triangular *macro-patches* M^i which are in one-to-one correspondence with the mesh facets. They are all polynomial images of the unit triangle in $\mathbb{R}^2$, composed of four Bézier triangles each, joining G^1 continuously. We assume the reader is familiar with Bézier curves and surfaces (Farin, 1997; Hoschek and Lasser, 1993).

The algorithm for constructing the spline surface consists mainly of three steps
- constructing boundary curves,
- constructing cross-boundary tangents,
- filling in the patches.

The boundary curves of the macro-patches are constructed in correspondence with a mesh edge. Therefore there is a one-to-one correspondence between the mesh faces and the macro-triangles of $\mathcal{S}$. It is therefore convenient for the following sections to choose a parameterization of the macro-patches M^i around a common vertex, sharing pairwise a common boundary as illustrated in Fig. 1.

All subscripts $i = 1, \ldots, n$ are taken modulo n, where n is the order of the mesh vertex corresponding to $M^i(0,0)$. The parameter u_i lies in the interval $[0,1]$.

In order to allow a unified treatment of the surface patches, the surface mesh $\mathcal{M}$ is supposed to be closed. We shall point out that since the scheme is local, there should be no theoretical difficulties in treating meshes with boundaries. This is left for further research.

2.2. G^1 continuity between two adjacent patches

Consider two adjacent patches $M^{i-1}(u_{i-1}, u_i)$ and $M^i(u_i, u_{i+1})$ that share a common boundary, i.e., $M^{i-1}(0, u_i) = M^i(u_i, 0)$ for $0 \leqslant u_i \leqslant 1$.

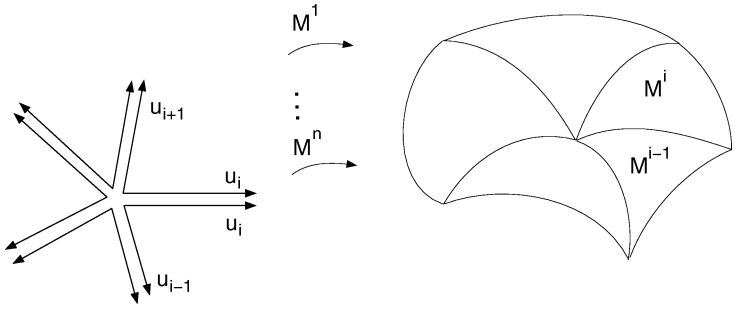


Fig. 1. Parameterization of macro-patches around a vertex.

Two adjacent patches M^i, M^{i-1} join at a common boundary with G^1 continuity if and only if there exist three scalar functions Φ_i, ν_i and μ_i such that

$$\Phi_i(u_i)M^i_{u_i}(u_i,0) = \nu_i(u_i)M^i_{u_{i+1}}(u_i,0) + \mu_i(u_i)M^{i-1}_{u_{i-1}}(0,u_i), \quad u_i \in [0,1], \qquad (I_C)$$

where $\nu_i(u_i)\,\mu_i(u_i) > 0$ (preservation of orientation) and $M^i_{u_i}(u_i,0) \times M^i_{u_{i+1}}(u_i,0) \neq 0$ (well defined normal vectors). $M^i_{u_i}$ denotes the partial derivative of M^i with respect to the parameter u_i.

2.3. G^1 continuity of a network of patches

If one wants to join several patches together in a network of patches with G^1 continuity, it can happen that satisfying condition (I_C) for all edges can present serious difficulties. This problem has been mentioned by several authors, first by Van Wijk (Van Wijk, 1986) and is called "*vertex consistency problem*". At a vertex, where n patches meet, G^1 continuity can generally not be achieved by simply solving the linear system of n equations (I_C). This system can have singularities, which are not easy to overcome. At such a vertex, the G^1 continuity is directly related to the twists (the second order mixed partial derivatives at a patch corner). For polynomial patches, which lie in the continuity class C^2, both twists are identical:

$$\frac{\partial^2 M^i}{\partial u_i \partial u_{i+1}}(0,0) = \frac{\partial^2 M^i}{\partial u_{i+1} \partial u_i}(0,0), \quad i = 1,\ldots,n.$$

Therefore, additional conditions at the patch corner, which involve the twists, have to be satisfied for G^1 continuity of a network of patches:

$$\begin{aligned} &\nu_i(0)M^i_{u_iu_{i+1}}(0,0) + \mu_i(0)M^{i-1}_{u_{i-1}u_i}(0,0) \\ &\quad = \Phi_i'(0)M^i_{u_i}(0,0) + \Phi_i(0)M^i_{u_iu_i}(0,0) + \nu_i'(0)M^i_{u_{i+1}}(0,0) + \mu_i'(0)M^{i-1}_{u_{i-1}}(0,0), \\ &i = 1,\ldots,n. \end{aligned} \qquad (I_T)$$

This system of equations is obtained by differentiating (I_C) with respect to u_i taken at $u_i = 0$.

Now, for solving the G^1 problem at a vertex two strategies can be employed:

- fix the boundary curves and solve (I_T) for the twists, or
- fix the twists and solve the n equations (I_T) for the boundary curves.

Both strategies, which should make n patches joining G^1 at a common vertex, will not give a solution in general for the same reason. They lead to linear systems of equations with a circulant matrix, which is singular if n is even and greater than 2 (Van Wijk, 1986; Peters, 1991b; Loop, 1994).

2.4. C^2-consistent boundary curves

A closer look to (I_T) shows that the right hand side only contains first and second derivatives of the patch boundary curves at the common vertex. Whether or not the linear system (I_T) can be solved depends therefore on the choice of the boundary curves. Boundary curves are called to be *C^2-consistent*, if the right hand side vectors

$[\ldots, M^i_{u_i}(0,0), \ldots]^{\mathrm{T}}$ and $[\ldots, M^i_{u_i u_i}(0,0), \ldots]^{\mathrm{T}}$ lie in the image space of the rank deficient system (I_T).

The present interpolation scheme solves the problem by first constructing C^2-consistent boundary curves of the patch network. This ensures G^1 continuity at the patch vertices by (I_T). In order to get an overall G^1 surface, (I_C) has to be satisfied between all adjacent patches. We therefore define cross-boundary tangents along each edge satisfying (I_C) and (I_T). It has to be noticed, that both steps are not independent, the values of the cross-boundary tangents at the vertices are already fixed by the boundary curves because of the following equality: $M^i_{u_{i+1}}(0,0) = M^{i+1}_{u_{i+1}}(0,0),\ i = 1, \ldots, n$.

3. Loop's scheme

Loop constructs sextic G^1 triangular Bézier patches in one-to-one correspondence with the input mesh faces. In this section we briefly recall the method of Loop in order to point out later the differences with our work. Details are in (Loop, 1994).

Boundary curves & vertex consistency

Around a vertex $\boldsymbol{p}$ of order n, with neighbor vertices $\boldsymbol{p}_i$ of order n_i, Loop uses the following scalar functions Φ_i, ν_i, μ_i in systems (I_C) and (I_T):

$$\Phi_i(u_i) = \cos\left(\frac{2\pi}{n}\right) B_0^2(u_i) + \frac{1}{2} B_1^2(u_i) + \left(1 - \cos\left(\frac{2\pi}{n_i}\right)\right) B_2^2(u_i), \quad \text{(quadratic)} \tag{1}$$

$$\nu_i(u_i) = \mu_i(u_i) = \frac{1}{2}. \quad \text{(constant)}$$

The following choice for the first three Bézier points $\boldsymbol{f}_i^0, \boldsymbol{f}_i^1, \boldsymbol{f}_i^2$ of the boundary curve between $\boldsymbol{p}$ and $\boldsymbol{p}_i$ enables us to find a solution to system (I_T) around $\boldsymbol{p}$:

$$\begin{aligned}
\boldsymbol{f}_i^0 &= \alpha \boldsymbol{p} + \frac{(1-\alpha)}{n} \sum_{j=1}^{n} \boldsymbol{p}_j, \\
\boldsymbol{f}_i^1 &= \alpha \boldsymbol{p} + \frac{1}{n} \sum_{j=1}^{n} \left(1 - \alpha + \beta \cos\left(\frac{2\pi(j-i)}{n}\right)\right) \boldsymbol{p}_j, \\
\boldsymbol{f}_i^2 &= \frac{1}{3}\boldsymbol{p} + \frac{1}{6}\boldsymbol{p}_{i-1} + \frac{1}{3}\boldsymbol{p}_i + \frac{1}{6}\boldsymbol{p}_{i+1}.
\end{aligned} \tag{2}$$

The boundary curve between $\boldsymbol{p}$ and $\boldsymbol{p}_i$ is of degree 4, and has control points $\boldsymbol{f}_i^0, \ldots, \boldsymbol{f}_i^4$, where $\boldsymbol{f}_i^3, \boldsymbol{f}_i^4$ are constructed as $\tilde{\boldsymbol{f}}_i^0, \tilde{\boldsymbol{f}}_i^1$ from the opposite vertex $\boldsymbol{p}_i$.

α and β in (2) are shape parameters. There is no shape parameter for $\boldsymbol{f}_i^2$. In fact, since the boundary curve has degree 4, the middle control point $\boldsymbol{f}_i^2$ must be computed symmetrically from both end-points.

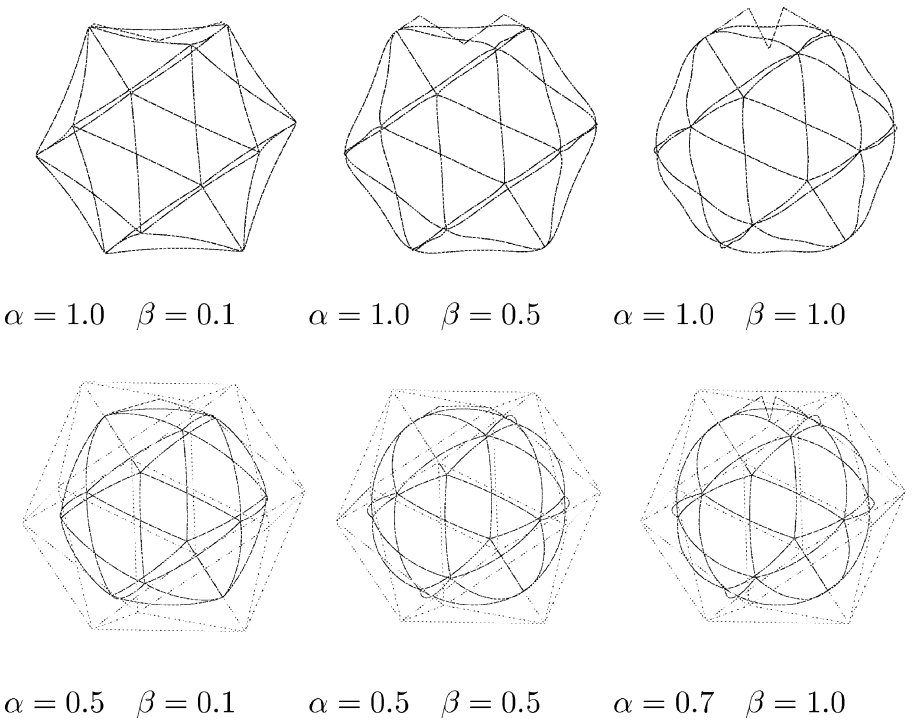


Fig. 2. A typical case where undulations in the curve network happen, when interpolating with Loop's patches. The top shows the interpolation case, $\alpha = 1$. To remove the oscillations, a smaller value of α has to be chosen, and the original mesh is not interpolated (bottom).

Cross-boundary tangents

The cross-boundary tangents are set to be equal

$$\begin{aligned} \frac{\partial H_i}{\partial u_{i+1}}(u_i, 0) &= \Phi_i(u_i)\frac{\partial H_i}{\partial u_i}(u_i, 0) + \Psi_i(u_i)\boldsymbol{V}_i(u_i), \qquad \text{(quintic)} \\ \frac{\partial H_{i-1}}{\partial u_{i-1}}(0, u_i) &= \Phi_i(u_i)\frac{\partial H_i}{\partial u_i}(u_i, 0) - \Psi_i(u_i)\boldsymbol{V}_i(u_i), \qquad \text{(quintic)} \end{aligned} \tag{3}$$

which ensures automatically that (I_C) is satisfied. The scalar function Ψ_i and the vector function $\boldsymbol{V}_i$ are built of minimal degree so as to interpolate the values of the cross-derivatives and the twists at the vertices $\boldsymbol{p}$ and $\boldsymbol{p}_i$:

$$\begin{aligned} \Psi_i(u_i) &= \sin\frac{2\pi}{n}(1 - u_i) + \sin\frac{2\pi}{n_i}u_i, \qquad \text{(linear)} \\ \boldsymbol{V}_i(u_i) &= \sum_{k=1}^{n} \boldsymbol{v}_i^k B_k^3(u_i), \qquad \text{(cubic)} \end{aligned} \tag{4}$$

where $\boldsymbol{v}_i^0 = \sum_{j=1}^n V_{ij}^0 \boldsymbol{p}_j$ and

$$\boldsymbol{v}_i^1 = \sum_{j=1}^{n} \kappa_i V_{ij}^0 + \boldsymbol{p}_j \begin{cases} \frac{2}{3}\frac{\Phi_i(0)}{\Psi_i(0)} & \text{if } j = i+1, \\ -\frac{2}{3}\frac{\Phi_i(0)}{\Psi_i(0)} & \text{if } j = i-1, \\ 0 & \text{otherwise,} \end{cases}$$

with

$$V_{ij}^0 = \frac{1}{n} 4\beta \sin\frac{2\pi(j-i)}{n}$$

and

$$\kappa_i = 1 - \frac{1}{3\Psi_i(0)}\left[\tan\frac{\pi}{n}\big(6\Phi_i(0) - \Phi_i'(0)\big) + \Psi_i'(0)\right].$$

$\boldsymbol{v}_i^2$ and $\boldsymbol{v}_i^3$ are constructed as $\tilde{\boldsymbol{v}}_i^0$, $\tilde{\boldsymbol{v}}_i^1$ from the opposite vertex.

Each triangular patch H must be of degree 6 because of the quintic cross-boundary tangent functions (3). From the boundary curves (2 times degree elevated) and the cross-boundary tangents the first two rows of Bézier control points of H are calculated. The remaining middle control point of each patch is chosen so that H has quintic precision. In two special cases Loop's patches are quintic (the three patch vertices have same order) or quartic (the three patch vertices are of order 6).

4. Regular 4-split

Subdivision of the domain into several pieces has been shown to be of benefit to interpolation by piecewise polynomial curves or tensor product surfaces. The polynomial degree can be kept low and additional degrees of freedom allow for shape improvements.

For the same purpose we split the domain triangles into 4 sub-triangles by joining the edge midpoints together, see Fig. 3. Each triangular *macro-patch* M, which interpolates the 3 vertices of a surface mesh triangle, will be a piecewise C^1 quintic surface.

For the following developments we first consider the macro-patch as a whole. The boundary curves and cross-boundary tangents are therefore *piecewise polynomial*

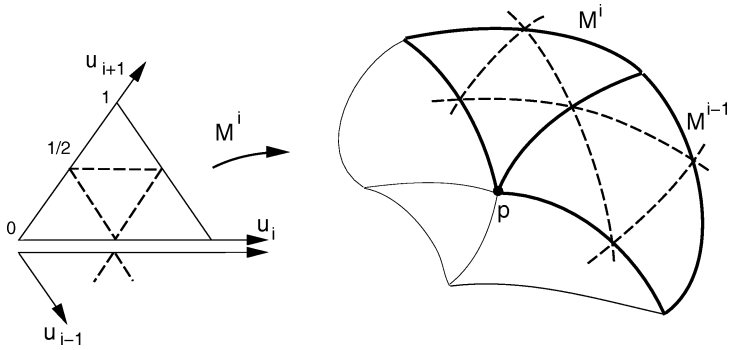


Fig. 3. 4-split of all domain triangles.

functions. The four sub-patches are then considered independently when filling-in the macro-patches with the C^1 quintic Bézier triangles.

4-splitting the domain triangles for parametric G^1 interpolation is the key issue of the present method. It doesn't cause additional problems, as one would probably think; in contrary! We shall point out, that we don't use the 4-split in order to solve the vertex consistency problem as the Clough–Tocher methods do. The advantages are obvious, because the number of degrees of freedom per macro-patch is increased. They can be used to perform interpolation of vertices and to efficiently control the shape. The additional vertices, which are created at the edge midpoints, are of order 6. But the vertex consistency problem is implicitly solved by the special construction of the boundary curves and the cross-boundary tangents of the macro-patches, as will be shown in Section 8.

5. Choice of scalar functions Φ_i, ν_i, μ_i

For the interpolating spline surface presented in this paper one of the most important targets is to keep the total degree of the patches as low as possible. If $M^i(u_i, u_{i+1})$ is a triangular surface of total parametric degree d, then $M^i_{u_i}(u_i, 0)$, $M^i_{u_{i+1}}(u_i, 0)$, and $M^{i-1}_{u_{i-1}}(0, u_i)$ are of degree $d-1$ in Eq. (I_C). When joining patches G^1 continuously, the conditions (I_C) and (I_T) must be satisfied. It is important to choose the scalar valued functions Φ_i, ν_i, μ_i such that they don't raise the degree of the final patches. Ideally this would mean to take Φ_i linear and ν_i, μ_i constant and the degree of the patches would not be raised when satisfying Eq. (I_C).

One of the main contributions of this paper is to show that we can make it possible. First important point is the choice of Φ_i. For locality reasons, Loop is not able to take Φ_i linear, he takes it quadratic, which finally leads to patches of degree six, one degree more than our proposal.

For symmetry reasons we choose $\nu_i = \mu_i = \frac{1}{2}$ and as a simplification we suppose that $\Phi^0 := \Phi_i(0)$ and $\Phi^1 := \Phi_i'(0)$ for $i = 1, \ldots, n$. These assumptions imply that the G^1 conditions now state as follows:

$$\Phi_i(u_i) M^i_{u_i}(u_i, 0) = \frac{1}{2} M^i_{u_{i+1}}(u_i, 0) + \frac{1}{2} M^{i-1}_{u_{i-1}}(0, u_i), \tag{II_C}$$

$$\nu_i(0) M^i_{u_i u_{i+1}}(0,0) + \mu_i(0) M^{i-1}_{u_{i-1} u_i}(0,0) = \Phi_i'(0) M^i_{u_i}(0,0) + \Phi_i(0) M^i_{u_i u_i}(0,0).$$

Varying i from 1 to n leads to the following linear system of equations:

$$T\bar{\boldsymbol{t}} = \Phi^1 \boldsymbol{r}^1 + \Phi^0 \boldsymbol{r}^2, \tag{II_T}$$

where

$$T = \begin{bmatrix} \frac{1}{2} & 0 & \cdots & & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & \cdots & & & 0 \\ & & \ddots & & & \\ 0 & \cdots & & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \cdots & & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix},$$

$$\boldsymbol{r}^1 = \begin{bmatrix} M^1_{u_1}(0,0) \\ \vdots \\ M^n_{u_n}(0,0) \end{bmatrix}, \qquad \boldsymbol{r}^2 = \begin{bmatrix} M^1_{u_1u_1}(0,0) \\ \vdots \\ M^n_{u_nu_n}(0,0) \end{bmatrix},$$

and $\bar{\boldsymbol{t}}$ is the vector of the twists. In (Loop, 1994) it was now shown, that it is possible to determine Φ^0 and Φ^1. For $u_i = 0$ it is easy to see that $M^i_{u_{i+1}}(0,0) = M^{i+1}_{u_{i+1}}(0,0)$. Eqs. (II_C) $(i = 1,\ldots,n)$ taken at $u_i = 0$ are therefore transformed into the following homogeneous system

$$\begin{bmatrix} \Phi^0 & -\frac{1}{2} & 0 & \ldots & 0 & -\frac{1}{2} \\ -\frac{1}{2} & \Phi^0 & -\frac{1}{2} & & & 0 \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & & & 0 \\ 0 & & & & & -\frac{1}{2} \\ -\frac{1}{2} & & & & -\frac{1}{2} & \Phi^0 \end{bmatrix} \begin{bmatrix} M^1_{u_1} \\ \\ \vdots \\ \\ M^n_{u_n} \end{bmatrix} = \boldsymbol{O}, \tag{5}$$

where the determinant is equal to

$$\prod_{i=0}^{n-1} \cos\left(\frac{2\pi k}{n}\right) - \Phi^0$$

for some integer k (Davis, 1979). A non-trivial solution exists if and only if $\Phi^0 = \cos(\frac{2\pi k}{n})$, where n is the order of the vertex $u_i = 0$. $k = 1$ was set to insure that the $M^i_{u_i}$ span a plane and are ordered properly, thus

$$\Phi^0 = \Phi_i(0) = \cos\left(\frac{2\pi}{n}\right). \tag{6}$$

In an analogous way, one obtains

$$\Phi_i(1) = 1 - \cos\left(\frac{2\pi}{n_i}\right), \tag{7}$$

where n_i is the order of the opposite vertex.

If one takes the functions Φ_i as linear blending functions, this would imply that $\Phi^1 = \Phi_i'(0)$ in (II_T) depends on the order n_i of the opposite vertex. This would make the algorithm global instead of local, which is not acceptable. Since (Loop, 1994) wanted a single polynomial patch per input triangle, he was forced to take Φ_i quadratic in order to separate vertex informations and to keep the algorithm local.

In our new method, the 4-splitting of domain triangles enables us to separate vertex informations by taking the functions Φ_i piecewise linear, continuous, defined on $[0, \frac{1}{2}]$ and $[\frac{1}{2}, 1]$, with $\Phi_i(\frac{1}{2}) = \frac{1}{2}$, as shown in Fig. 4.

$$\Phi_i(u_i) = \begin{cases} \cos\frac{2\pi}{n}(1-2u_i) + u_i & \text{for } u_i \in [0, \frac{1}{2}], \\ (1-u_i) + (1-\cos\frac{2\pi}{n_i})(2u_i - 1) & \text{for } u_i \in [\frac{1}{2}, 1]. \end{cases} \tag{8}$$

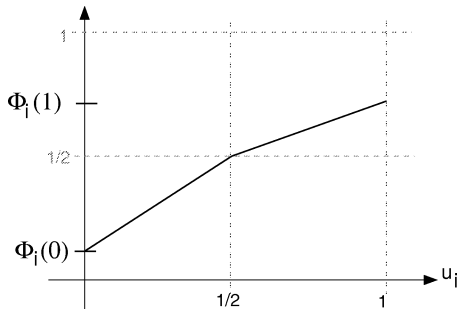


Fig. 4. Scalar valued function $\Phi_i(u_i)$, piecewise linear.

This choice is justified by the observations that $n = n_i$ implies $\Phi_i(1) = 1 - \Phi_i(0)$ (and therefore Φ_i is a single linear function) and $n = n_i = 6$ implies $\Phi_i(u_i) = \frac{1}{2}$ for all $u_i \in [0, 1]$.

This choice for Φ_i would not have been possible without 4-splitting the domain triangles.

6. Boundary curve network

The boundary curves of the macro-patches are constructed in correspondence to the edges of $\mathcal{M}$. This is the most important step in the surface construction method, because the shape of this curve network has great influence on the surface shape. The requirements on the boundary curves are the following:

- interpolating the vertices of $\mathcal{M}$,
- satisfying the G^1 conditions (II_C), (II_T) at the end points,
- keeping the surface scheme local.

The locality requirement imposes to construct the curves such that they satisfy (II_C), (II_T) at one vertex (end point) independently from the opposite vertex. The first and second derivatives at the curve's end points are involved in system (II_T). A polynomial curve which separates these informations of both end points should be of degree $\geqslant 5$. The advantage of the domain 4-split is now, that it allows to take *piecewise C^1 polynomial curves of degree* 3. Each boundary curve between two adjacent mesh vertices consists of 2 cubic pieces, which are constructed independently from each other.

Let denote the polynomial piece of the boundary curve between the neighboring vertices $\boldsymbol{v}$ of order n and the vertex $\boldsymbol{p}_i$ of order n_i in Bézier form by the control points $\boldsymbol{b}_0^i, \ldots, \boldsymbol{b}_3^i$, see Fig. 5. $\{0, \frac{1}{2}, 1\}$ is the subdivision of the parameter interval for the whole boundary curve. Around each vertex of $\mathcal{M}$ the control points $\boldsymbol{b}_0^i, \boldsymbol{b}_1^i, \boldsymbol{b}_2^i,\ i = 1, \ldots, n$, of all incident boundary curves can be constructed independently from the joining curve piece of the opposite vertices, i.e., the first and second derivatives can be isolated at each vertex. The "midpoints" $\boldsymbol{b}_3^i$ are constructed in order to have C^1 joints between the two curve pieces. These points correspond to the parameter $u_i = \frac{1}{2}$, i.e., the midpoint of an edge of $\mathcal{M}$, where the 4-split has been accomplished. The control points of the joining pieces $\boldsymbol{b}_0^k, \boldsymbol{b}_1^k, \boldsymbol{b}_2^k$ and

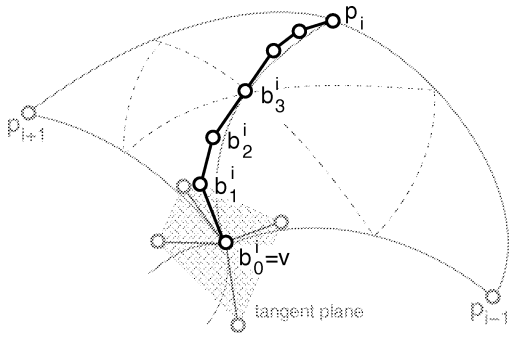


Fig. 5. Control points of the boundary curves at vertex $\boldsymbol{v}$.

$\boldsymbol{b}_3^k = \boldsymbol{b}_3^i$ are found when treating the boundary curve pieces incident in $\boldsymbol{p}_i$, where k is the index of $\boldsymbol{v}$ relative to the neighborhood of $\boldsymbol{v}$.

For simplification it is convenient to adopt a matrix notation:

$$\bar{\boldsymbol{b}}_0 := \begin{bmatrix} \boldsymbol{b}_0^1 \\ \vdots \\ \boldsymbol{b}_0^n \end{bmatrix}, \qquad \bar{\boldsymbol{b}}_1 := \begin{bmatrix} \boldsymbol{b}_1^1 \\ \vdots \\ \boldsymbol{b}_1^n \end{bmatrix}, \qquad \bar{\boldsymbol{b}}_2 := \begin{bmatrix} \boldsymbol{b}_2^1 \\ \vdots \\ \boldsymbol{b}_2^n \end{bmatrix}, \qquad \bar{\boldsymbol{p}} := \begin{bmatrix} \boldsymbol{p}_1 \\ \vdots \\ \boldsymbol{p}_n \end{bmatrix},$$

where $\bar{\boldsymbol{p}}$ is referred as to the *vertex neighborhood* of $\boldsymbol{v}$.

The boundary curves have to be constructed in order to have one-to-one correspondence between the macro-patches and the faces of $\mathcal{M}$. Therefore the boundary curve end points should correspond to the vertices of $\mathcal{M}$. Looking at vertex $\boldsymbol{v}$,

$$\boldsymbol{b}_0^i = \boldsymbol{v}, \quad i = 1, \ldots, n,$$

should hold for interpolation. This is a special case of the more general setting

$$\boldsymbol{b}_0^i = \alpha \boldsymbol{v} + (1-\alpha) \sum_{j=1}^{n} \frac{\boldsymbol{p}_j}{n}, \tag{9}$$

where $\alpha \in \mathbb{R}$ is a shape parameter controlling the interpolation. In matrix representation (9) corresponds to

$$\bar{\boldsymbol{b}}_0 = \alpha \bar{\boldsymbol{v}} + B^0 \bar{\boldsymbol{p}}, \tag{10}$$

where B^0 is a $n \times n$ matrix with $B_{ij}^0 = \frac{1-\alpha}{n}$, $i, j = 1, \ldots, n$, and $\bar{\boldsymbol{v}} = [\boldsymbol{v}, \ldots, \boldsymbol{v}]^{\mathrm{T}}$.

The points $\boldsymbol{b}_1^i$ define the tangent plane of $\mathcal{S}$ and the first derivative at the boundary curve end point:

$$\boldsymbol{r}_i^1 := M_{u_i}^i(0,0) = 6\big(\boldsymbol{b}_1^i - \boldsymbol{b}_0^i\big). \tag{11}$$

Additionally, they have to make $\boldsymbol{r}^1$ lying in the image space of T in (II_T) and to satisfy (II_C). A solution to that problem is to take $\boldsymbol{r}^1$ as the local averaging of the vertex

neighborhood of $\boldsymbol{v}$ which is known as *first order discrete Fourier approximation to* $\bar{\boldsymbol{p}}$ (Van Wijk, 1986; Peters, 1993; Loop, 1994):

$$\boldsymbol{r}_i^1 = \frac{6\beta}{n}\sum_{j=1}^{n}\cos\left(\frac{2\pi(j-i)}{n}\right)\boldsymbol{p}_j, \quad i = 1,\dots,n, \tag{12}$$

where β is a shape parameter controlling the magnitude of the tangent vectors.

Combining (11) with (10) and (12) gives

$$\bar{\boldsymbol{b}}_1 = \alpha\bar{\boldsymbol{v}} + B^1\bar{\boldsymbol{p}}, \tag{13}$$

where

$$B_{ij}^1 = \frac{1-\alpha+\beta\cos\left(\frac{2\pi(j-i)}{n}\right)}{n}, \quad i,j = 1,\dots,n.$$

The points $\boldsymbol{b}_2^i$ are related to the second derivatives at the boundary curve end point:

$$\boldsymbol{r}_i^2 := M^i_{u_i u_i}(0,0) = 12\left(\boldsymbol{b}_2^i - 2\boldsymbol{b}_1^i + \boldsymbol{b}_0^i\right) \tag{14}$$

and have to lie in the image space of T in (II_T).

It has been shown (Loop, 1994) that $\boldsymbol{d}^i = \frac{1}{6}(2\boldsymbol{v} + \boldsymbol{p}_{i-1} + 2\boldsymbol{p}_i + \boldsymbol{p}_{i+1})$ suffices to that condition. Since any affine combination of points, which lie in the image space of T also does, let define

$$\begin{aligned}\boldsymbol{b}_2^i &:= \gamma_0\boldsymbol{b}_0^i + \gamma_1\boldsymbol{b}_1^i + \gamma_2\boldsymbol{d}^i, \quad \gamma_0+\gamma_1+\gamma_2 = 1\\ &= \boldsymbol{b}_0^i + \gamma_1\left(\boldsymbol{b}_1^i - \boldsymbol{b}_0^i\right) + \gamma_2\left(\boldsymbol{d}^i - \boldsymbol{b}_0^i\right),\end{aligned}$$

where $\gamma_0, \gamma_1, \gamma_2$ are shape parameters controlling the value of the second derivative. The matrix expression is given by

$$\bar{\boldsymbol{b}}_2 = \left[(\gamma_0+\gamma_1)\alpha + \frac{\gamma_2}{3}\right]\bar{\boldsymbol{v}} + B^2\bar{\boldsymbol{p}}, \tag{15}$$

where

$$B_{ij}^2 = \frac{(\gamma_0+\gamma_1)(1-\alpha)+\gamma_1\beta\cos\left(\frac{2\pi(j-i)}{n}\right)}{n} + \gamma_2\begin{cases}1/6 & \text{if } j = i-1, i+1,\\ 1/3 & \text{if } j = i\ ,\\ 0 & \text{otherwise.}\end{cases}$$

The boundary curves have to be C^1-continuous at the knot $u_i = 1/2$ in order to get continuous cross boundary tangents later, which results in

$$\boldsymbol{b}_3^i = \frac{1}{2}\left(\boldsymbol{b}_2^i + \boldsymbol{b}_2^k\right), \tag{16}$$

where $\boldsymbol{b}_2^k$ belongs to the joining curve piece constructed from the vertex neighborhood of $\boldsymbol{p}_i$.

The piecewise cubic boundary curves of the macro-patches of $\mathcal{S}$ can now be calculated by using Eqs. (10), (13), (15) and (16) for each vertex $\boldsymbol{v}$ of $\mathcal{M}$. They form a C^2-consistent curve network.

The first and second derivatives at the corners, $\boldsymbol{r}_i^1$, $\boldsymbol{r}_i^2$, lie in the image space of T. It is now possible to solve (II_T) for the twist

$$\bar{\boldsymbol{t}} = \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_n \end{bmatrix}, \quad \text{with } \boldsymbol{t}_i = M^i_{u_i u_{i+1}}(0,0),\ i = 1,\dots,n,$$

by observing that the control points (10), (13), (15) are constructed such that they lie in the image space of T, i.e., there exist some points $\widetilde{\boldsymbol{b}}_0^i, \widetilde{\boldsymbol{b}}_1^i, \widetilde{\boldsymbol{b}}_2^i$ such that

$$T\widetilde{\boldsymbol{b}}_0^i = \boldsymbol{b}_0^i, \qquad T\widetilde{\boldsymbol{b}}_1^i = \boldsymbol{b}_1^i, \qquad T\widetilde{\boldsymbol{b}}_2^i = \boldsymbol{b}_2^i.$$

Due to the simple structure of the matrix T, it is easy to verify that

$$\begin{aligned}
\widetilde{\boldsymbol{b}}_0^i &= \frac{1}{2}\big(\boldsymbol{b}_0^i + \boldsymbol{b}_0^{i-1}\big), \\
\widetilde{\boldsymbol{b}}_1^i &= \alpha\boldsymbol{v} + \sum_{j=1}^{n} \frac{(1-\alpha) + \beta\big[\cos\big(\frac{2\pi(j-i)}{n}\big) + \tan\big(\frac{\pi}{n}\big)\sin\big(\frac{2\pi(j-i)}{n}\big)\big]}{n}\boldsymbol{p}_j, \\
\widetilde{\boldsymbol{b}}_2^i &= \gamma_0\boldsymbol{b}_0^i + \gamma_1\widetilde{\boldsymbol{b}}_1^i + \frac{1}{3}\gamma_2(\boldsymbol{v} + \boldsymbol{p}_i + \boldsymbol{p}_{i+1}),
\end{aligned}$$

are solutions of these three equations.

It follows that

$$\begin{aligned}
\Phi^1\boldsymbol{r}^1 + \Phi^0\boldsymbol{r}^2 &= 6\Phi^1\big(\boldsymbol{b}_1^i - \boldsymbol{b}_0^i\big) + 24\Phi^0\big(\boldsymbol{b}_2^i - 2\boldsymbol{b}_1^i + \boldsymbol{b}_0^i\big) \\
&= \big(-6\Phi^1 + 24\Phi^0\big)\boldsymbol{b}_0^i + \big(6\Phi^1 - 48\Phi^0\big)\boldsymbol{b}_1^i + 24\Phi^0\boldsymbol{b}_2^i \\
&= T\big[\big(-6\Phi^1 + 24(1+\gamma_0)\Phi^0\big)\boldsymbol{b}_0^i + \big(6\Phi^1 + (-48 + 24\gamma_1)\Phi^0\big)\widetilde{\boldsymbol{b}}_1^i \\
&\quad + 8\gamma_2\Phi^0(\boldsymbol{v} + \boldsymbol{p}_i + \boldsymbol{p}_{i+1})\big].
\end{aligned}$$

From Eq. (II_T) the following expression of the twists is obtained:

$$\begin{aligned}
\boldsymbol{t}_i &= 8\gamma_2\Phi^0(1-3\alpha)\boldsymbol{v} \\
&\quad + \sum_{j=1}^{n} \frac{-24\gamma_2\Phi^0(1-\alpha)}{n} \\
&\quad + \frac{(6\Phi^1 + (24\gamma_1 - 48)\Phi^0)\beta\big[\cos\frac{2\pi(j-i)}{n} + \tan\big(\frac{\pi}{n}\big)\sin\big(\frac{2\pi(j-i)}{n}\big)\big]}{n}\boldsymbol{p}_j \\
&\quad + 8\gamma_2\Phi^0(\boldsymbol{p}_i + \boldsymbol{p}_{i+1}), \quad i = 1,\dots,n.
\end{aligned} \tag{17}$$

Since the method of this paper is an interpolation scheme, $\alpha = 1$ is generally chosen. In order to avoid undulations of the boundary curves, for each vertex a set of three free shape parameters $\beta, \gamma_1, \gamma_2$ ($\gamma_0 = 1 - \gamma_1 - \gamma_2$) is available. As mentioned above, β controls the magnitude of the tangents and γ_1, γ_2 the second derivatives and therefore the shape of the curves, see also Section 9.

7. Cross boundary tangents

Once C^2-consistent boundary curves have been found, the second step in constructing a network of G^1 continuous patches is to define the cross-boundary tangents $M^i_{u_{i+1}}(u_i, 0)$ and $M^{i-1}_{u_{i-1}}(0, u_i)$ for each boundary curve of the curve network. The conditions on them are three

- satisfy the G^1 condition (II_C) along the boundary curve,
- satisfy the twist constraint at the end points,
- be consistent to the curve network.

With the curve network, the values of the cross-boundary tangent functions at the corners are already fixed, see Fig. 6.

A convenient way to define cross-boundary tangents that ensure G^1 continuity is the following:

$$\begin{aligned} M^i_{u_{i+1}}(u_i, 0) &= \Phi_i(u_i) M^i_{u_i}(u_i, 0) + \Psi_i(u_i) \boldsymbol{V}_i(u_i), \\ M^{i-1}_{u_{i-1}}(0, u_i) &= \Phi_i(u_i) M^i_{u_i}(u_i, 0) - \Psi_i(u_i) \boldsymbol{V}_i(u_i), \end{aligned} \tag{18}$$

where Ψ_i is some scalar function and $\boldsymbol{V}_i$ some vector function.

To see that (18) implies (II_c), simply add the two equations in (18). To see that (II_c) implies (18), choose $\Psi_i(u_i)\boldsymbol{V}_i(u_i) = \Phi_i(u_i) M^i_{u_i}(u_i, 0) - M^{i-1}_{u_{i-1}}(0, u_i) = -\Phi_i(u_i) M^i_{u_i}(u_i, 0) + M^i_{u_{i+1}}(u_i, 0)$.

The degree of Ψ_i and $\boldsymbol{V}_i$ in (18) decides now about the degree of the surface $\mathcal{S}$. The product $\Phi_i M^i_{u_i}$ is piecewise C^0 cubic, see (8) and Section 6. Therefore, $\Psi_i \boldsymbol{V}_i$ should not be of degree higher than 3. Due to the domain 4-split it will again be possible to construct these functions continuous and piecewise polynomial of degree 1 and 2.

The function values $\Psi_i(0)$ and $\boldsymbol{V}_i(0)$ are now determined following (Loop, 1994). The cross-boundary tangents have to correspond to the tangents of the boundary curve tangents at the end points. The first equation of (18) evaluated at $u_i = 0$, gives

$$\begin{aligned} & M^i_{u_{i+1}}(0, 0) = \Phi_i(0) M^i_{u_i}(0, 0) + \Psi_i(0) \boldsymbol{V}_i(0) \\ & \quad \Leftrightarrow \boldsymbol{r}^1_{i+1} = \Phi^0 \boldsymbol{r}^1_i + \Psi^0 \boldsymbol{V}_i(0) \end{aligned} \tag{19}$$

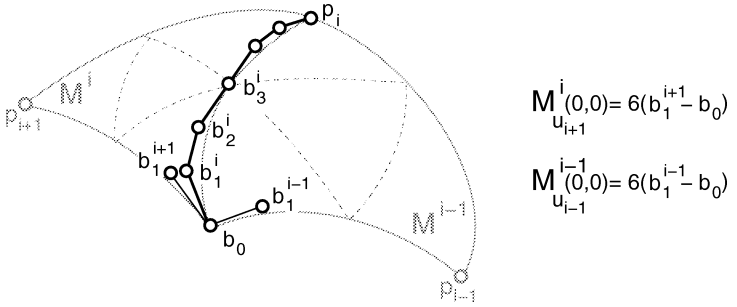


Fig. 6. Cross-boundary tangents at the vertices are fixed by the boundary-curves.

in terms of Section 5, with the simplifying assumption $\Psi_i(0) = \Psi^0$ for all i. Expanding (19) by using (12) and (6) results in

$$\Psi^0 \boldsymbol{V}_i(0) = \sin\left(\frac{2\pi}{n}\right) \sum_{j=1}^{n} \frac{6\beta}{n} \sin\frac{2\pi(j-i)}{n} \boldsymbol{p}_j.$$

An appropriate choice of Ψ^0 is therefore

$$\psi^0 = \sin\left(\frac{2\pi}{n}\right).$$

From the opposite end point, the product $\Psi_i(1)\boldsymbol{V}_i(1)$ can be obtained analogously, which gives $\Psi_i(1) = \sin\frac{2\pi}{n_i}$, where n_i is the order of vertex $\boldsymbol{p}_i$. Hence the function Ψ_i can be chosen linear, which is minimal degree:

$$\Psi_i(u_i) = \sin\left(\frac{2\pi}{n}\right)(1-u_i) + \sin\left(\frac{2\pi}{n_i}\right)u_i, \quad u_i \in [0,1]. \tag{20}$$

The function $\boldsymbol{V}_i$ cannot be taken linear, because it's derivative depends on the twists. When differentiating the first equation of (18) with respect to u_i and evaluating at $u_i = 0$

$$\begin{aligned} M^i_{u_{i+1}u_i}(0,0) = {} & \Phi_i'(0) M^i_{u_i}(0,0) + \Phi_i(0) M^i_{u_i u_i}(0,0) + \Psi_i'(0)\boldsymbol{V}_i(0) \\ & + \Psi_i(0)\boldsymbol{V}_i'(0). \end{aligned} \tag{21}$$

The derivative $\boldsymbol{V}_i'(0)$ appears in relation to the twist $\boldsymbol{t}_i = M^i_{u_i u_{i+1}}(0,0) = M^i_{u_{i+1}u_i}(0,0)$. It explains, why the cross-boundary tangents have to be constructed subject to the twists. From (21) one gets therefore

$$\boldsymbol{V}_i{}'(0) = \frac{1}{\Psi^0}\left[\boldsymbol{t}_i - \Phi^1 \boldsymbol{r}_i^1 - \Phi^0 \boldsymbol{r}_i^2 - \Psi_i^1 \boldsymbol{V}_i(0)\right], \tag{22}$$

where $\Psi_i^1 = \Psi_i'(0) = \sin(\frac{2\pi}{n_i}) - \sin(\frac{2\pi}{n})$ depends on i.

$\boldsymbol{V}_i(1)$ and $\boldsymbol{V}_i{}'(1)$ are known from the opposite vertex $\boldsymbol{p}_i$. A Hermite interpolation of these four values $\boldsymbol{V}_i(0), \boldsymbol{V}_i{}'(0), \boldsymbol{V}_i(1), \boldsymbol{V}_i{}'(1)$ would result in a cubic polynomial. The domain 4-split of the present method allows to lower the degree by one by taking $\boldsymbol{V}_i$ as a piecewise C^0 quadratic function requiring that $\boldsymbol{V}_i(\frac{1}{2}^-) = \boldsymbol{V}_i(\frac{1}{2}^+)$. In fact, as will be shown in Section 8.3, it is only required that $\boldsymbol{V}_i$ is C^0-continuous.

For the quadratic piece of $\boldsymbol{V}_i$ corresponding to the vertex $\boldsymbol{v}$ one gets in terms of Bernstein–Bézier representation the following control points:

$$\begin{aligned} \boldsymbol{v}_0^i &= \sum_{j=1}^{n} \frac{6\beta}{n} \sin\frac{2\pi(j-i)}{n} \boldsymbol{p}_j, \\ \boldsymbol{v}_1^i &= \sum_{j=1}^{n} \frac{1}{\psi_i^0}\left[\left(6\phi^1 - 48\phi^0 + 24\phi^0\right)\tan\left(\frac{\pi}{n}\right) - 6\psi_i^1\right] \\ &\quad \times \frac{\beta}{n}\sin\left(\frac{2\pi(j-i)}{n}\right)\boldsymbol{p}_j + \frac{4}{\psi_i^0}\gamma_2\phi^0(\boldsymbol{p}_{i+1} - \boldsymbol{p}_{i-1}), \\ \boldsymbol{v}_2^i &\quad \text{free subject to } \boldsymbol{V}_i\left(\frac{1}{2}^-\right) = \boldsymbol{V}_i\left(\frac{1}{2}^+\right). \end{aligned} \tag{23}$$

In matrix form, the control points of $\boldsymbol{V}_i$ are given by

$$\bar{\boldsymbol{v}}_0 = V^0 \bar{\boldsymbol{p}},$$
$$\bar{\boldsymbol{v}}_1 = V^1 \bar{\boldsymbol{p}},$$

where

$$V^0_{ij} = \frac{6\beta}{n} \sin\left(\frac{2\pi(j-i)}{n}\right), \quad i, j = 1, \ldots, n,$$
$$V^1_{ij} = \frac{1}{\psi^0_i}\left[\left(6\phi^1 - 48\phi^0 + 24\phi^0\right)\tan\left(\frac{\pi}{n}\right) - 6\psi^1_i\right]\frac{\beta}{n}\sin\left(\frac{2\pi(j-i)}{n}\right) + \frac{4}{\psi^0_i}\gamma_2\phi^0 \begin{cases} 1 & \text{if } j = i+1, \\ -1 & \text{if } j = i-1. \end{cases}$$

As written above, $\boldsymbol{V}_i$ is only required to be C^0-continuous, and therefore the value of $\boldsymbol{v}^i_2$ is free. Nevertheless, in the example shown in Section 9, we have chosen C^1-continuous $\boldsymbol{V}_i$ functions by taking $\boldsymbol{v}^i_2 = \frac{1}{2}\boldsymbol{v}^i_1 + \frac{1}{2}\boldsymbol{v}^k_1$, where $\boldsymbol{v}^k_1$ is known from the opposite vertex $\boldsymbol{p}_i$.

Piecewise cubic cross-boundary tangents have been constructed in this section. However, the surface will only be piecewise quintic, because up to now, it is an open question how to use the degrees of freedoms in order to obtain a local piecewise quartic surface. This is subject of current research.

8. Macro-patches in Bézier form

From now up, the macro-patches are considered individually. The domain 4-split leads to the construction of 4 triangular patches per macro-patch for which the Bézier control points will be given in this section. The border and first inner row of control points of the macro-patch can be found from the boundary curves (Section 6) and the cross-boundary tangents (Section 7). They ensure the G^1-join to the neighboring macro-patches. In order to have an overall visually smooth surface, the remaining inner control points are used to join the 4 sub-patches C^1-continuously. Six control points per macro-patch remain free for local shape control.

8.1. Notations

A triangular Bézier patch of degree d is given by

$$\boldsymbol{B}(u, v, w) = \sum_{\substack{i+j+l=d \\ i,j,l \geqslant 0}} \boldsymbol{b}_{(i,j,l)} B^d_{i,j,l}(u, v, w), \quad u + v + w = 1,$$

where $u, v, w \in [0, 1]$ are the barycentric coordinates of a point inside the domain triangle, and $\boldsymbol{b_i}$ are the Bézier control points. The basis functions $B^d_{i,j,l}(u, v, w) = \frac{i!j!l!}{d!}u^i v^j w^l$ are known as generalized Bernstein polynomials. For more details about triangular Bézier patches, see (Farin, 1997; Hoschek and Lasser, 1993).

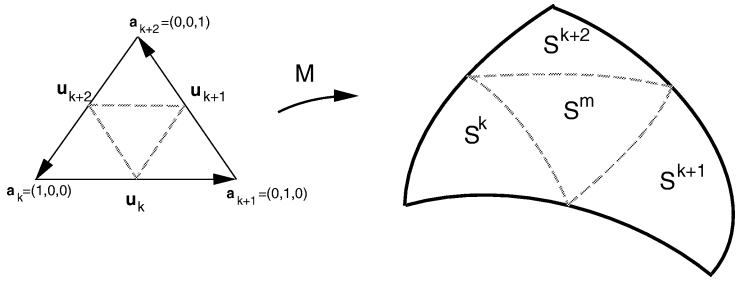


Fig. 7. Parameterization of the macro-patch M, labeling of sub-patches and derivative directions.

The 4 triangular Bézier patches of degree 5 which compose the macro-patch M are denoted by S^1, S^2, S^3, S^m and are parameterized as in Fig. 7.

The Bézier control points of M are therefore denoted resp. by $\boldsymbol{s}^1_{(i,j,l)}$, $\boldsymbol{s}^2_{(i,j,l)}$, $\boldsymbol{s}^3_{(i,j,l)}$, $\boldsymbol{s}^m_{(i,j,l)}$, where $i+j+l=5$. $\boldsymbol{u}_k = \boldsymbol{a}_{k+1} - \boldsymbol{a}_k$ are vectors between the domain vertices and define the directional derivatives $[\boldsymbol{D}_{\boldsymbol{u}_k} M](u,v,w)$ of M. Furthermore, let $E_k(u) = \boldsymbol{a}_k(1-u) + \boldsymbol{a}_{k+1}u$, for $u \in [0,1]$, define an edge function and let n_k be the order of the mesh vertex which is interpolated by $M(E_k(0))$.

8.2. Finding boundary and first (row) derivative control points of M

Let denote the piecewise cubic boundary curve by

$$M\big(E_k(u)\big) = \begin{cases} \sum_{i=0}^3 \boldsymbol{b}_i^L B_i^3(2u) & \text{for } u \in [0, \frac{1}{2}], \\ \sum_{i=0}^3 \boldsymbol{b}_i^R B_i^3(2u-1) & \text{for } u \in [\frac{1}{2}, 1], \end{cases} \quad k = 1,2,3, \tag{24}$$

where $\boldsymbol{b}_i^L, \boldsymbol{b}_i^R$ are the control points of the two curve pieces, computed in Section 6.

The cross-boundary tangent $M^i_{u_{i+1}}(u_i, 0)$ of Section 7 is here given by

$$[-D_{\boldsymbol{u}_{k+2}} M]\big(E_k(u)\big) = \Phi_k(u)[D_{\boldsymbol{u}_k} M]\big(E_k(u)\big) + \Psi_k(u)\boldsymbol{V}_k(u), \tag{25}$$

where

$$[D_{\boldsymbol{u}_k} M](E_k(u)) = \begin{cases} 6\sum_{i=0}^2 (\boldsymbol{b}_{i+1}^L - \boldsymbol{b}_i^L) B_i^2(2u) & \text{for } u \in [0, \frac{1}{2}], \\ 6\sum_{i=0}^2 (\boldsymbol{b}_{i+1}^R - \boldsymbol{b}_i^R) B_i^2(2u-1) & \text{for } u \in [\frac{1}{2}, 1], \end{cases} \quad k = 1,2,3, \tag{26}$$

is the derivative of $M(E_k(u))$ along the edge $\boldsymbol{u}_k$,

$$\boldsymbol{V}_k(u) = \begin{cases} \sum_{i=0}^2 \boldsymbol{v}_i^L B_i^2(2u) & \text{for } u \in [0, \frac{1}{2}], \\ \sum_{i=0}^2 \boldsymbol{v}_i^R B_i^2(2u-1) & \text{for } u \in [\frac{1}{2}, 1], \end{cases} \quad k = 1,2,3, \tag{27}$$

is the cross-derivative function of Section 7, and Φ_k, Ψ_k are the scalar functions defined, respectively, in Sections 3 and 7 by

$$\Phi_k(u) = \begin{cases} \cos\frac{2\pi}{n_k}(1-2u) + u & \text{for } u \in [0, \frac{1}{2}], \\ (1-u) + (1-\cos\frac{2\pi}{n_{k+1}})(2u-1) & \text{for } u \in [\frac{1}{2}, 1], \end{cases} \tag{28}$$

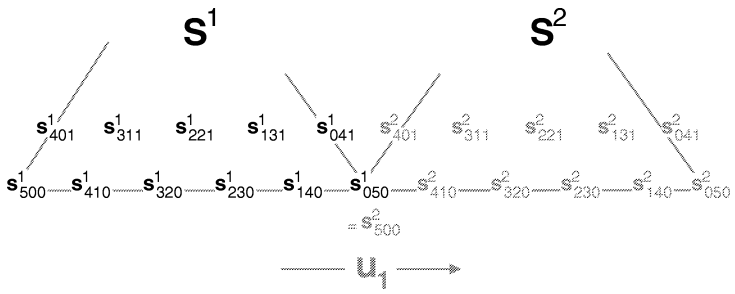


Fig. 8. Boundary and first derivative control points of M corresponding to boundary $\boldsymbol{u}_1$.

$$\Psi_k(u) = \sin\left(\frac{2\pi}{n_k}\right)(1-u) + \sin\left(\frac{2\pi}{n_{k+1}}\right)u, \quad u \in [0,1]. \tag{29}$$

Let us now consider the boundary of M corresponding to $\boldsymbol{u}_1$ which is common to the patches S^1 and S^2. The control points are labeled as in Fig. 8.

A double degree elevation of (24) results in the control points of the piecewise C^1 quintic Bézier curve, which is the boundary curve of M corresponding to direction $\boldsymbol{u}_1$:

$$\begin{aligned}
\boldsymbol{s}^1_{(5,0,0)} &= \boldsymbol{b}^L_0,\\
\boldsymbol{s}^1_{(4,1,0)} &= \frac{2}{5}\boldsymbol{b}^L_0 + \frac{3}{5}\boldsymbol{b}^L_1,\\
\boldsymbol{s}^1_{(3,2,0)} &= \frac{1}{10}\boldsymbol{b}^L_0 + \frac{3}{5}\boldsymbol{b}^L_1 + \frac{3}{10}\boldsymbol{b}^L_2,\\
\boldsymbol{s}^1_{(2,3,0)} &= \frac{3}{10}\boldsymbol{b}^L_1 + \frac{3}{5}\boldsymbol{b}^L_2 + \frac{1}{10}\boldsymbol{b}^L_3,\\
\boldsymbol{s}^1_{(1,4,0)} &= \frac{3}{5}\boldsymbol{b}^L_2 + \frac{2}{5}\boldsymbol{b}^L_3, \qquad \boldsymbol{s}^1_{(0,5,0)} = \boldsymbol{b}^L_3.
\end{aligned} \tag{30}$$

$\boldsymbol{s}^2_{(5-i,i,0)}$, $i = 0,\dots,5$, are found analogously from $\boldsymbol{b}^R_j$, $j = 0,\dots,5$.

The first row of inner control points $\boldsymbol{s}^1_{(4-i,i,0)}, \boldsymbol{s}^2_{(4-i,i,0)}$, $i = 0,\dots,4$, of M can be found by the cross-boundary derivatives

$$\begin{aligned}
&[-D_{\boldsymbol{u}_3}M]\big(E_1(u)\big)\\
&\quad= \begin{cases} 10\sum_{i=0}^4 (\boldsymbol{b}^L_{(4-i,i,1)} - \boldsymbol{b}^L_{(5-i,i,0)}) B^4_i(2u) & \text{for } u \in [0, \frac{1}{2}],\\ 10\sum_{i=0}^4 (\boldsymbol{b}^R_{(4-i,i,1)} - \boldsymbol{b}^R_{(5-i,i,0)}) B^4_i(2u-1) & \text{for } u \in [\frac{1}{2}, 1]. \end{cases}
\end{aligned} \tag{31}$$

The left hand-side of (31) can be found by combining (25) with (26)–(29). It is piecewise cubic and must be degree elevated, before rearranging of (31) finally leads to

$$\begin{aligned}
\boldsymbol{s}^1_{(4,0,1)} &= \boldsymbol{s}^1_{(5,0,0)} - \frac{3c_1}{5}\boldsymbol{b}^L_0 + \frac{3c_1}{5}\boldsymbol{b}^L_1 + \frac{s_1}{10}\boldsymbol{v}^L_0,\\
\boldsymbol{s}^1_{(3,1,1)} &= \boldsymbol{s}^1_{(4,1,0)} - \frac{6c_1+3}{40}\boldsymbol{b}^L_0 + \frac{3-6c_1}{40}\boldsymbol{b}^L_1 + \frac{3c_1}{10}\boldsymbol{b}^L_2 + \frac{3s_1+s_2}{80}\boldsymbol{v}^L_0 + \frac{s_1}{20}\boldsymbol{v}^L_1,\\
\boldsymbol{s}^1_{(2,2,1)} &= \boldsymbol{s}^1_{(3,2,0)} - \frac{1}{20}\boldsymbol{b}^L_0 - \frac{1+4c_1}{20}\boldsymbol{b}^L_1 + \frac{c_1+1}{10}\boldsymbol{b}^L_2 + \frac{c_1}{10}\boldsymbol{b}^L_3 + \frac{s_1+s_2}{120}\boldsymbol{v}^L_0
\end{aligned}$$

$$
\begin{aligned}
&+\frac{3s_1+s_2}{60}\boldsymbol{v}_1^L+\frac{s_1}{60}\boldsymbol{v}_2^L,\\
\boldsymbol{s}^1_{(1,3,1)} &= \boldsymbol{s}^1_{(2,3,0)}-\frac{3}{20}\boldsymbol{b}_1^L+\frac{3-6c_1}{40}\boldsymbol{b}_2^L+\frac{6c_1+3}{40}\boldsymbol{b}_3^L+\frac{s_1+s_2}{40}\boldsymbol{v}_1^L+\frac{3s_1+s_2}{80}\boldsymbol{v}_2^L,\\
\boldsymbol{s}^1_{(0,4,1)} &= \boldsymbol{s}^1_{(1,4,0)}-\frac{3}{10}\boldsymbol{b}_2^L+\frac{3}{10}\boldsymbol{b}_3^L+\frac{s_1+s_2}{20}\boldsymbol{v}_2^L,
\end{aligned}
\tag{32}
$$

and

$$
\begin{aligned}
\boldsymbol{s}^2_{(4,0,1)} &= \boldsymbol{s}^2_{(5,0,0)}-\frac{3}{10}\boldsymbol{b}_0^R+\frac{3}{10}\boldsymbol{b}_1^R+\frac{s_1+s_2}{20}\boldsymbol{v}_0^R,\\
\boldsymbol{s}^2_{(3,1,1)} &= \boldsymbol{s}^2_{(4,1,0)}-\frac{9-6c_2}{40}\boldsymbol{b}_0^R+\frac{3-6c_2}{40}\boldsymbol{b}_1^R+\frac{3}{20}\boldsymbol{b}_2^R+\frac{s_1+3s_2}{80}\boldsymbol{v}_0^R+\frac{s_1+s_2}{40}\boldsymbol{v}_1^R,\\
\boldsymbol{s}^2_{(2,2,1)} &= \boldsymbol{s}^2_{(3,2,0)}-\frac{1}{20}\boldsymbol{b}_0^R-\frac{2-c_2}{10}\boldsymbol{b}_1^R+\frac{5-4c_2}{20}\boldsymbol{b}_2^R+\frac{1}{20}\boldsymbol{b}_3^R+\frac{s_2}{60}\boldsymbol{v}_0^R\\
&\quad+\frac{s_1+3s_2}{60}\boldsymbol{v}_1^R+\frac{s_1+s_2}{120}\boldsymbol{v}_2^R,\\
\boldsymbol{s}^2_{(1,3,1)} &= \boldsymbol{s}^2_{(2,3,0)}-\frac{3-3c_2}{10}\boldsymbol{b}_1^R+\frac{3-6c_2}{40}\boldsymbol{b}_2^R+\frac{9-6c_2}{40}\boldsymbol{b}_3^R+\frac{s_2}{20}\boldsymbol{v}_1^R+\frac{s_1+3s_2}{80}\boldsymbol{v}_2^R,\\
\boldsymbol{s}^2_{(0,4,1)} &= \boldsymbol{s}^2_{(1,4,0)}-\frac{3-3c_2}{5}\boldsymbol{b}_2^R+\frac{3-3c_2}{5}\boldsymbol{b}_3^R+\frac{s_2}{10}\boldsymbol{v}_2^R,
\end{aligned}
\tag{33}
$$

where $c_j=\cos(\frac{2\pi}{n_j})$, $s_j=\sin(\frac{2\pi}{n_j})$. The control points corresponding to the boundaries $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ are obtained by shifting the indices in (30) and (32), (33) once and twice to the left.

8.3. Filling-in the macro-patches by piecewise quintic Bézier triangles

All control points, which are involved in joining the macro-patches pairwise G^1 are highlighted in Fig. 9. In this section, it will be shown that it is possible to join the 4 sub-patches S^1, S^2, S^3, S^m with C^1-continuity and how the remaining control points are used for that.

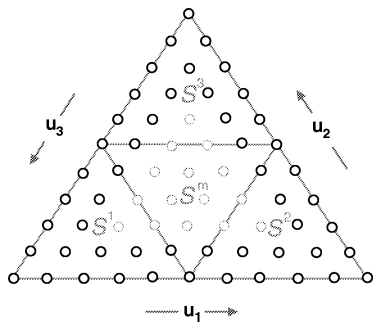


Fig. 9. Control points known from the boundary curves and their cross-boundary tangents, ensuring pairwise G^1 continuity between macro-patches.

Vertex consistency and C^1-continuity at the edge mid-points

Two questions should be answered for the general understanding of the present triangular interpolation scheme:

– The macro-triangle edge mid-points are vertices of order 6. Why don't they present the vertex consistency problem?
– Why using (the stronger) C^1 conditions for filling-in the macro-patches (instead of G^1 conditions)?

In fact, both questions can be answered simultaneously: it turns out that the cross-boundary tangents constructed in Section 7 already ensure continuity of the first partial derivatives at the edge mid-points. To prove this, we temporarily switch to the notations of Section 7. The partial derivatives around an edge mid-point are shown in Fig. 10.

We have to prove that $M^i_{u_{i+1}}(\frac{1}{2}^+, 0) = M^i_{u_{i+1}}(\frac{1}{2}^-, 0)$, $M^{i-1}_{u_{i-1}}(0, \frac{1}{2}^+) = M^{i-1}_{u_{i-1}}(0, \frac{1}{2}^-)$ (continuity at $\frac{1}{2}$ inside one macro-patch) and $M^i_{u_{i+1}}(\frac{1}{2}, 0) = M^{i-1}_{u_{i+1}}(0, \frac{1}{2})$, $M^i_{u_{i-1}}(\frac{1}{2}, 0) = M^{i-1}_{u_{i-1}}(0, \frac{1}{2})$ (continuity between the two macro-patches joining at the common edge). The other identities follow from the C^1-continuity of the common boundary curve.

The continuity of the partial derivatives inside one macro-patch can easily be seen from (18): since $\Phi_i(u_i), M^i_{u_i}(u_i, 0), \Psi_i(u_i), \boldsymbol{V}_i(u_i)$ are all continuous at $u_i = \frac{1}{2}$, then $M^i_{u_{i+1}}(u_i, 0)$ and $M^{i-1}_{u_{i-1}}(0, u_i)$ are also continuous at $u_i = \frac{1}{2}$.

It remains to prove the continuity of the partial derivatives between the joining macro-patches. We will prove the first identity, $M^i_{u_{i+1}}(\frac{1}{2}, 0) = M^{i-1}_{u_{i+1}}(0, \frac{1}{2})$, the second identity $M^i_{u_{i-1}}(\frac{1}{2}, 0) = M^{i-1}_{u_{i-1}}(0, \frac{1}{2})$ can be proven analogously.

Since $u_{i+1} = u_i - u_{i-1}$, it follows that $M^{i-1}_{u_{i+1}}(0, \frac{1}{2}) = M^{i-1}_{u_i}(0, \frac{1}{2}) - M^{i-1}_{u_{i-1}}(0, \frac{1}{2})$. In this last identity, we replace $M^{i-1}_{u_{i-1}}(0, \frac{1}{2})$ by its value from (18):

$$M^{i-1}_{u_{i+1}}\left(0, \frac{1}{2}\right) = M^{i-1}_{u_i}\left(0, \frac{1}{2}\right) - \left[\Phi_i\left(\frac{1}{2}\right) M^i_{u_i}\left(\frac{1}{2}, 0\right) - \Psi_i\left(\frac{1}{2}\right) \boldsymbol{V}_i\left(\frac{1}{2}\right)\right].$$

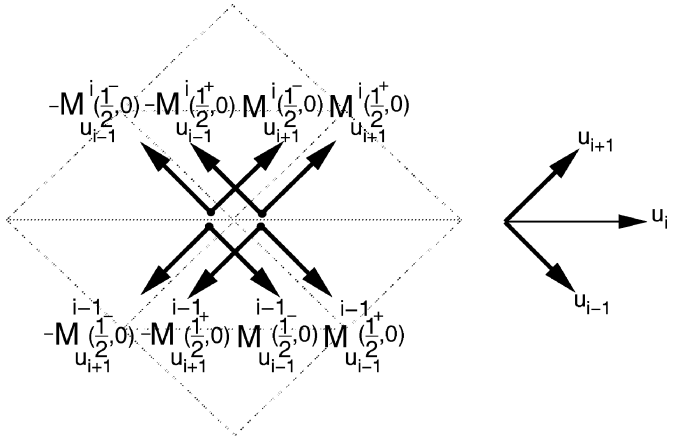


Fig. 10. Partial derivatives at the edge mid-points.

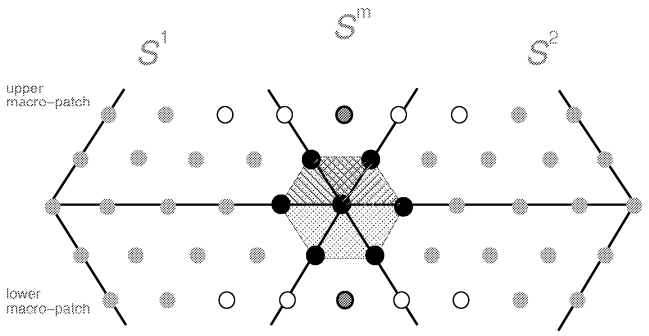


Fig. 11. The control points around macro-patch boundary mid-points form an affine transformation of a regular 6-gon.

But $\Phi_i(\frac{1}{2}) = \frac{1}{2}$, and $M^{i-1}_{u_i}(0, \frac{1}{2}) = M^i_{u_i}(\frac{1}{2}, 0)$ (the two macro-patches share the same common C^1-continuous boundary curve), therefore:

$$M^{i-1}_{u_{i+1}}\left(0, \frac{1}{2}\right) = \Phi_i\left(\frac{1}{2}\right) M^i_{u_i}\left(\frac{1}{2}, 0\right) + \Psi_i\left(\frac{1}{2}\right) \boldsymbol{V}_i\left(\frac{1}{2}\right) = M^{i+1}_{u_{i+1}}\left(\frac{1}{2}, 0\right).$$

Thus we have shown that all C^1-continuity conditions around the edge mid-points are already fulfilled by the cross-boundary tangents constructed in Section 7. Therefore there is no vertex-consistency problem at these points, and it is natural to use the stronger C^1-continuity conditions for filling-in the macro-patches. In other words, the 6 Bézier points around an edge mid-point form an affine transformation of a regular 6-gon, as shown in Fig. 11.

Computing the unknown Bézier points

The necessary and sufficient C^1-continuity conditions between two internal Bézier patches inside one macro-patch are shown in Fig. 12: all pairs of adjacent triangles in Fig. 12 must form a parallelogram.

From the previous proof, we know that the first and last pairs of adjacent triangles in Fig. 12 already form parallelograms.

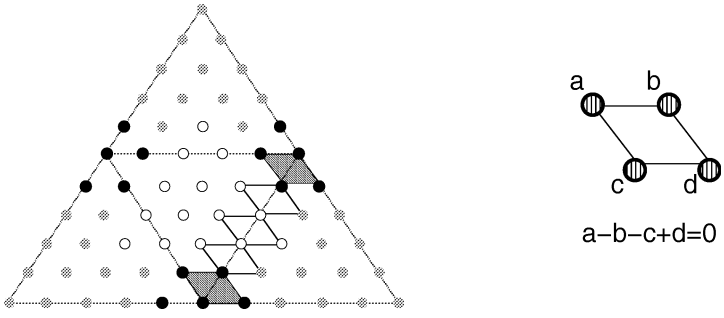


Fig. 12. C^1-conditions between two adjacent quintic Bézier patches inside one macro-patch.

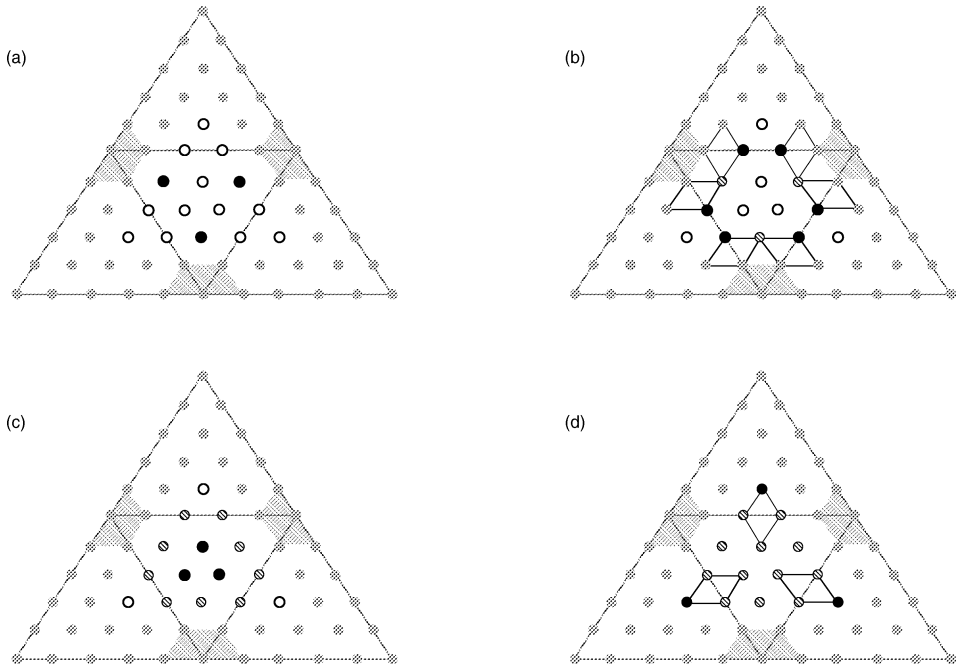


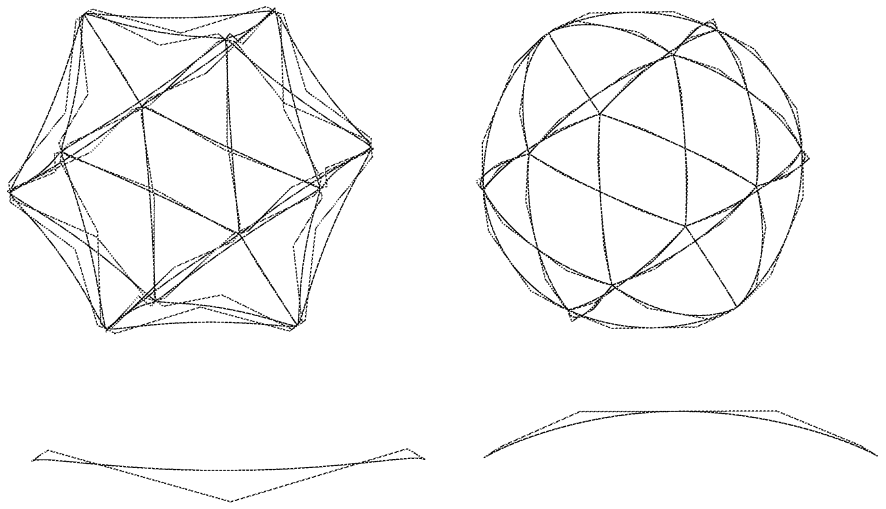
Fig. 13. Four steps for filling-in the macro-patch M with C^1-continuity: (a) choose the three twists of S^m, which are free for local shape control, (b) compute the third and fourth Bézier points along each edge using the C^1-continuity conditions, (c) choose the three last free Bézier points of S^m, which are also free shape parameters, (d) compute the remaining three Bézier points using the C^1-continuity conditions.

It remains to compute the free Bézier points such that the other three pair of triangles along each edge inside on macro-patch also form parallelograms. This is be done in four steps:

- choose the three twists points of the internal Bézier patch arbitrarily; these are free shape parameters (see Fig. 13(a)),
- compute the third and fourth Bézier points along each internal curve joining two Bézier patches using the second and fourth parallelogram conditions (see Fig. 13(b)),
- choose the remaining three unknown Bézier points of the central patches arbitrarily; these are free shape parameters (see Fig. 13(c)),
- compute the three remaining unknown Bézier points of the outer patches using the third parallelogram condition along each edge (see Fig. 13(d)).

9. Results

The first example shows how the new shape parameters γ_0, γ_1 ($\gamma_2 = 1 - \gamma_0 - \gamma_1$) can be used to remove unwanted undulations in the curve network interpolating the



Interpolation as in [10], $\alpha = 1$, $\beta = 0.1$

Interpolation with the new method, $\alpha = 1$, $\beta = 0.1$, $\gamma_0 = -3.7$, $\gamma_1 = 4.6$, $\gamma_2 = 0.1$

Fig. 14. Removing unwanted undulations in the interpolating curve network. The left part shows the interpolation of an icosahedron with the method of (Loop, 1994). The right part shows the interpolation with the new method presented in this article. The new shape parameters enable us to remove the undulations. The control-polygons are in red, and the boundary curves are in blue.

input triangular mesh. The left part of Fig. 14 shows the result from (Loop, 1994) when interpolating an icosahedron by choosing $\alpha = 1.0$. The value of β is 0.1. The right part shows the curve network from the new interpolant presented in this paper, with the same values for α and β, but with the new shape parameters $\gamma_0 = -3.7$, $\gamma_1 = 4.6$, $\gamma_2 = 0.1$. The boundary curves are blue and the control-polygons are red. The bottom part of Fig. 14 shows one of the boundary curve in blue, with its control-polygon in red. The method in (Loop, 1994) yields a single quartic control-polygon (bottom left), while our method yields two cubic control-polygons joining C^1-continuously (bottom right). Since the same β value was chosen in both methods, the first and second control-points at each end of the boundary curves are the same in both results. Choosing negative values for the new shape parameter γ_0 enables to smooth out the control-polygon and the associated boundary curve.

The color plate (Fig. 15) shows the interpolation of a deformed icosahedron ((a) and (b)) and of a triangular mesh with vertices of various orders ((c) and (d), vertices of orders 3, 4, 5, 6, and 8 are visible on these views). For the surfaces shown in the color plate, the six free control-points mentioned in section 8 were computed automatically based on the minimization of an local energy functional.

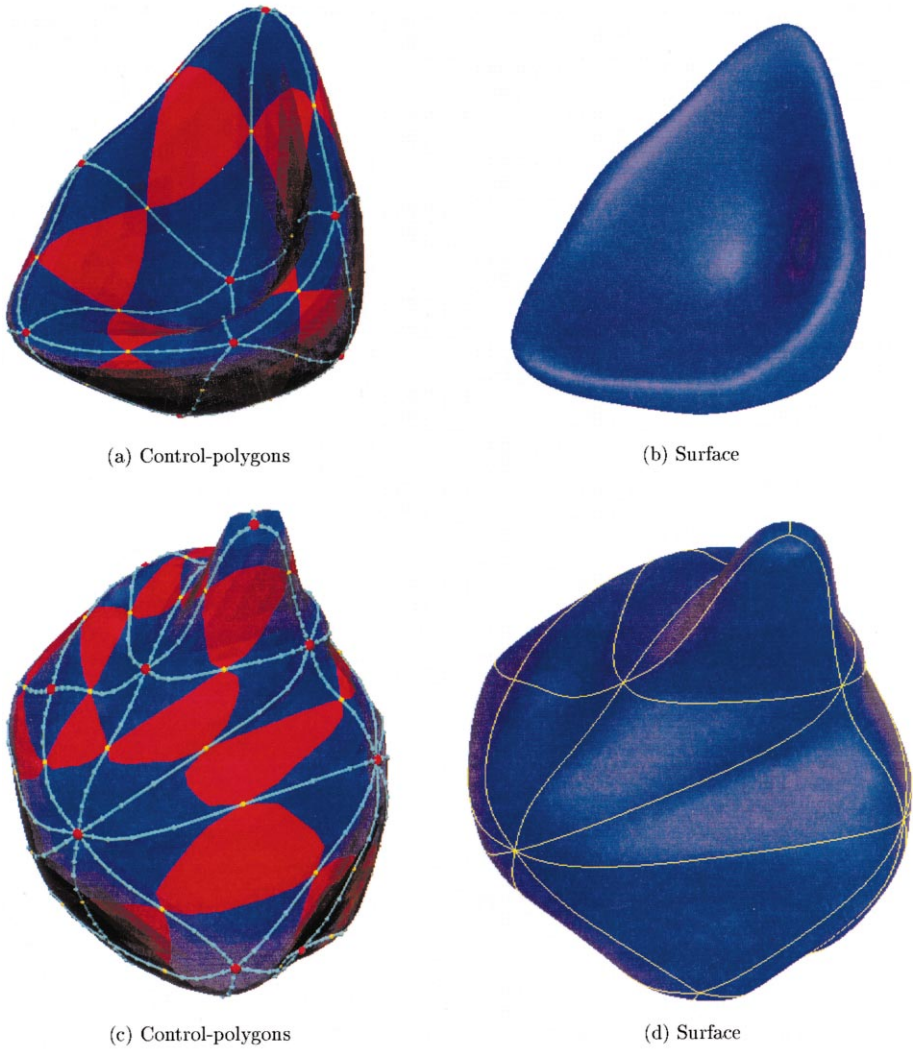


Fig. 15. Color plate: (a) and (b) show the interpolation of a deformed icosahedron, where all vertices have order 5, (c) and (d) show a surface interpolating a mesh with vertices of various orders. The left part shows the control-polygons of the Bézier patches (in blue and red), and of the boundary curves (in turquoise). The yellow points along the boundary curves are the edge middle-points. The interpolated points are red.

10. Discussion and conclusion

This paper has introduced a new method for interpolating a triangular surface mesh with arbitrary topological type by a piecewise quintic G^1 spline surface. A C^2-consistent

curve network solves the vertex consistency problem. Each macro-patch depends on only 3 vertex neighborhoods. The interpolant is therefore local.

The main contribution of this paper is to base the interpolation method on a 4-split of the domain triangles. It has been shown that it can be of benefit for several reasons: First, each macro-patch consists of four quintic Bézier patches. The number of control points is therefore high enough in order to solve the G^1 interpolation problem for arbitrary triangular meshes. Second, the boundary curves of the macro-patches are allowed to be piecewise polynomial curves. The degree of these curves can therefore be kept low.

Both characteristics of the 4-split imply finally that several degrees of freedom are available for additional shape control. Thus, for a given input mesh quite different shapes of the final surface are possible. This can go from a nearly flat surface imitating the control net to a rounded and blew up surface. An in some sense optimal shape would probably lie between it. The question of how to choose the degrees of freedom can not be treated here in detail, since too much possibilities exist depending of the result one wishes to obtain. However, in a first time one can proceed as follows: the degrees of freedom can be separated into two groups whether they act on the boundary curves or inside a macro-patch. The last group consists of the six control points per macro-patch which are highlighted in Figs. 13(a) and 13(c). A minimization of an energy norm keeps the scheme local and gives satisfactory results. The shape parameters of the boundary curves are the scalar values $\alpha, \beta, \gamma_0, \gamma_1 \in \mathbb{R}$ (see Chapter 6). For each mesh vertex one set of these parameters is available. α is generally set to be equal 1, otherwise the control net is not interpolated but approximated by the surface. β controls the length of the tangent vectors at the vertex and γ_0, γ_1 determine the inner shape of the curve pieces incident to a vertex. It is obvious that the curve network is of essential importance for the final surface shape. Up to now we proceeded by choosing one global $\beta > 0$ by hand for the entire surface and then by determining γ_0, γ_1 for each vertex by minimizing some constrained curve energy norm. This simple heuristic has both advantages, it keeps the scheme local and linear.

However, ongoing research focuses on completely automatic ways to determine these scalar parameters. The free control points can also be incorporated into these considerations subject to smooth the surface. Future work will also focus on the proper handling of open triangular meshes.

References

Davis, P. (1979), Circulant Matrices, Wiley.

Farin, G. (1982), A construction for visual C^1 continuity of polynomial surface patches, Computer, Graphics and Image Processing 20, 272–282.

Farin, G. (1997), Curves and Surfaces for Computer Aided Geometric Design, 4th edn., Academic Press, New York.

Gregory, J.A. (1986), N-sided surface patches, in: Gregory, J. (Ed.), The Mathematics of Surfaces, Clarendon Press, Oxford, 217–232.

Hagen, H. (1986), Geometric surface patches without twist constraints, Computer Aided Geometric Design 3, 179–184.

Hagen, H. and Pottmann, H. (1989), Curvature continuous triangular interpolants, in: Lyche, T. and Schumaker, L.L. (Eds.), Mathematical Methods in Computer Aided Geometric Design, Academic Press, New York, 373–384.

Herron, G. (1985), Smooth closed surfaces with discrete triangular interpolants, Computer Aided Geometric Design 2, 297–306.

Hoschek, J. and Lasser, D. (1993), Fundamentals of Computer Aided Geometric Design, A.K. Peters.

Jensen, T. (1987), Assembling triangular and rectangular patches and multivariate splines, in: Farin, G. (Ed.), Geometric Modeling: Algorithms and New Trends, SIAM , 203–220.

Loop, C. (1994), A G^1 triangular spline surface of arbitrary topological type, Computer Aided Geometric Design 11, 303–330.

Mann, S. (1992), Surface approximation using geometric Hermite patches, Ph.D. Dissertation, University of Washington.

Mann, S., Loop, C., Lounsbery, M., Meyers, D., Painter, J., DeRose, T. and Sloan, K., (1992), A survey of parametric scattered data fitting using triangular interpolants, in: Hagen, H. (Ed.), Curve and Surface Design, SIAM, 145–172.

Neamtu, M. and Pluger, P. (1994), Degenerate polynomial patches of degree 4 and 5 used for geometrically smooth interpolation in $\mathbb{R}^3$, Computer Aided Geometric Design 11, 451–474.

Nielson, G. (1987), A transfinite, visually continuous, triangular interpolant, in: Farin, G. (Ed.), Geometric Modeling: Algorithms and new Trends, SIAM, 235–246.

Peters, J. (1991a), Local cubic and bicubic C^1 surface interpolation with linearly varying boundary normal, Computer Aided Geometric Design 7, 499–516.

Peters, J. (1991b), Smooth interpolation of a mesh of curves, Constructive Approximation 7, 221–246.

Peters, J. (1993), Smooth free-form surfaces over irregular meshes generalizing quadratic splines, Computer Aided Geometric Design 10, 347–361.

Piper, B.R. (1987), Visually smooth interpolation with triangular Bézier patches, in: Farin, G. (Ed.), Geometric Modeling: Algorithms and new Trends, SIAM, 221–233.

Sarraga, R.F. (1987), G^1 interpolation of generally unrestricted cubic Bézier curves, Computer Aided Geometric Design 4, 23–39.

Shirman, L.A. and Séquin, C.H. (1987), Local surface interpolation with Bézier patches, Computer Aided Geometric Design 4, 279–295.

Van Wijk, J.J. (1986), Bicubic patches for approximating non-rectangular control meshes, Computer Aided Geometric Design 3, 1–13.