Transfinite surface interpolation over irregular n-sided domains

Tamás Várady [a], Alyn Rockwood [b], Péter Salvi [a,*]

[a] Budapest University of Technology and Economics, Magyar Tudósok Krt. 2 (IB-412/C), H-1117 Budapest, Hungary
[b] King Abdullah University of Science and Technology, 4700 KAUST, Thuwal 23955-6900, Kingdom of Saudi Arabia

ARTICLE INFO

ABSTRACT

Transfinite surface interpolation is a classic topic of computer-aided geometric design (CAGD), and many non-quadrilateral schemes are known. Surfaces defined solely by means of their boundary curves and cross-tangent functions are needed, for example, in three-dimensional curve network-based design, and to fill complex irregular holes such as in vertex blending. This paper deals with interpolating so-called tangential ribbons. Former schemes are enhanced and extended in order to minimize shape artifacts and to provide a more natural patch interior. The proposed representation is based on irregular convex domains that correspond to the lengths and orientations of the boundary curves. The mapping of the individual ribbons within the n-sided domain is calculated by focused parameterization methods that ensure a balanced orientation related to the center of the domain and avoid parametric shearing. Distance-based blending functions ensure that modifying or inserting a small edge will have only a local effect over the n-sided patch. Constructions to create one-sided or two-sided patches are also presented. Examples and open research topics conclude the paper.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

A fundamental theme in computer-aided geometric design (CAGD) is to create mathematical representations for complex free-form objects, which are composed of several smoothly connected surface patches. While the majority of such patches are four-sided, almost all industrial objects contain general n-sided patches that are inserted into some arrangement of the quadrilaterals. Most frequently three-, five-, and six-sided patches are needed; however one-sided or two-sided patches may also occur in practical design.

There are two important applications of n-sided patches. The surface may be a relatively large functional part, satisfying aesthetic and/or engineering requirements, where shape control and fairness are of primary importance. Alternatively, the surface is a relatively small piece to fill holes or create vertex blends; these are fundamentally defined by boundary constraints, and only natural transition in the interior matters.

Several techniques have been published in the CAGD literature for creating general topology surfaces [1]. We enumerate basic approaches characterizing how boundaries are defined.

(i) *Trim and stitch.* In most CAD/CAM systems four-sided surfaces are created by standard operations (sweeping or lofting, etc.);

then non-four-sided pieces are obtained by removing certain parts using Boolean operations or intersections. The adjacent patches can only be stitched together with approximate G^1 or G^2 continuity, within given tolerances.

(ii) *Quadrilaterals.* Often only four-sided surfaces are available, and n-sided pieces must be subdivided into quadrilaterals. These generally share common internal boundaries that connect the midpoints of the sides with a well-chosen center point (central splitting). Picking optimal subdivision curves and providing internal smoothness are difficult problems. T-splines [2] also belong to this category, and they represent a promising new approach.

(iii) *Polyhedral surfaces.* These surfaces are created by various procedural techniques, where a composite surface is created by (recursively) subdividing a topologically general three-dimensional (3D) control polygon. Applying the well-known recursive subdivision methods or surface splines, a set of smoothly connected quadrilaterals together with n-sided surface elements is created [1]. This approach may face difficulties when explicitly prescribed boundaries and cross-derivative functions need to be interpolated.

(iv) *Genuine n-sided patches.* In this case, we define the patch by a single equation (or procedure). The tensor product schemes will not work for $n \neq 4$ sides in general. There is a wide variety of n-sided patches, where "non-standard" representations are applied; see below. In transfinite surface interpolation we assume that there are well-defined boundaries and cross-derivative functions. The boundaries are parametric curves, but the actual representation

* Corresponding author. Tel.: +36 1 463 2467.
E-mail address: salvi@iit.bme.hu (P. Salvi).

has no significance. We do not want to or cannot provide control points for the interior, and our interest is to create natural transitions *solely* by blending together the given boundaries and tangential constraints.

Practical experience shows that classical schemes may produce unexpected shape artifacts when the boundaries have uneven lengths or are highly curved. This motivated our work to enhance existing representational schemes to provide fair patches even for irregular boundary configurations. After briefly reviewing the *n*-sided surface literature (Section 2), we go back to Coons patches and generalize this approach in three different ways (Section 3). Then various aspects of patch construction are investigated, including domain definition, distance-based blending functions, and different local parameterization schemes (Sections 4 and 5). Illustrative test examples (Section 6) and suggestions for future work conclude the paper.

2. Previous work

Transfinite surface interpolation is a classical area of CAGD. Its origin goes back to the late 1960s, when Coons formulated his Boolean sum surface [3]. This was followed by Gordon's generalization of interpolating a rectangular network of curves [4]. In the next two decades, several new approaches were developed moving first to triangular domains, then later to *n*-sided domains [1]. In spite of the important developments, it seems that there are only two comprehensive reviews on *n*-sided patches [5,6], but these have become somewhat dated. In this section, we refer only to a selection of papers, which were found to be the most pertinent to our current work.

The research on genuine *n*-sided patches started with the pioneering work of Gregory and Charrot [7,8]; this was followed by a sequence of early contributions by Sabin [9,10], Storry and Ball [11], Kato [12], and Várady [13]. Later, Plowman and Charrot tuned Gregory's patch to meet the requirements for setback vertex blending [14]. Methods differ in the way how boundary functions are blended together; for example, Gregory suggested the use of *corner interpolants*, while Kato combined *side interpolants* to obtain the final surface. Another aspect is whether additional *correction surfaces* are needed or not, as in Coons' formulation.

A group of important contributions directly produced control point-based *n*-sided patches that are insertable into rectangular Bézier or B-spline curve networks, including the method of Hosaka and Kimura [15], and the multi-sided generalization of Bézier and B-spline patches by Loop and DeRose [16,17]. The necessity of using two-sided patches was also pointed out in [18].

Concerning the *domain* of *n*-sided patches, the majority of authors use regular *n*-sided domains; Kato [19] proposed a concave projectable domain construction. Serious efforts have been directed to handle internal holes; see [12,20]. The importance of using non-uniform polygonal domains has turned out to be a crucial quality issue for us, and we will return to this topic in Section 4. Note that it is not necessary to have a closed domain in the parameter space: Gao and Rockwood in [21] suggested a scheme based on a special assignment between domain curves and 3D feature curves, which is capable of interpolating these curves in a fairly general manner.

The most important difference amongst the various transfinite approaches is the method of creating the *local parameterization* and blending functions. One option is to use dependent local coordinates, i.e. take two variables and compute the remaining $n - 2$ parameters from them; see, for example, [15,9,10]. Another option is to use the so-called overlap parameterization of $2n$ variables (see [13]), where the parameters are constrained only along the sides. There are several methods in which the local parameters are computed independently by different geometric

procedures, such as dropping perpendicular distances to the polygon sides, radial constructions, and line sweeps. These will be analyzed in detail later; see Section 5.

The degree of continuity between the multi-sided patch and the adjacent patches can be a crucial issue. The majority of solutions ensure only G^1 continuity. Some methods generalize for G^2 in a relatively natural manner [3,12,16,17], and others apply more complex reparameterization techniques; see, for example, [22,23].

A significant amount of research effort has been directed recently to explore generalized barycentric coordinates. The original motivation was to provide an adequate parameterization for data points within convex and concave polygonal domains; see papers on mean value coordinates, including a good list of related references in [24,25]. This idea was later generalized for 3D polyhedra, closed triangular meshes [26], and general polytopes. Main applications include mesh parameterization, interpolating discrete data of vector fields, texture mapping, mesh deformations, amongst several others. The idea of using generalized mean value coordinates to interpolate functions and derivatives has emerged recently in [27,28], where barycentric blending functions are defined by integrals over arbitrary domains.

In the following sections we will deal with various transfinite schemes defined as the convex combination of individual parametric surface interpolants. Our focus is to compute non-regular polygonal domains based on a set of given 3D boundary curves.

We investigate how to enhance the constituents of transfinite representations focusing on special distance-based blending functions and new parameterizations.

3. Coons patches and extensions

First, we revisit classic Coons patches, and then we derive three generalized patch formulations over *n*-sided domains.

3.1. Basic scheme

A G^1 Coons patch is defined by four given boundary curves $S(u, 0)$, $S(u, 1)$, $S(0, v)$, and $S(1, v)$, and four cross-derivative functions $S_v(u, 0), S_v(u, 1), S_u(0, v)$, and $S_u(1, v)$. These parametric functions determine a four-sided interpolating surface $S(u, v)$ with a common parameterization over the $[u, v]$ unit square. The Coons patch is the Boolean sum of two side-to-side surfaces and a four-sided *correction patch* that eliminates the extra boundary terms produced by the side interpolants. Using cubic Hermite blending functions ($\alpha_0(u) = 2u^3 - 3u^2 + 1, \alpha_1(u) = -2u^3 + 3u^2, \beta_0(u) = u^3 - 2u^2 + u, \beta_1(u) = u^3 - u^2$) this can be written as follows:

$$U = [\alpha_0(u) \quad \beta_0(u) \quad \alpha_1(u) \quad \beta_1(u)],$$
$$V = [\alpha_0(v) \quad \beta_0(v) \quad \alpha_1(v) \quad \beta_1(v)],$$
$$S^u = [S(u, 0) \quad S_v(u, 0) \quad S(u, 1) \quad S_v(u, 1)],$$
$$S^v = [S(0, v) \quad S_u(0, v) \quad S(1, v) \quad S_u(1, v)],$$
$$S^{uv} = \begin{bmatrix} S(0,0) & S_u(0,0) & S(1,0) & S_u(1,0) \\ S_v(0,0) & S_{uv}(0,0) & S_v(1,0) & S_{uv}(1,0) \\ S(0,1) & S_u(0,1) & S(1,1) & S_u(1,1) \\ S_v(0,1) & S_{uv}(0,1) & S_v(1,1) & S_{uv}(1,1) \end{bmatrix},$$
$$S(u, v) = V(S^u)^T + S^v U^T - V S^{uv} U^T.$$

The matrices S^u and S^v contain the boundary constraints for the side-to-side interpolants, and matrix S^{uv} contains position, derivative, and mixed derivative (twist) vectors at the four corners. A schematic figure (Fig. 1(a)) shows the logic of Coons patches; black ribbons represent positional and tangent functions to be interpolated, grey ribbons the extra boundary terms to be eliminated. The Boolean sum reproduces the prescribed curves and G^1 constraints along the sides of the domain.
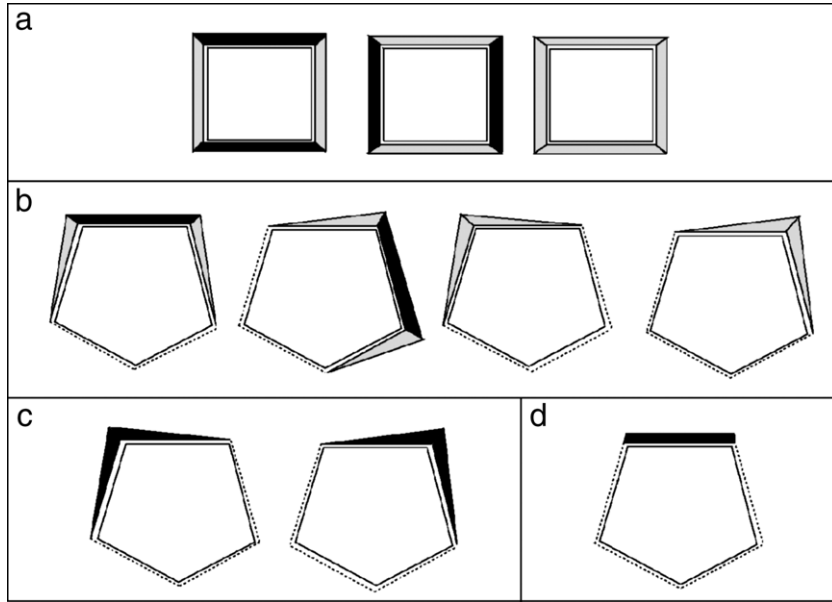
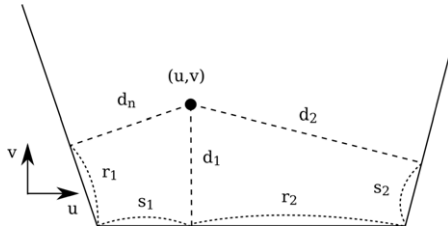**Fig. 1.** Schemes for Coons and n-sided patches.

Fig. 2. Local coordinates.

Coons patch rearranged. For our forthcoming purposes a different side-based indexing will be used. We retain the four-sided domain $[u, v]$ and introduce local *side coordinates* (s_i, r_i), where $s_i = s_i(u, v)$, and $r_i = 1 - s_{i-1}$ parameterizes each side in the reverse direction. Assume a counterclockwise indexing in a circular manner; in the four-sided case, $s_1 = u, r_1 = v, s_2 = v, r_2 = 1 - u$, etc., as shown in Fig. 2.

Accordingly, we use different, side-based notations for the positional and tangential constraints, $P_i(s_i)$ and $T_i(s_i)$, respectively. Then $P_1(s_1) = S(u, 0)$, $T_1(s_1) = S_v(u, 0)$, $P_2(s_2) = S(1, v)$, $T_2(s_2) = -S_u(1, v)$, and so on. We will need to use the mixed partial derivatives $W_i(0)$ at the corners; they can be derived from the cross-derivative functions T_i. (For the time being, we assume that the twist vectors of the adjacent ribbons are compatible; otherwise, Gregory's correction terms [1] need to be applied.)

We collect the constant vector quantities that belong to a corner in order to form separate correction terms. For example, at the second corner, $P_2(0) = P_1(1) = S(1, 0)$, $T_2(0) = -S_u(1, 0)$, $T_2^*(0) = T_1(1) = S_v(1, 0)$, and $W_2(0) = S_{uv}(1, 0)$. The notation $T_i^*(r_i)$ is used instead of $T_{i-1}(s_{i-1})$ to match the corresponding parameter r_i. So we can rewrite the original equation by combining the four side-interpolants and the four correction terms, as follows:

$$S(u, v) = \sum_{i=1}^{4} [\alpha_0(r_i)\ \beta_0(r_i)] \begin{bmatrix} P_i(s_i) \\ T_i(s_i) \end{bmatrix}$$
$$- \sum_{i=1}^{4} [\alpha_0(r_i)\ \beta_0(r_i)] \begin{bmatrix} P_i(0) & T_i^*(0) \\ T_i(0) & W_i(0) \end{bmatrix} \begin{bmatrix} \alpha_0(s_i) \\ \beta_0(s_i) \end{bmatrix}.$$

Ribbon-based extension. It will be useful to concatenate the positional and tangential constraints and introduce G^1 ribbons

defined in the form of

$$R_i(s_i, r_i) = P_i(s_i) + r_i T_i(s_i).$$

Related correction terms are given as

$$Q_i(s_i, r_i) = P_i(0) + r_i T_i(0) + s_i T_i^*(0) + r_i s_i W_i(0).$$

Here, only a single Hermite blending function α_0 associated with each side is needed to formulate a ribbon-based patch:

$$S(u, v) = \sum_{i=1}^{4} R_i(s_i, r_i)\alpha_0(r_i) - \sum_{i=1}^{4} Q_i(s_i, r_i)\alpha_0(r_i)\alpha_0(s_i). \tag{1}$$

This is not identical to the Coons patch, but it can easily be shown that the positional and tangential boundary constraints are satisfied, as expected.

3.2. Blending functions

For n-sided patches, convex polygonal domains in the (u, v) parameter plane will be used. We are going to introduce general blending functions based on a set of *distance parameters* $d_i = d_i(u, v)$, which represent some distance measure from each side of the polygon (Fig. 2). At this point, the only property we use is that $d_i = 0$ on side i and grows in a monotonic way as we move away from this side.

We sum up interpolant surfaces multiplied by blending functions. Let us evaluate the i-th product at an arbitrary point of the i-th boundary as a function of d_i, i.e., $S_i(d_i) = R_i(d_i)\alpha_i(d_i)$. In order to interpolate the positional data, $\alpha_i(0)$ must be equal to 1. In order to interpolate the tangential data, $\frac{\partial S_i}{\partial d_i} = R_i'(d_i)\alpha_i(d_i) + R_i(d_i)\alpha_i'(d_i)$ must yield only the tangential term on the border, i.e., $\alpha_i'(0)$ must be 0. For G^1 continuous cross-derivative constraints, this means that it is sufficient to use quadratic terms in the blending functions. As can easily be shown, this also guarantees that the effect of the $k \neq i$ boundaries and their cross-derivative functions will vanish on the i-th side.

Three different blending functions – side blending λ_i, corner blending κ_i, and a special side blending μ_i – will be investigated. Let $D_{i1,i2,\dots,in}^n$ denote $\prod_{i \neq i1,i2,\dots,in}^{n} d_i^2$.

The *side blending functions* must be equal to 1 on side i, and vanish from 1 to 0 along sides $i - 1$ and $i + 1$. On the remaining

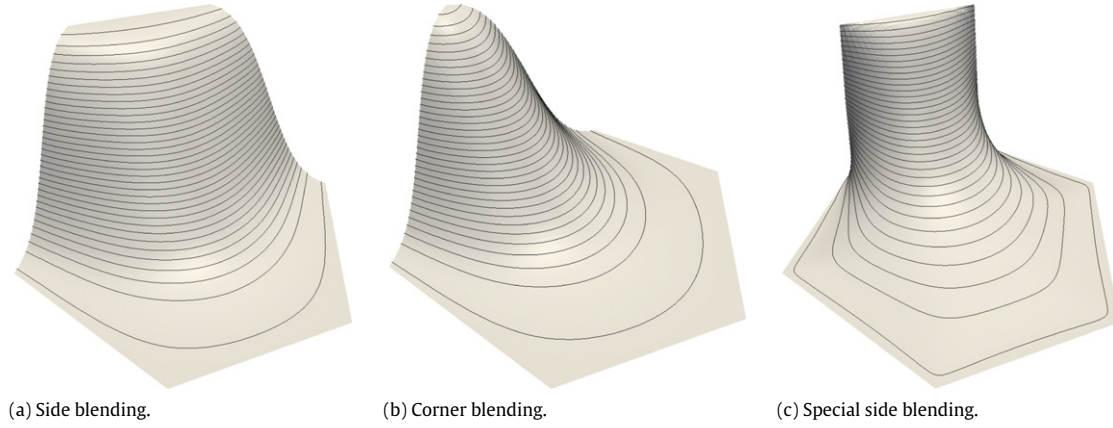(a) Side blending.　　　　　　　　(b) Corner blending.　　　　　　　　(c) Special side blending.

Fig. 3. Blending functions with contours.

$n - 3$ sides ($k \neq i - 1, i, i + 1$), they should vanish; see Fig. 3(a). This can be achieved by

$$\lambda_i(d_1, \ldots, d_n) = \frac{D_{i-1,i}^n + D_{i,i+1}^n}{\sum\limits_{j=1}^{n} D_{j-1,j}^n}.$$

For example, at $n = 4, i = 1$,

$$\lambda_1(d_1, d_2, d_3, d_4) = \frac{d_2^2 d_3^2 + d_3^2 d_4^2}{d_1^2 d_2^2 + d_2^2 d_3^2 + d_3^2 d_4^2 + d_4^2 d_1^2};$$

i.e., if $d_1 = 0$, then $\lambda_1 = 1$; if $d_3 = 0$, then $\lambda_1 = 0$; and if d_4 or d_2 is equal to 0 at $d_1 = d_3$, then $\lambda_1 = 0.5$.

For this set of blending functions $\sum_i^n \lambda_i = 2$, which will be compensated by the corner blending functions (see below).

The *corner blending functions* are equal to 1 at corner i and vanish from 1 to 0 along sides $i - 1$ and i. On the remaining $n - 2$ sides they must be 0 (see Fig. 3(b)):

$$\kappa_i(d_1, \ldots, d_n) = \frac{D_{i-1,i}^n}{\sum\limits_{j=1}^{n} D_{j-1,j}^n}.$$

For example, at $n = 4, i = 1$,

$$\kappa_1(d_1, d_2, d_3, d_4) = \frac{d_2^2 d_3^2}{d_1^2 d_2^2 + d_2^2 d_3^2 + d_3^2 d_4^2 + d_4^2 d_1^2};$$

i.e., if $d_4 = 0$ and $d_1 = 0$, then $\kappa_1 = 1$; if $d_2 = 0$ or $d_3 = 0$, then $\kappa_1 = 0$; and if $d_1 = 0$ and $d_4 = d_2$, then $\kappa_i = 0.5$.

The κ_i blending functions have the partition of unity property. Note that squared terms of generalized barycentric coordinates [25] may also produce corner blending functions with similar properties.

The *special side blending functions* are equal to 1 along side i, and 0 for all the remaining $n - 1$ sides where $k \neq i$:

$$\mu_i(d_1, \ldots, d_n) = \frac{D_i^n}{\sum\limits_{j=1}^{n} D_j^n}.$$

This blending function is singular at the corner points; for example, there is a jump between $\mu_1(0, d_2, \ldots, d_{n-1}, \varepsilon) = 1$ and $\mu_1(\varepsilon, d_2, \ldots, d_{n-1}, 0) = 0$. This singularity vanishes when two adjacent blending functions are added at a given corner:

$$\lim_{\substack{d_{i-1} \to 0, \\ d_i \to 0}} \mu_{i-1}(d_1, d_2, \ldots, d_n) + \mu_i(d_1, d_2, \ldots, d_n) = 1.$$

Thus, for all domain points the μ_i blending functions also have the partition of unity property. The blending functions are depicted in Fig. 3(c). For example, at $n = 4, i = 1$,

$$\mu_1(d_1, d_2, d_3, d_4) = \frac{d_2^2 d_3^2 d_4^2}{d_1^2 d_2^2 d_3^2 + d_2^2 d_3^2 d_4^2 + d_3^2 d_4^2 d_1^2 + d_4^2 d_1^2 d_2^2},$$

i.e., if $d_1 = 0$, then $\mu_1 = 1$; if $d_2 = 0$ or $d_3 = 0$ or $d_4 = 0$, then $\mu_1 = 0$.

Keep in mind that all side and distance parameters (s_i, d_i) depend on the domain parameters (u, v), and the actual properties of the patch will be determined by the blending functions and the local parameterization of the ribbons.

3.3. Three approaches

Direct generalization of Coons patches. The genuine generalization of Coons' method is to take formula (1), run the index from 1 to n, and apply the side and corner blending functions to merge the ribbon and correction patches; i.e.,

$$S(u, v) = \sum_{i=1}^{n} R_i(s_i, d_i) \lambda_i(d_1, \ldots, d_n)$$

$$- \sum_{i=1}^{n} Q_i(s_i, r_i) \kappa_i(d_1, \ldots, d_n).$$

To our best knowledge, this direct formula has not been proposed earlier, possibly due to the difficulties of finding appropriate common parameterizations. The scheme is depicted as in Fig. 1(b): the ribbons interpolate the i-th side and their effect gradually vanishes along the $(i-1)$-th and $(i+1)$-th sides due to the blending functions. The superfluous boundary data are eliminated by the i-th and the $(i + 1)$-th correction patches, yielding a correct G^1 interpolation.

Combining corner interpolants. Gregory et al. [7,14] suggested applying local corner interpolants, each created by two adjacent boundaries and cross-derivative functions. It is possible to incorporate the previous correction terms into the corner interpolant, and then only corner-type blending functions need to be used:

$$C_i(s_i, r_i) = P_{i-1}^*(r_i) + P_i(s_i) + s_i T_{i-1}^*(r_i) + r_i T_i(s_i) - Q_i(s_i, r_i)$$

$$S(u, v) = \sum_{i=1}^{n} C_i(s_i, r_i) \kappa_i(d_1, \ldots, d_n).$$

This scheme is shown in Fig. 1(c). The weighted corner interpolants gradually vanish as they reach the adjacent corners, and
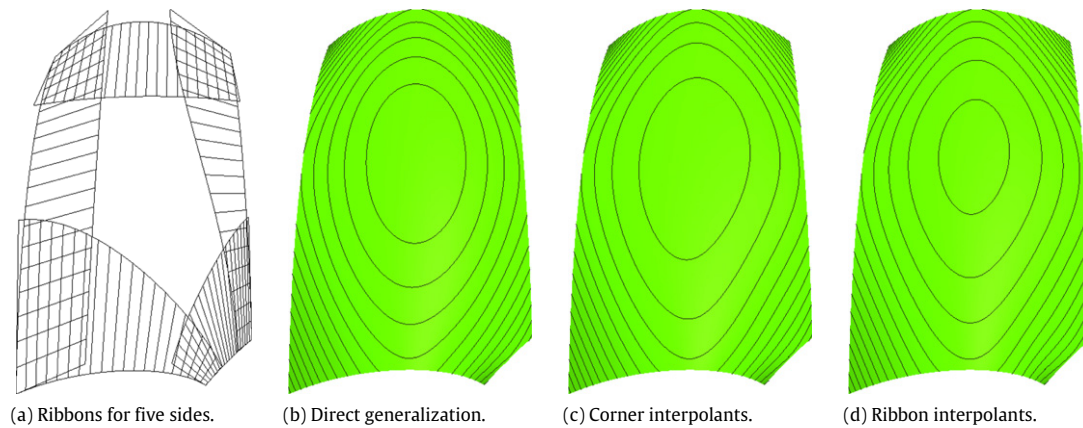
(a) Ribbons for five sides.　　(b) Direct generalization.　　(c) Corner interpolants.　　(d) Ribbon interpolants.

Fig. 4. Three basic approaches.

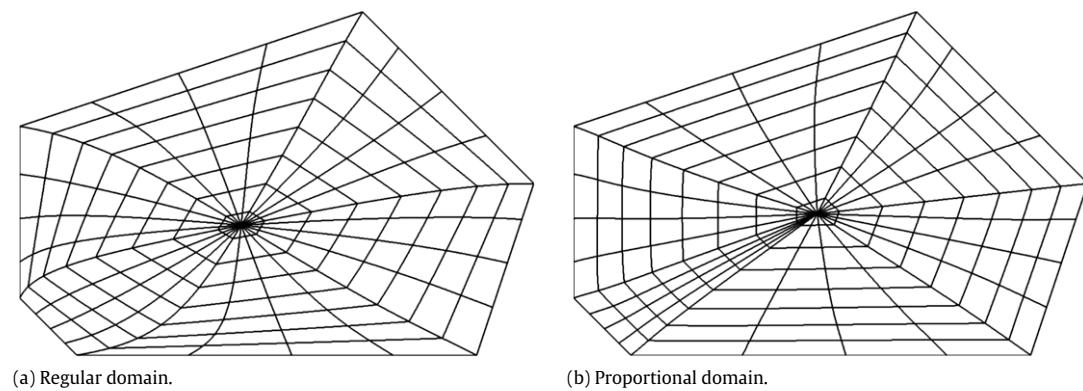(a) Regular domain.　　　　　　　　　　　　(b) Proportional domain.

Fig. 5. Six-sided patch with "spider" lines.

the blending functions and the corresponding parameterization ensure that the boundary constraints are reproduced along the sides.

Combining ribbon interpolants. It is also possible to create ribbon-based patches *without* correction terms using a special type of side blending that avoids producing extra boundary data to be eliminated. A similar solution was suggested by Kato [12], amongst others. In this case, only the ribbons are weighted, and a special convex combination is applied:

$$S(u, v) = \sum_{i=1}^{n} R_i(s_i, d_i)\mu_i(d_1, \ldots, d_n).$$

This scheme is depicted as in Fig. 1(d).

Fig. 4 shows three transfinite patches using the same ribbon input. As the contour lines show, it is not easy to distinguish by shape, and different applications may set different priorities to select the most favorable. In the following sections, we will focus on other aspects, in particular domain and parameterization, which have significant influence in creating nice *n*-sided patches.

4. Domain polygons

Our goal is to determine an appropriate *non-regular* convex domain, based on the given loop of 3D boundary curves. The use of non-regular polygons is needed for quality purposes, since domain parameterization should "mimic" the shape of the *n*-sided patch. It is similar to using non-uniform parameters for B-spline curves and
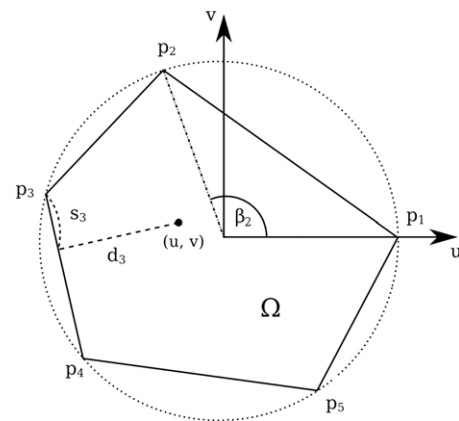

Fig. 6. Circular polygonal domain I.

surfaces. An "evenly" located set of constant parameter lines in the domain should be mapped roughly into an "evenly" distributed set of curves on the 3D surface.

We have found that, when boundaries of different lengths are used with a regular domain polygon, a strong distortion of the parameterization may occur, leading to undesirable shape artifacts. In Fig. 5, we show spider-net curves, i.e. constant parameter lines parallel to the domain sides; observe the
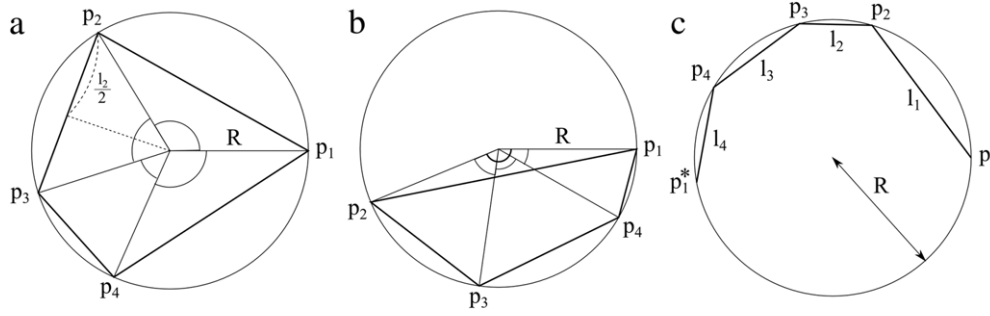
Fig. 7. Circular polygonal domain II.

difference between the parameterization of the regular and non-regular domains. Another example comparing mean curvature maps will be shown later in Fig. 17.

Let Ω denote the convex domain in the (u, v) plane, and Γ its boundary. $p_i = (u_i, v_i)$, $i = 1, \ldots, n$ are the vertices of the polygon to be determined. Index i runs in a counterclockwise order (Fig. 6). Denote the arc lengths of the given 3D boundary curves by L_i, and the angles between the end tangents of the $(i-1)$-th and i-th boundaries by ϕ_i. We would like to compute a convex domain "similar" to the 3D configuration. Denoting the sides and the angles of the domain by l_i and α_i, respectively, we seek to minimize the squared deviation of the chord lengths and the angles, i.e., $\sum (l_i - c_{\text{length}} L_i)^2 + \sum (\alpha_i - c_{\text{angle}} \phi_i)^2$, where c_{length} and c_{angle} are properly chosen constants. This is a non-linear problem, but simple heuristic methods are proposed that work well in practice.

(i) The simplest is to place domain vertices proportionally by arc length on the perimeter of a unit circle (inscribed polygon). Place the first vertex on the u axis, and the subsequent ones ($i = 2, \ldots, n-1$) at angle

$$\beta_i = 2\pi \cdot \frac{\sum\limits_{k=1}^{i-1} L_k}{\sum\limits_{k=1}^{n} L_k}$$

(see Fig. 6).

(ii) For a better solution, instead of central angles, make the sides of the polygon proportional to the arc lengths of the boundaries. As Fig. 7 shows, two cases need to be distinguished: in case A, the center point of the circle is contained in the convex hull (Fig. 7(a)), while, in case B, it lies outside (Fig. 7(b)).

Now, take a sufficiently large circle with radius R, and place the chords L_i onto the circle one by one, placing an endpoint of the largest side (denoted here by L_1) onto the u axis (Fig. 7(c)). Then start decreasing the radius, and let the chord endpoints slide towards the other end, i.e., p_1^* moves towards p_1. In case A, this will be successful when the central half-angles of the chords satisfy $\sum \arccos \frac{L_i}{2R} = \pi$, assuming $L_1 < 2R$; otherwise, at one instant of the circle shrinking process, $2R$ becomes equal to L_1 without closing the loop. Then radius R needs to start growing again (case B), until we find an appropriate configuration where $\sum\limits_{2}^{n} \arccos \frac{L_i}{2R}$ is equal to $\arccos \frac{L_1}{2R}$. Note that, based on the given construction, for the existence of such a domain polygon it is sufficient to have $L_1 < \sum\limits_{2}^{n} L_i$.

(iii) The third simple heuristic takes into consideration not only the lengths of the boundaries, but the local 3D angles, ϕ_i, as well. First, we normalize the angles to satisfy the necessary angle criterion for the n-sided convex polygon; i.e., let $c_{\text{angle}} = (n-2)\pi / \sum \phi_i$; then $\alpha_i = c_{\text{angle}} \phi_i$. Now, take the polygon sides in sequence, retaining the angles, which will likely yield an open polyline,
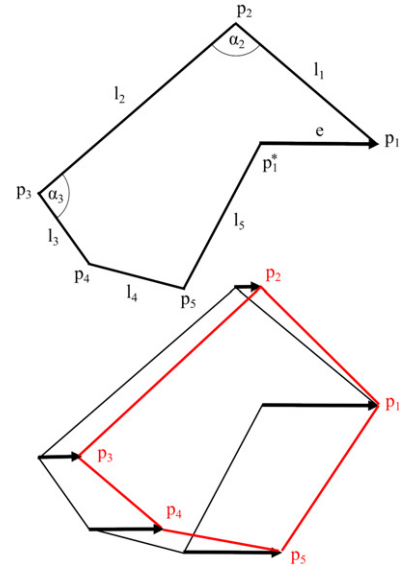

Fig. 8. Length-/angle-based convex polygonal domain.

having a difference vector $\underline{e}$ between the first and the last points. In order to improve this, we fix the very first point, and modify the subsequent ones sequentially, first by $\frac{1}{n}\underline{e}$, then by $\frac{i}{n}\underline{e}$ (Fig. 8). In the closed polygon obtained, both the chord lengths and the angles are somewhat distorted, but the result proved to be satisfactory in our experiments.

5. Local parameterization schemes

The essence of all transfinite schemes is parameterization. Having a given point in the domain, (i) we determine n corresponding data points on the individual interpolants and (ii) combine these by corresponding blending functions. For example, the ribbon mapping $(u, v) \rightarrow (s_i, d_i)$ produces local ribbon coordinates to be substituted into $R_i(s_i, d_i)$; and $(u, v) \rightarrow d_i$ produces n distance values to compute the weights of the blending functions $\mu_i = \mu_i(d_1, \ldots, d_n)$. These mappings are fundamental in defining the shape of the patch and the differential properties along the sides.

In this section, the subject of our investigation is how to calculate these local coordinates, and, as will be shown, there is a wide variety of algorithms to do this. Simple methods of calculating distance parameters include *perpendicular projection*, where we take a (u, v) pair, and drop a perpendicular line to each side. Alternatively, *side-based barycentric coordinates* can be defined that
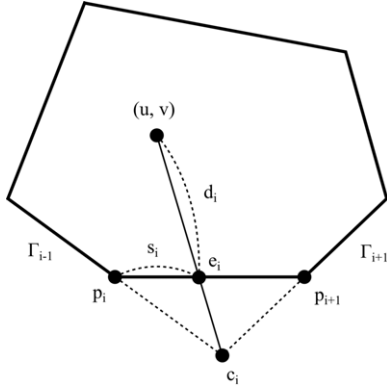
Fig. 9. Computing the radial parameterization.

Fig. 10. Computing the central line sweep parameterization.

divide the area of the triangle $[(u, v), p_i, p_{i+1}]$ by that of the whole polygon; then this value is normalized by a constant $n/2$ to obtain d_i. Another measure – *chord-based coordinates* – was suggested by Kato [19]. Its main advantage is that it can be used for domains with internal hole loops and concave corners as well. Take the chords that connect the domain point with the corners of the i-th side; then

$$d_i = |(u, v) - p_i| + |(u, v) - p_{i+1}| - |p_{i+1} - p_i|.$$

Distance parameters may be inadequate to determine the related side parameters. For example, in the perpendicular method, the footpoint may fall beyond the actual side, so it is not a good way to define s_i. Kato proposed $s_i = d_{i-1}/(d_{i-1} + d_{i+1})$, which guarantees that s_i will be in the $[0, 1]$ interval. However, this will not yield a linear mapping for s_i. For example, the parametric midpoint is not necessarily identical to the midpoint of the chord, in which case further reparameterization is needed.

The (s_i, d_i) parameters can be determined by so-called line-sweep constructions as well, as described below.

Radial distance functions were suggested by Charrot and Gregory [7], and these also work for non-regular domains. As shown in Fig. 9, the extended sides Γ_{i-1} and Γ_{i+1} intersect at point c_i. A line sweep connecting c_i and (u, v) intersects side i at point e_i; then $d_i = |(u, v) - e_i|$ and $s_i = |e_i - p_i|$.

We propose another method, called *central line sweep*, to create sweeping lines that run from the left edge Γ_{i-1} to the right edge Γ_{i+1} in a different way than the radial projector. As shown later, this helps to avoid skewed parameterizations by forcing the middle line of a ribbon to be mapped onto a line that connects the midpoint of side i and the center point of the domain $c = (c^u, c^v)$; see Fig. 10. c might be calculated as the average of the corners, but practice shows that for non-regular domains a weighted average of chord lengths works better:

$$c = \frac{0.5 \sum_i p_i (l_{i-1} + l_i)}{\sum_i p_i l_i}.$$

Our goal is to find a parameterizing function $r(s, d)$ for which the $s = 0.5$ constant parameter line contains the center point of the polygon, i.e., for some unknown d_c parameter value $r(0.5, d_c) = c$. We deal with the local parameters of side 1. For simplicity's sake, let us position corner p_1 at the origin, and place p_2 on the u-axis. A linear by quadratic map is introduced:

$$r(s, d) = p_2 s + [w_1 (1-s)^2 + 2w_{12}(1-s)s + w_2 s^2]d, \qquad (2)$$

where vectors w_1, w_{12}, w_2 define the direction of the sweep. Not only is the parameter value (d_c) unknown, but the vector $w_{12} =$
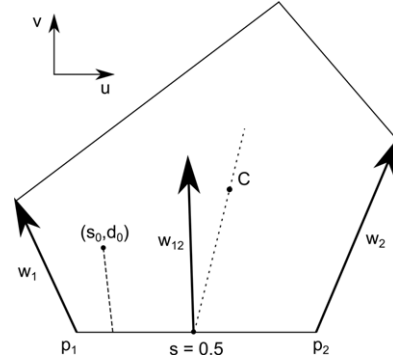
(w^u_{12}, w^v_{12}), as well. To simplify our calculation, we require that $w^v_{12} = 0.5(w^v_1 + w^v_2)$. On the halving line s is 0.5, so at the center

$$c^v = 0.25[w^v_1 + 2w^v_{12} + w^v_2]d_c,$$

and thus $d_c = 2c^v/(w^v_1 + w^v_2)$. From the other coordinate equation,

$$c^u = p^u_2 0.5 + 0.25[w^u_1 + 2w^u_{12} + w^u_2]d_c,$$

so we can express the missing u component of w_{12}. Having the three direction vectors defined, we can determine (s_0, d_0) for any domain point (u_0, v_0). Express d_0 from Eq. (2); then, after solving a quadratic equation for s_0, we obtain the requested local coordinates:

$$\begin{aligned}
d_0 &= \frac{u_0 - p^u_2 s_0}{w^u_1 (1-s_0)^2 + 2w^u_{12}(1-s_0)s_0 + w^u_2 s^2_0} \\
&= \frac{v_0}{w^v_1 (1-s_0) + w^v_2 s}.
\end{aligned}$$

If needed, one can reparameterize $(s, d) \rightarrow (s, t)$ to force the center point of the ribbon on its middle line to correspond to the center of the domain. In this case, $t = [(1-s)^2 + 2\frac{1-d_c}{d_c}(1-s)s + s^2]d$ ensures that $r(0.5, 0.5) = c$.

Both the radial and the central line sweep parameterizations map the linear s-constant parameter lines of a ribbon into straight lines in the domain space; however, this is not necessary. Instead of a line sweep it is also possible to create a doubly curved, *biquadratic mapping*, as suggested by Várady [13]. In this case, the (s, d) coordinates are determined by biquadratic Bézier control points, naturally placed on the sides of the domain; see Fig. 11. While this mapping is linear on the ith-side, it is quadratic for a general (s, d) within the domain:

$$(u, v) = \sum_{i=0}^{2} \sum_{j=0}^{2} C_{ij} B^2_i(s) B^2_i(d).$$

To compute (s, d), an inverse mapping from (u, v) is performed by a few Newton–Raphson iterations.

Space limits us from going into the details of how to parameterize interpolants for the direct generalization approach (see Section 3). On each boundary, three vector terms – one from the ribbon interpolant and two from the correction patches – are added together. In order to reproduce the cross-derivatives of the ribbon, the parameterization must satisfy certain differential properties. As shown in Fig. 11(a), the cross-direction parameterization on the i-th side is determined by the local control point pairs 00–10, 01–11, 02–12. On the $i + 1$-th side the parameterization will be defined by pairs 02–01, 12–11, 22–21. These are identical to the control point pairs of the subsequent $(i, i+1)$ corner parameterization, denoted by 00–10, 01–11, 02–12 in Fig. 11(b), which compensate
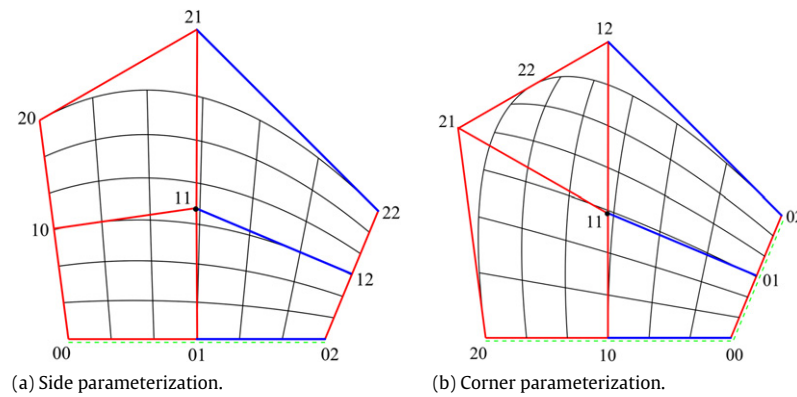
(a) Side parameterization.   (b) Corner parameterization.

Fig. 11. Biquadratic parameterization.

the extra term coming from the i-th side interpolant along the $i+1$-th side. In this way, the biquadratic scheme automatically satisfies the requested differential properties for the direct generalization approach.

6. Discussion and examples

In the previous sections, we have discussed the basic elements for constructing n-sided transfinite surfaces. We continue analyzing important quality issues that are needed for practical applications.

(i) It is an obvious and necessary condition that the boundaries and the cross-derivatives must be "fair". We desire compatible twist vectors for the cross-derivatives; however, this is not required for the "combining ribbons" scheme, as the rational blending functions will average incompatible twists.

(ii) We have already expressed our view concerning the importance of non-regular domains; see Fig. 5, shown earlier.

(iii) The next issue is investigating the influence of the individual boundaries for the overall shape, which is strongly related to the distance measures applied. Loosely speaking, one can compare the strength of the individual ribbons near the boundaries with the interior where the convex combination dominates. If the ribbons are "too wide", then the surface will adhere to them, and there may be sudden curvature changes in the interior. In the "too narrow" case, the influence of the ribbons is weak, and sudden curvature changes may occur close to the boundaries. Finding a good compromise is a delicate challenge, and it is not straightforward to pick the best method.

Fig. 12 shows color maps that correspond to the influence of the individual blending functions on the domain. Each border curve is colored differently, and the vanishing effect of the blends is depicted. Black lines limit the areas where the weight of the given interpolants is more than 90%. White lines indicate locations where two distances taken from two different sides have the same value. There are vertices inside the domain where more than two distances are identical.

The first example in Fig. 12 shows roughly equal distances from the sides of the domain, which means that the midpoint of the surface will be a combination of the midpoints of the ribbons by weight $\frac{1}{n}$. In other words, the ribbons, even the small green one, are almost uniformly pulled towards the center point of the surface. The second example shows a proportional distance computation, where shorter edges have smaller effect towards the interior. In this figure, perpendicular distances were used to obtain this Voronoi-like structure of white edges. The effect of the small green edge quickly vanishes, and the three longer sides on the left side blend together and entirely dominate the small one. The left part
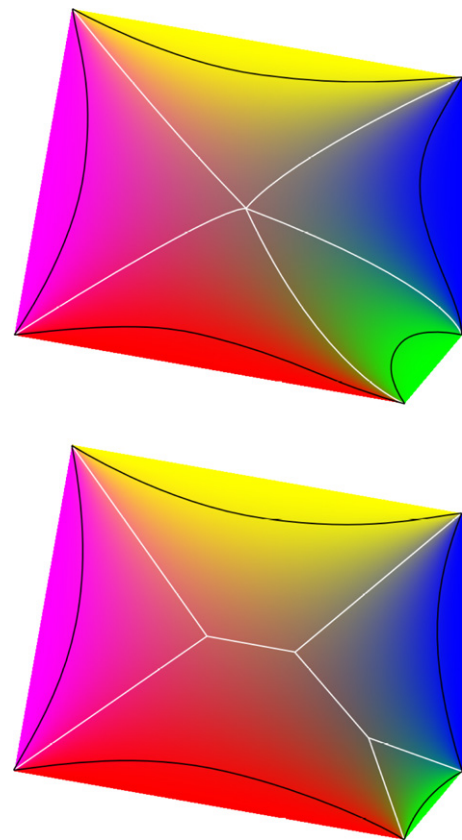

Fig. 12. Blending function distributions.

of the surface acts independently, regardless of whether the green edge is inserted or removed. Clearly, this behavior is desirable in the majority of practical applications.

(iv) The individual ribbons should map onto the domain in a well-oriented manner; for example, we want the middle parameter lines of each ribbon to lie somewhere around an approximate halfway line in the domain. Unfortunately, this is not always the case, and shearing effects can be observed when the domain polygon has large angles at the corners. Consider the radial parameterization in Fig. 13. The first set of constant parameter lines (s_i, d_i) relates to the top right side, and the second set (s_{i-1}, s_i) to the corner between the right and top right sides. Finally, the third picture set shows all the halfway lines of the individual ribbons $s_i = 0.5$ mapped to the domain. Observe that the halfway lines are awkwardly distributed, and accordingly the halfway lines
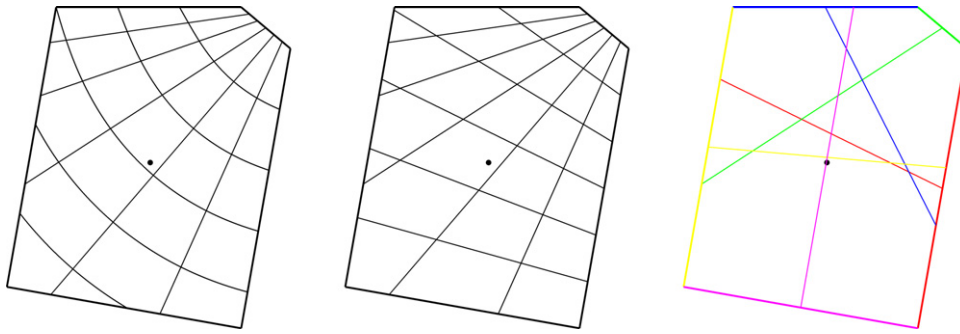
Fig. 13. Radial distance function: (a) (s_3, d_3) and (b) (s_2, s_3) parameter lines, (c) halving lines.
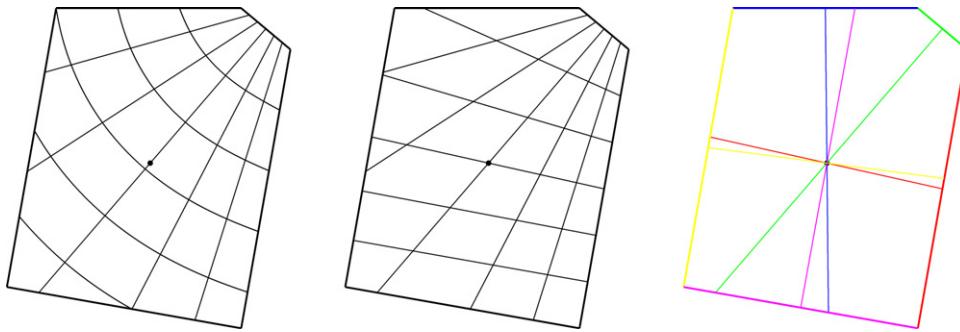

Fig. 14. Central line sweep: (a) (s_3, d_3) and (b) (s_2, s_3) parameter lines, (c) halving lines.

of the ribbons are mapped in a distorted manner. Note that the blue halfway line coming from the top side is strongly skewed to the right side. Ideally it should be located somewhere in the center.

This deficiency is rectified by the central line sweep method. Compare the constant parameter lines in Fig. 14 with the previous subfigures. By construction, the halfway lines go through the center point, and, as a result, the ribbons will be mapped in a more balanced manner. This will improve the shape of the surface and minimize shape artifacts in skewed configurations.

(v) Without going into details, we note that the line sweep technique can be applied to generate two-sided and one-sided patches, as well; see Figs. 15 and 16. The domain of the two-sided patch is bounded by two parabolic arcs, and the distance parameters are computed using simple line sweeps by a quadratic function. The ribbons are combined by simple blending functions of type

$$K_i(d_1, d_2) = \frac{d_j^2}{d_1^2 + d_2^2}, \quad i, j \in \{1, 2\}, \ i \neq j.$$

The one-sided patch is also a combination of two entities. Take a closed curve and an associated ribbon without local self-intersection, and an auxiliary point with a normal vector to define an annular ribbon in the middle of the patch. Apply the above blending functions to obtain the surface. One-sided patches are not only of theoretical interest, but there are practical cases where protrusions or depressions controlled by ribbons need to be created.

To conclude our discussion, it would be hard to pick a single best method from the three proposed transfinite schemes, i.e., generalized Coons patches, corner interpolant-based patches and side interpolant-based patches. In the majority of ordinary cases, it is hard to visually distinguish between them (see Fig. 4), and shape artifacts can only be observed when the boundary segments have uneven lengths and are highly curved.
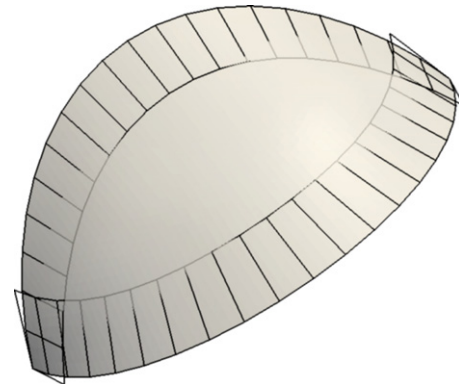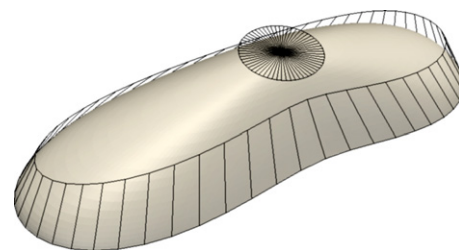

Fig. 15. Two-sided patch.

Fig. 16. One-sided patch.

This may disappoint some of our readers, but there are extraordinary configurations where corner-based patches produce better curvature distribution, while in other cases side-based patches are better. Generalized Coons patches – in some sense – combine the previous two approaches, and thus they merge the good and bad shape features. At the same time, their
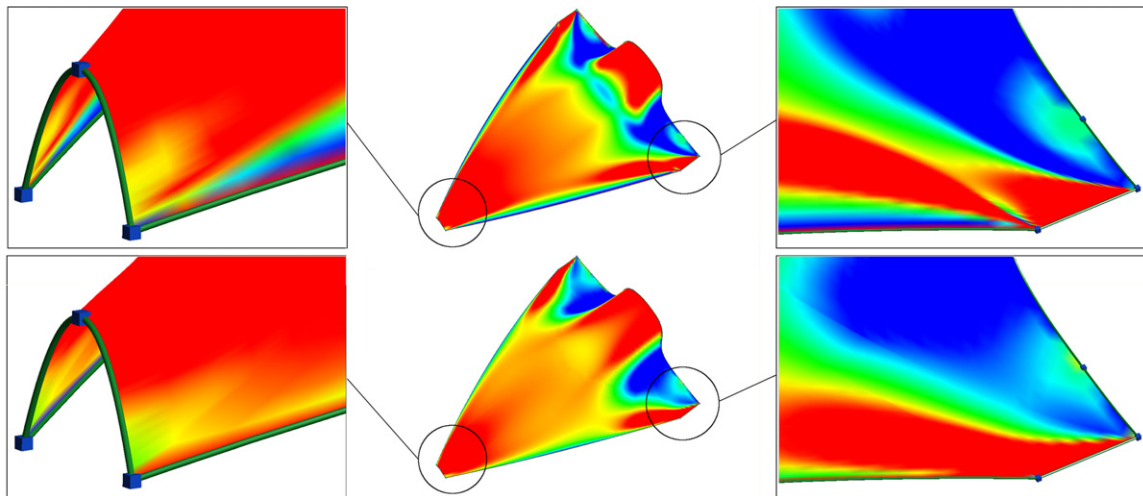
Fig. 17. Mean curvature map of a model using regular (top) and non-regular (bottom) domain polygons.

parameterization is more demanding, and the evaluation of the patch takes somewhat longer.

Nevertheless, our experience shows that by using non-regular domains and the central line sweep parameterization it is possible to significantly improve the surface quality and avoid shape artifacts in extraordinary cases. A six-sided test example to compare regular and non-regular domains is shown in Fig. 17; the mean curvature map shows the differences in the interior and at the corners.

Another simple test object is shown in Fig. 18 using central line sweeps. The given 3D curve network defines two three-sided patches, one five-sided patch, and one six-sided patch. As can be seen, the configuration includes smooth and sharp edges. The boundary ribbons are automatically generated based on the curve network and the patches by means of these ribbons. For smooth edges, the ribbons are G^1 compatible; thus the adjacent patches are also G^1. The ribbons have an extra degree of freedom in magnitude, which can be useful for users to adjust the fullness of these patches.

7. Conclusion and future work

Different schemes to create n-sided transfinite surfaces have been investigated. After generalizing Coons' classical formulation, various extensions to enhance n-sided patch constructions were derived by which more natural shapes emerge, especially when the boundaries of the patch have uneven lengths, or are strongly curved.

There are many opportunities for future research. Introducing non-convex domains or domains with curved boundaries can reduce the number of artificial subdivisions and enhance the surface quality. Providing additional shape control for the interior of the patches is a challenging issue. The automatic generation of natural and compatible cross-derivative functions based on a general free-form curve network with G^1 and G^2 continuity would be a disruptive technological advance. Finally, it is an important open area to develop fairing techniques for ribbon-based transfinite surfaces.
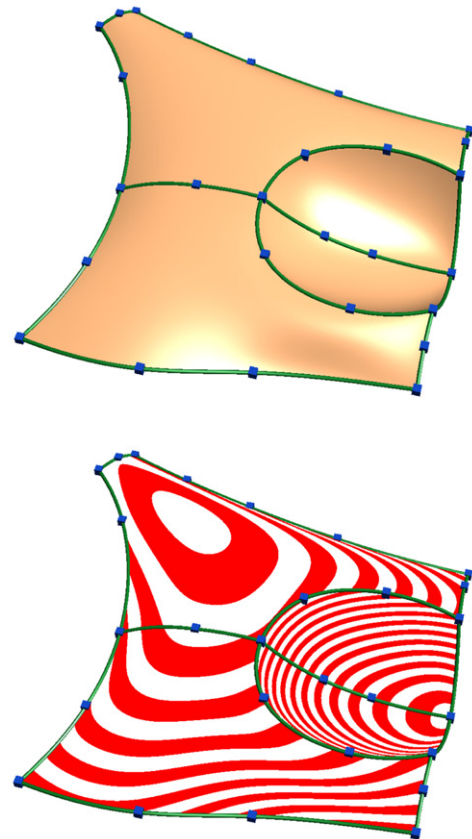
Fig. 18. Simple test object with smooth and sharp edges, using shading (top) and contouring (bottom).

References

[1] Farin G. Curves and surfaces for CAGD: a practical guide. 5th ed. San Francisco (CA, USA): Morgan Kaufmann Publishers Inc.; 2002.
[2] Sederberg TW, Zheng J, Bakenov A, Nasri A. T-splines and T-NURCCs, SIGGRAPH'03. New York (NY, USA): ACM; 2003. p. 477–84.

[3] Coons SA. Surfaces for computer-aided design of space forms. Tech. rep. Cambridge (MA, USA): Massachusetts Institute of Technology; 1967.

[4] Gordon WJ. Spline blended surface interpolation through curve networks. Journal of Mathematics and Mechanics 1969;18(10):931–52.

[5] Varady T. Survey and new results in n-sided patch generation. In: Proceedings on mathematics of surfaces II. New York (NY, USA): Clarendon Press; 1988. p. 203–35.

[6] Malraison P. A bibliography for n-sided surfaces. In: Mathematics of surfaces (VIII), information geometers. Winchester (UK); 1998. p. 419–30.

[7] Charrot P, Gregory JA. A pentagonal surface patch for computer aided geometric design. Computer Aided Geometric Design 1984;1(1):87–94.

[8] Gregory JA. N-sided surface patches. In: The mathematics of surfaces. USA: Oxford University Press; 1986. p. 217–32.

[9] Sabin M. Non-rectangular surfaces patches suitable for inclusion in a B-spline surface. In: Eurographics'83. North Holland; 1983. p. 57–70.

[10] Sabin M. Some negative results in n sided patches. Computer-Aided Design 1986;18(1):38–44.

[11] Storry D, Ball A. Design of an n-sided surface patch from Hermite boundary data. Computer Aided Geometric Design 1989;6(2):111–20.

[12] Kato K. Generation of n-sided surface patches with holes. Computer-Aided Design 1991;23(10):676–83.

[13] Varady T. Overlap patches: a new scheme for interpolating curve networks with n-sided regions. Computer Aided Geometric Design 1991;8(1):7–27.

[14] Plowman D, Charrot P. A practical implementation of vertex blend surfaces using an n-sided patch. In: Proceedings of the 6th IMA conference on the mathematics of surfaces. New York (NY, USA): Clarendon Press; 1996. p. 67–78.

[15] Hosaka M, Kimura F. Non-four-sided patch expressions with control points. Computer Aided Geometric Design 1984;1(1):75–86.

[16] Loop CT, DeRose TD. A multisided generalization of Bézier surfaces. ACM Transactions on Graphics 1989;8:204–34.

[17] Loop CT, DeRose TD. Generalized B-spline surfaces of arbitrary topology. In: Proceedings of the 17th annual conference on computer graphics and interactive techniques, SIGGRAPH'90. New York (NY, USA): ACM; 1990. p. 347–56.

[18] Sabin M. Transfinite surface interpolation. In: Proceedings of the 6th IMA conference on the mathematics of surfaces. New York (NY, USA): Clarendon Press; 1996. p. 517–34.

[19] Kato K. n-sided surface generation from arbitrary boundary edges. In: Laurent P-J, Sablonnière P, Schumaker LL, editors. Curve and surface design: Saint-Malo 1999, innovations in applied mathematics. Nashville (TN): Vanderbilt University Press; 2000. p. 173–81.

[20] Sabin M. Further transfinite surface developments. In: Mathematics of surfaces (VIII), information geometers. Winchester (UK); 1998. p. 161–73.

[21] Gao K, Rockwood A. Multi-sided attribute based modeling. In: Martin R, Bez H, Sabin M, editors. Mathematics of surfaces XI. Lecture notes in computer science, vol. 3604. Berlin (Heidelberg): Springer; 2005. p. 219–32.

[22] Gregory JA, Hahn JM. A C^2 polygonal surface patch. Computer Aided Geometric Design 1989;6:69–75.

[23] Gregory JA, Lau VKH, Zhou J. Smooth parametric surfaces and n-sided patches. In: Dahmen W, Gasca M, Micchelli CA, editors. Computation of curves and surfaces. Kluwer Academic Publishers; 1990. p. 499–528.

[24] Floater MS. Mean value coordinates. Computer Aided Geometric Design 2003; 20(1):19–27.

[25] Hormann K, Floater MS. Mean value coordinates for arbitrary planar polygons. ACM Transactions on Graphics 2006;25:1424–41.

[26] Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. ACM Transactions on Graphics 2005;24:561–6.

[27] Floater MS, Schulz C. Pointwise radial minimization: Hermite interpolation on arbitrary domains. In: Proceedings of the symposium on geometry processing, SGP'08, Eurographics association. Switzerland; 2008. p. 1505–12.

[28] Manson J, Schaefer S. Moving least squares coordinates. Computer Graphics Forum 2010;29(5):1517–24.