Transfinite Surface Interpolation with Interior Control

Tamás Várady[†], Péter Salvi[†], Alyn Rockwood[‡]
[†] Budapest University of Technology and Economics
[‡] King Abdullah University of Science and Technology

Abstract

There are various techniques to design complex free-form shapes with general topology. In contrast to the approaches based on trimmed surfaces and control polyhedra, in curve network-based design feature curves can be directly created and edited in 3D. Multi-sided patches interpolate this curve network with slopes given by associated tangent ribbons. The patches are smoothly connected and yield a natural and predictable surface model. This paper focuses on special design techniques to adjust the interior of transfinite patches when further shape control is needed. While the boundary constraints are retained, additional vertices, curves and even interior control surfaces are supplemented to gain more design freedom. The main idea is to apply different distance-based blending functions with special parameterizations over non-regular, n-sided domains. This concept can be naturally extended to create one- and two-sided patches as well. Shape variations will be demonstrated by a few simple examples.

Key words: Curve network-based design, transfinite surface interpolation, blending functions, interior shape control.

1. Introduction

Creating complex free-form objects, composed of smoothly connected surface patches is a fundamental goal in Computer Aided Geometric Design. While the majority of such patches are four-sided, almost all industrial objects contain general n-sided regions that need to be inserted into some arrangement of quadrilaterals.

The standard technique to create general topological surfaces is based on trimmed quadrilaterals. First a sequence of surface intersections is performed, followed by operations to stitch together trimmed components and/or insert connecting surfaces, such as fillets. Typically, only tolerance-based G^1 or G^2 continuity can be achieved. One particular difficulty is that curves are not "uniform", so trimming curves and the associated cross-tangent constraints cannot be directly manipulated as with the original boundaries of the quadrilaterals.

An alternative approach is driven by general topology control polyhedra using recursive subdivision or surface splines [3]; these yield a set of smoothly connected quadrilaterals combined with n-sided surface patches. Difficulties of this approach include the "ab initio" creation of good control polyhedra and the lack of straightforward operations to directly interpolate and edit prescribed free-form curves and tangential constraints.

Curve network-based design is a third approach, where a collection of feature curves is created and each curve is directly editable. The input is often a collection of manually drawn sketch curves. The curve network defines a set of so-called ribbons, which are smoothly interpolated by multi-sided patches (transfinite interpolation). The shape of these patches can be locally adjusted. This property is an advantage and a disadvantage at the same time: global fairing of curve networks is possible, but methods for global fairing of general topology surface models are not known.

In order to focus on shape concepts and aesthetic requirements, a high level of design flexibility is needed. While the interpolating patches are supposed to yield a natural blend between the boundaries, their interior is not uniquely defined. In this paper new techniques are proposed that extend the capabilities of conventional transfinite interpolation. These provide additional control for the interior, and offer further degrees of freedom to perfect the shapes while the initial boundary constraints are retained.

Transfinite surface interpolation is a classical area of CAGD. Its origin goes back to the late 60's, when Coons formulated his Boolean sum surface [2]. This was followed by Gordon's generalization to interpolate a rectangular network of curves [4]. In the next two decades, several papers were published, first on triangular patches (see summary in [3]), and later on genuine n-sided patches, including the pioneering work of Gregory and Charrot [1,5], Sabin [13,14], Kato [8], Várady [17], and Plowman and Charrot

[12]. The alternatives of creating n-sided transfinite patches with different blending functions and different parameterizations have been recently published in a companion paper by the current authors [19]. As discussed, transfinite patches may blend together *corner interpolants* or *side interpolants*, however, there exists a third approach, that generalizes Coons' original Boolean sum concept using side interpolants and corner correction terms [15].

The outline of this paper is as follows. In Section 2 we briefly revisit transfinite surface interpolation using tangential ribbons; this will be the basis of the subsequent discussions. In Section 3 we introduce the notion of auxiliary vertices and curves to adjust the shape interior, while in Section 4 the application of so-called interior surfaces will be discussed. Section 5 is devoted to describe one- and two-sided patches. Finally, examples and suggestions for future work conclude the paper.

2. Transfinite patches with ribbons

The boundary constraints of transfinite patches are computed from a curve network. In this section we revisit our companion paper [19] and briefly present a particular transfinite scheme out of the many, that blends together ribbon surfaces. Our discussion will be based on this scheme. Nevertheless, the following ideas can be expanded for other types of interpolants and parameterization schemes, as well.

Curve networks. In the classical paper of Nielson [10] a minimum norm network of cubic arcs were created over a planar triangular domain; in the paper of Moreton and Sequin [9] vertices and prescribed normals were interpolated by an optimized curve network. In our context the curve network is explicitly defined by the user, and one of the crucial tasks is to provide proper cross-derivative constraints for the individual surface patches. These are defined by the construction in an automatic manner. Space limitation prevents us to give a detailed description here, but we present the main steps of an algorithm that provides G^1 continuity between the adjacent patches.

The curves are cubic B-splines, and for each vertex of the network there exists a normal vector. For each curve we generate a Rotation Minimizing Frame (or a *normal fence* in our terminology), that interpolates the normals at the vertices and has the well-known property of minimizing the torsion of the local frame along the curve [20]. If needed, additional normals can also be specified by the user to adjust the shape of the normal fence. The cross-derivatives are computed edge by edge; a ribbon for a given patch boundary is forced to be perpendicular everywhere to the normal fence and at the two corners it interpolates the derivatives of the neighboring "left" and "right" boundaries. In this way, if patch A and B are both perpendicular to the normal fence, G^1 continuity is automatically satisfied. A sequence of curves, fences, ribbons and patches is shown later in Section 6.

For the patch formulation used in this paper ribbons with incompatible twists cause no problem, since the applied blending functions behave as Gregory's twists at the corners (see later). At the same time it is possible to create twist compatible ribbons by requesting that a common surface curvature is also associated with the vertices of the network (see Peters [11]). These types of networks are dominated by T-nodes and X-nodes (see e.g. Figure 15), for which it is easy to determine surface curvatures. For general n-valent vertex configurations the best surface curvature can only be set in a least-squares sense, which may lead to some minor perturbation of the incoming curves (see Várady and Hermann [18]). Networks that match not only normal vectors, but surface curvatures as well, produce more pleasing surface models.

Our n-sided patch is defined over a non-regular, convex polygonal domain as a convex combination of ribbon surfaces:

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

The polygonal domain is defined in the (u, v) plane, and the sides of the polygon, Γ_i, correspond to the individual patch boundaries in 3D.

Ribbon surfaces are parametric surfaces with their local parameters s_i and d_i, given as

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

$P_i(s_i)$ is the boundary curve along the i-th side, and $T_i(s_i)$ is the cross-tangent function associated with the boundary. The local parameters of a ribbon depend on u and v; $s_i = s_i(u, v)$ is the *side parameter*, and $d_i = d_i(u, v)$ is the *distance parameter*, that represents some distance measure. Roughly speaking, d_i is 0 on the i-th side and increases in a monotonic way as we move inwards.

Blending functions. The ribbons are weighted by special blending functions $\mu_i(d_1 \ldots, d_n)$ defined over the full domain. Let $D^n_{i1,i2,\ldots,in}$ denote $\prod^n_{i\neq i1,i2,\ldots,in} d_i^2$. Then

$$\mu_i(d_1, \ldots, d_n) = \frac{D^n_i}{\sum_{j=1}^n D^n_j}.$$

μ_i is equal to 1 along side i, and 0 for the remaining $n-1$ sides. For all domain points the μ_i-s have the partition of unity property. A blending function is shown in Fig. 1. These types of blending functions are 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:

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

These blending functions ensure that the ribbons will be reproduced along the sides. For simplicity's sake, we take an arbitrary point on the i-th boundary and evaluate the contribution of the i-th ribbon as a function of d_i only,

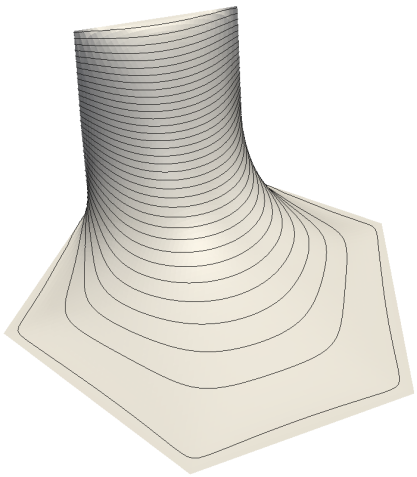


Fig. 1. Blending function with contours

i.e., $\hat{S}_i(d_i) = \hat{R}_i(d_i)\hat{\mu}_i(d_i)$. In order to interpolate the positional data, $\hat{\mu}_i(0)$ must be equal to 1. The cross-derivative can be written as $\hat{S}_i'(d_i) = \hat{R}_i'(d_i)\hat{\mu}_i(d_i) + \hat{R}_i(d_i)\hat{\mu}_i'(d_i)$, so in order to reproduce the tangential data, $\hat{\mu}_i'(0)$ must be 0. These requirements correspond to the properties stated above. For G^1-continuous cross-derivative constraints, it is sufficient to use quadratic terms; for G^2 constraints cubic terms are needed. It can easily be shown, that this also guarantees that the effect of the other boundaries and their cross-derivative functions will vanish on the i-th side.

Take an example, for $n = 4,\ i = 1$,

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

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

Note: an equivalent, but computationally more efficient formula can be used to evaluate the blending functions at the interior points of the domain:

$$\mu_i(d_1, \ldots, d_n) = \frac{d_i^{-2}}{\sum_{j=1}^{n} d_j^{-2}}.$$

This expression is singular on the sides, and there the original formula must be used, or the value of the curve substituted as they will be the same.

Domain. Our goal is to determine an appropriate *non-regular* convex domain based on the given loop of 3D boundary curves. This was found to be a useful enhancement with respect to formerly suggested transfinite schemes, that helps to avoid undesirable shape artifacts when the lengths of the boundaries significantly differ. The sides and the angles of the domain are denoted by l_i and α_i, respectively. The arc-lengths of the given 3D boundary curves are denoted by L_i, and the angles between the end tangents of the $(i-1)$-th and i-th boundaries by ϕ_i. We seek to minimize the squared deviation of the chord lengths and the angles $\sum(l_i - c_{\text{length}} L_i)^2 + \sum(\alpha_i - c_{\text{angle}}\phi_i)^2$, where c_{length} and c_{angle} are properly chosen constants. This is a non-linear problem, but the following simple heuristic method was found to be satisfactory in practice.

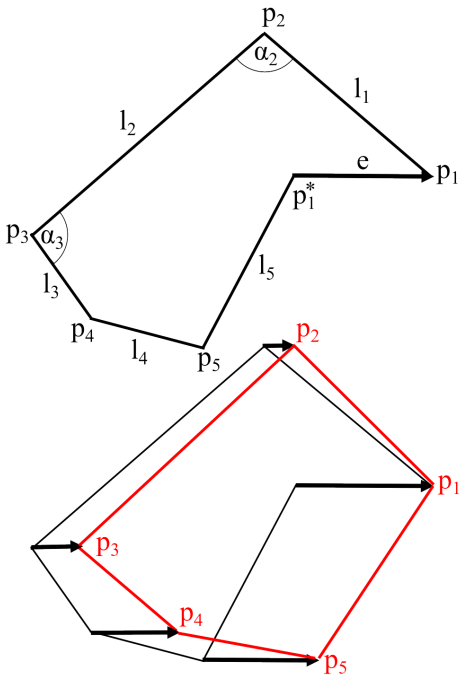


Fig. 2. Length/angle-based convex polygonal domain

First we normalize the angles to satisfy the necessary angle criterion for the n-sided domain 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 a sequence retaining the angles, which will likely yield an open polyline, 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 next ones sequentially, first by $\frac{1}{n}\underline{e}$, then by $\frac{i}{n}\underline{e}$ (Fig. 2). As a result, we obtain a closed polygon where both the chord lengths and the angles are distorted only to a small extent. This technique is known as edge-tweaking and has been used earlier in [16].

Parameterization. A critical part of transfinite schemes is parameterization. It helps define the shape of the patch and its differential properties along the sides. Having a given point in the domain (i) we need to determine n corresponding 3D data points on the n individual interpolants and (ii) combine these by the corresponding blending functions. The ribbon mapping $(u, v) \to (s_i, d_i)$ produces local ribbon coordinates to be substituted into $R_i(s_i, d_i)$; and $(u, v) \to d_i$ produces n distance values to compute the weights of the blending functions $\mu_i = \mu_i(d_1, \ldots, d_n)$.

In [19] a method called *central line sweep* was suggested, that ensures a balanced distribution of the iso-parameter lines of the individual ribbons, and its computation is reasonably simple. The sweep lines run from the left edge Γ_{i-1} to the right edge Γ_{i+1} of the domain polygon in such a way, that the middle line of the ribbon is 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. 3. This method helps to avoid skewed parameterizations and produces better results than the classical radial line sweep of Gregory et al. [6].

So having a parameterizing function $r(s, d)$, the $s = 0.5$

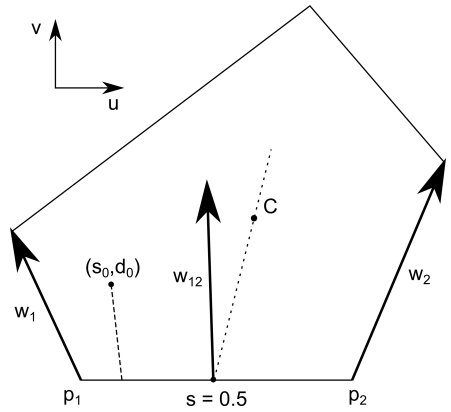


Fig. 3. Computing the central line sweep parameterization

constant parameter line must contain the center point, i.e., for some unknown d_c parameter value $r(0.5, d_c) = c$. Without loss of generality, we deal with the local parameters of side 1 and for simplicity's sake we position the corner p_1 at the origin, and place p_2 on the u-axis. A linear by quadratic map is introduced:

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

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

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

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. 1, then solving the quadratic equation for s_0 we obtain the required local coordinates of the sweep line:

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

The central sweep line parameterization is depicted in Figure 4, where the sweep lines are shown for the short top-right edge and the top edge.

In the next sections we will discuss how the interior of these patches can be modified by various techniques. We do not go into details of ribbon construction here; just assume that the ribbons were computed from the given 3D curve network, and they guarantee tangent plane continuity across the common boundaries of adjacent patches, and for each patch twist compatibility at the corners is satisfied.

3. Simple shape modifications

Adjusting ribbons. In curve network-based design, feature curves are the basic entities to define a shape, but the

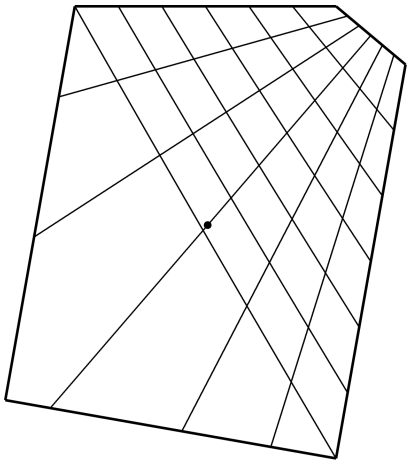


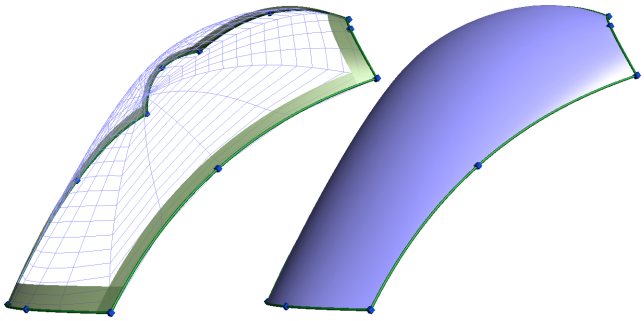
Fig. 4. Sweep lines in a five-sided domain

interior of the patches are not uniquely defined, and ribbons can provide further shape control. Assume that the boundaries and the cross-tangent directions are given. The most straightforward editing operation is to set the magnitudes of the ribbons, as these balance how much the surface patch is "glued" to the ribbons in the vicinity of the boundaries, and where the convex combination starts to dominate as we are moving inwards. The simplest solution is to multiply the direction terms by w_i scalar values or scalar reparameterization functions, then

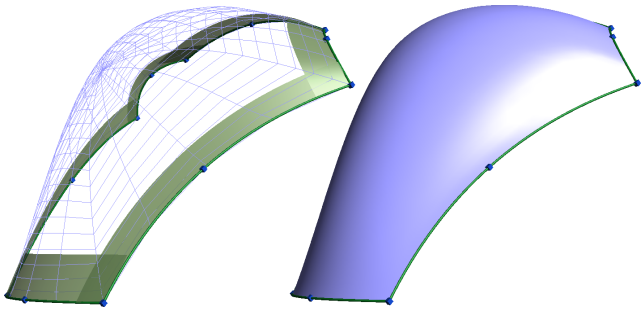
$$R_i(s_i, d_i) = P_i(s_i) + d_i w_i(s_i) T_i(s_i).$$

Adjusting simultaneously the magnitude of the ribbons yields a global change affecting the "fullness" of the patch, as shown in Figures 5a–5b. Modifying the width of an individual ribbon creates a local effect as shown in Figure 6. Here a better curvature distribution is obtained by narrowing the top-right ribbon. The latter example illustrates that ribbon width multipliers can support fairing procedures, as well. This is going to be the subject of another publication; here we just note that by means of the ribbons widths it is possible to optimize fairness energies not only for individual patches, but for collections of adjacent patches as well.

Auxiliary vertices and curves. While we wish to preserve the basic interpolatory nature of curve network-based design, it is possible to assign further entities to the interior of the patch and provide shape control directly where it is needed. Think of lifting certain interior vertices or prescribe interior feature curves while the external ribbon constraints are retained. Recall that in Section 2, the only property of the blending functions was that the i-th blend is 1 on the i-th boundary and 0 on all other boundaries. By definition, an auxiliary element has an image within the domain, and a distance measure can be defined, which guarantees that it becomes zero on the image of the auxiliary element. Then the corresponding blending function will be 1 there and will vanish elsewhere. This also means that the patch equation needs to be modified, and $n + k$ entities will be blended together (k denotes the number of the auxiliary elements). The blending functions μ_i will also change and

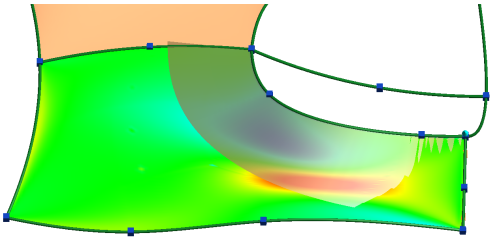


(a) Weight=0.25

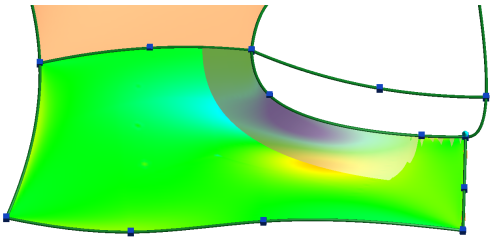


(b) Weight=0.5

Fig. 5. Adjusting fullness for a six-sided patch (spider lines, shaded).



(a) Initial ribbon

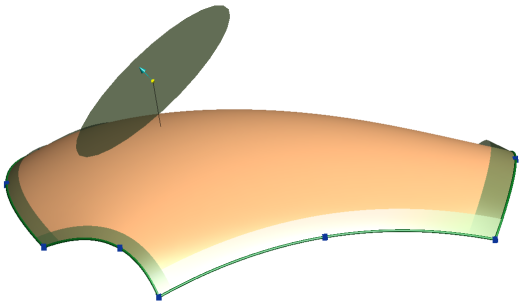


(b) Modified ribbon

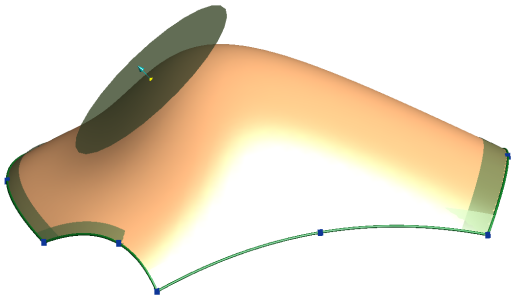
Fig. 6. Ribbon modification of a five-sided patch.

instead of the D_i^n terms, now D_i^{n+k} will be used combining $n+k$ distance values.

This concept is illustrated by two simple examples. In the first, an auxiliary vertex has been chosen on the surface, which defined its parametric position. Lifting the vertex and creating a circular ribbon around it determines the local properties of the modified surface; see Figures 7a and 7b. The circular ribbon is given as $R_i(s_i, d_i) = P_i + d_i w_i T_i(s_i)$; P_i is the center point, N_i is the normal vector, w_i is the weight to set the magnitude of the ribbon. $T_i(s_i)$ represents a rotating line which is always perpendicular to N_i, and is defined by an angular parameter s_i in the domain ($s_i \in [0, 2\pi]$). The other parameter d_i gives the distance from the image of P_i in the domain.



(a) Auxiliary vertex and ribbon



(b) After snapping

Fig. 7. Snapping a five-sided patch to an auxiliary vertex.

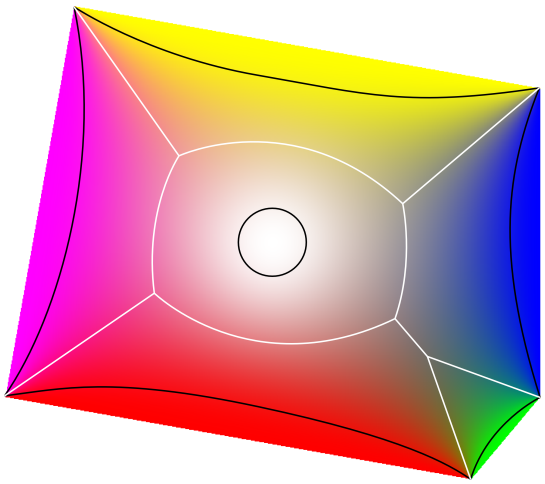
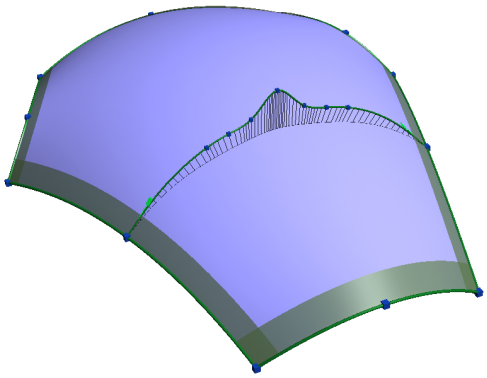


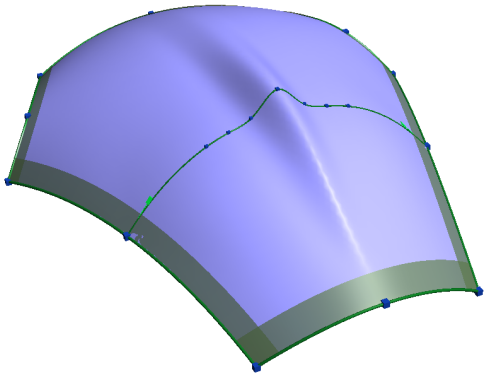
Fig. 8. Blending function distributions in the domain.

The distribution of the blending functions are illustrated in Figure 8, where the "strength" of the blends is shown. The black curves show the areas where the influence of the i-th blend is more than 90 percent, the white curves show when the influence between the i-th and j-th blends is equal, thus providing a Voronoi-like structure in the domain.

The second example shows an auxiliary curve. Initially, the curve is also defined on the surface, in order to obtain its parametric image in the domain. Its equation can be given as



(a) Auxiliary curve and its image on the surface



(b) After snapping

Fig. 9. Snapping a six-sided patch to an auxiliary curve.

$$R_i(s_i, d_i) = P_i(s_i) \pm d_i w_i(s_i) T_i(s_i).$$

A point in the domain will lie either on the "left" or "right" side of the image of the auxiliary curve, this will determine the sign of T_i.

After lifting the points of the curve a new profile is created (Fig. 9a). As we snap the surface onto it, the interior will change (Fig. 9b). By adjusting the width of the associated ribbon the strength of the lifting effect can be controlled.

4. Interior patches

In the previous section we have introduced auxiliary vertices and curves. We have increased the number of interpolants, but used the same family of blending functions with $n+k$ terms. Now we are going to introduce a so-called interior surface $S_{\text{int}}(u, v)$, which is defined over the same domain and serves to modify the interior of the original $S(u, v)$. Here we are going to apply alternative blending functions. Let

$$S^*(u, v) = \sum_{i=1}^{n} R_i(s_i, d_i)\nu_i(d_1, \ldots, d_n) + S_{\text{int}}(u, v)\nu_0(d_1, \ldots, d_n).$$

We use the notations of Section 2 with the additional term of $D_0^n = \prod_{j=1,n} d_j^2$. Then the blending functions are defined as

$$\nu_i(d_1, \ldots, d_n) = \frac{D_i^n}{\sum_j D_j^n + wD_0^n},\ i = 1, \ldots, n,$$

and

$$\nu_0(d_1, \ldots, d_n) = \frac{wD_0^n}{\sum_j D_j^n + wD_0^n}.$$

Here w is a positive constant characterizing the blend family, this will be set later. As it can be seen, the side blends have an extended denominator, which will not change the basic properties, i.e., $\nu_i = 1$ on the i-th side and 0 on the other sides. The new blending function ν_0 is 0 on each side, which means the interior surface will have no effect on the boundaries.

The above surface equation can be formulated in another way:

$$S^*(u, v) = \alpha S(u, v) + (1 - \alpha) S_{\text{int}}(u, v) \qquad (2)$$

where

$$\alpha = \frac{\sum_j D_j^n}{\sum_j D_j^n + wD_0^n}.$$

This expression reproduces S along the edges and gives a weighted average of the original and the interior surface inside the patch. We define the constant w by means of α. Take a domain point c as center point; this is where the weighted average is prescribed. At point c let us evaluate all distances, thus we obtain constant terms

$$E_j^c = D_j^n(d_1, d_2, ..., d_n),\ j = 1, \ldots, n.$$

Then $\alpha = \frac{\sum_j E_j^c}{\sum_j E_j^c + wE_0^c}$ and $w = \frac{(1-\alpha)\sum_j E_j^c}{wE_0^c}$. After dividing by E_0^c we obtain an alternative expression of $w = \sum_j \frac{1-\alpha}{d_j^2}$, where the non-zero distances d_j are determined by c.

$\alpha = 0.5$ will average the original and the interior surface at the center point. If we want to interpolate the interior surface at c and tightly approximate it in the vicinity of c, another surface, called the auxiliary surface $S_{\text{aux}}(u, v)$, needs to be used in Equation 2 above. Let us assume that

$$S_{\text{int}}(u, v) = \alpha S(u, v) + (1 - \alpha) S_{\text{aux}}(u, v)$$

then

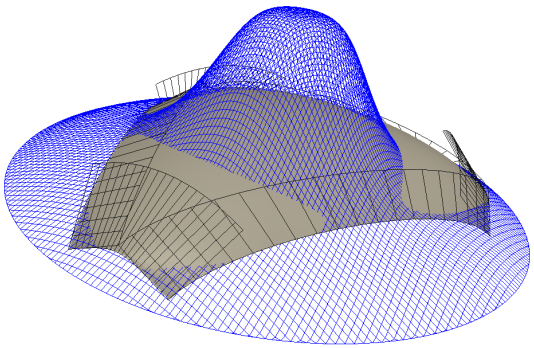
$$S_{\text{aux}}(u, v) = \frac{S_{\text{int}}(u, v) - \alpha S(u, v)}{1 - \alpha}$$

For example, at $\alpha = 0.5$

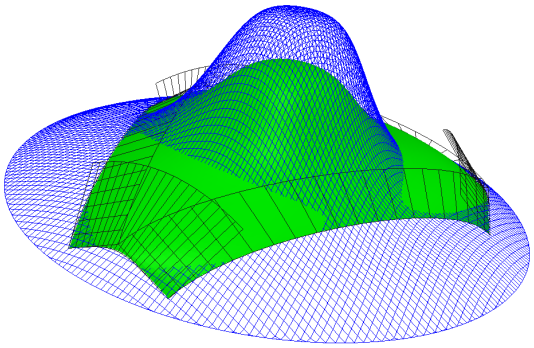
$$S_{\text{aux}}(u, v) = 2 \cdot S_{\text{int}}(u, v) - S(u, v).$$

(Note, that instead of linear blending by α, Hermite functions can also be used.)

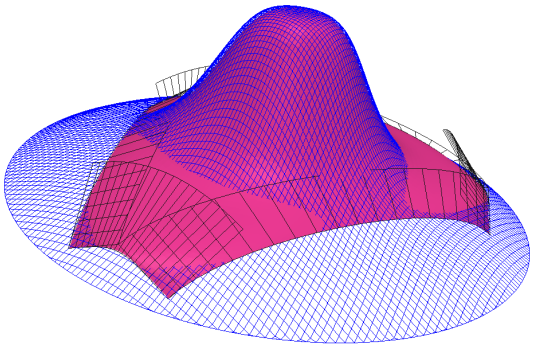
The parametric assignment of S and S_{int} can be realized in a projective sense, as earlier. Imagine that we have already computed the surface S; then any point on the interior surface with parameters (u^*, v^*) can be projected back to S, which will yield a parameter pair (u, v) for creating a parametric assignment to combine the points of the two surface entities. The effect of using interior surfaces is demonstrated in Figure 10. The first one shows the input: a



(a) Patch with its ribbons and an interior surface (blue)



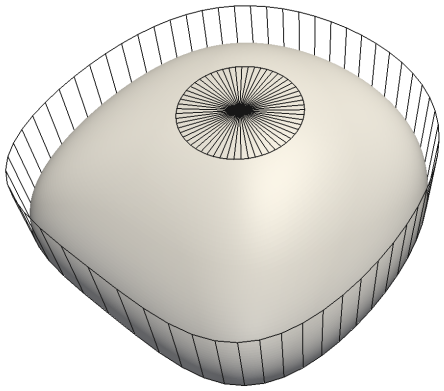
(b) Superimposing the interior surface



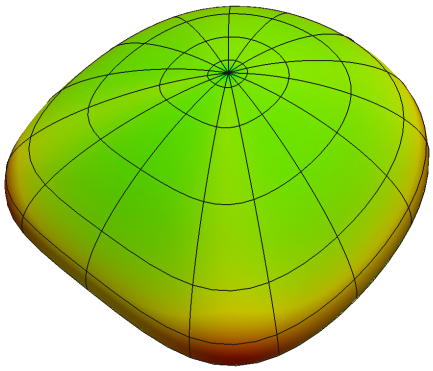
(c) Superimposing the auxiliary surface

Fig. 10. Reshaping the interior of a six-sided patch.

patch to be modified and the interior surface used for shape adjustment. The second picture shows the averaging effect, while the third one illustrates how we can reproduce the interior surface using a corresponding auxiliary surface. In both cases the original boundary constraints are retained.



(a) Patch and its ribbons



(b) Mean curvature and spider lines

Fig. 11. One-sided patch example "cap".

5. One- and two-sided patches

In practical curve network-based design, one- and two-sided patches often occur. These also must interpolate the boundaries and match the ribbon surfaces determined by the network. Fortunately, sweep line parameterization and distance-based blending can be applied in a similar way as before.

One-sided patches. Take a closed curve $r(t)$ and a center point c in 3D and associate an additional ribbon with it. Let us use a circle as domain with radial sweep lines. We apply the same solution as for auxiliary points in Section 3 and combine the two ribbons by simple blending functions of the type

$$\mu_i(d_1, d_2) = \frac{d_j^2}{d_1^2 + d_2^2}, \; i, j \in \{1, 2\}, \; i \neq j, \tag{3}$$

where d_1 and d_2 represent the distances in the domain from the perimeter circle and from the center point, respectively. As an example, Figure 11a shows the defining ribbons and the corresponding cap-like surface patch. Figure 11b shows the curvature map of the surface together with spider-like constant parameter lines drawn on the surface in 3D.

One interesting issue is to find a good location for the center. In the majority of cases this will be set by the user, however, setting a good default may be necessary. One simple heuristic is to optimize the angles between the imag-

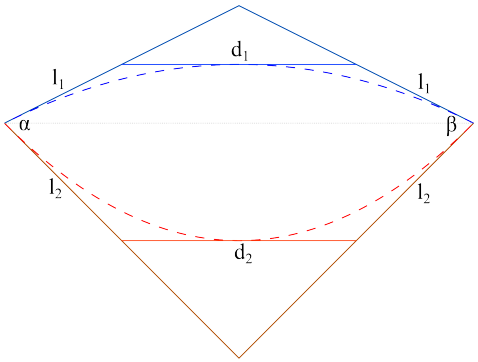


Fig. 12. Computing a two-sided domain

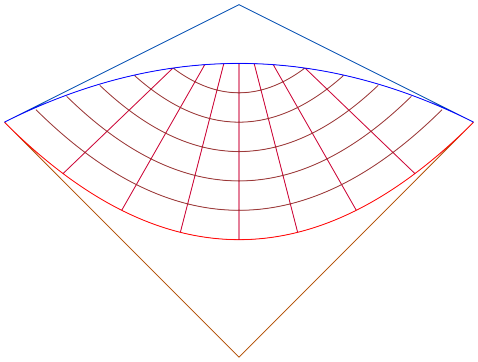


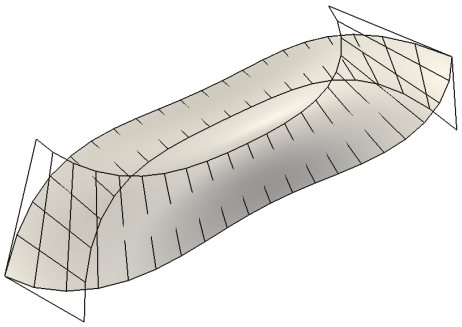
Fig. 13. Sweep lines in a parametric domain

inary 3D sweep lines and the tangents at sampled data points on the boundary, i.e.,

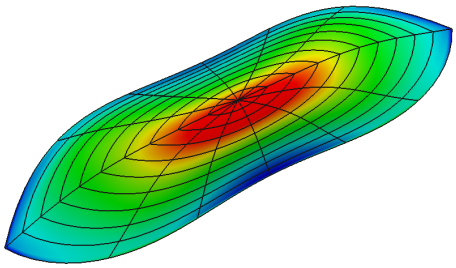
$$\sum_i (\langle c - r(t_i), \dot{r}(t_i) \rangle)^2 + \omega |c - r(t_i)|^2 = \min .$$

The second term is needed to control the sum of the chord lengths between the center point and the points of the boundary; without this, the minimum found by the related system of equations would push the center point infinitely far from the closed boundary curve. The constant ω can be manually adjusted.

Two-sided patches. The domain of the two-sided patch is bounded by two parabolic arcs. As it was written in Section 2, we search for a domain that mimics simultaneously the 3D angles between the two given boundary curves and their arc lengths. Figure 12 shows a simple, heuristic solution using a quadrangle. We inherit the 3D angles denoted by α and β, and define the parabolas in such a way, that their approximate arc lengths are proportional to the 3D boundaries. The arc length of a parabolic arc is estimated by $2l_i + d_i$, as shown in the figure, and simple algebra determines the missing parameters. Having the domain, sweep lines are created by connecting the parabolas with the opposite corners of the quadrangle (see one set in Figure 13). Then normalized distances measured on the sweep lines yield the distance parameters, which are used for the same type of blending functions as above in Equation 3 to combine the two ribbons. A simple example is depicted in Figures 14a and 14b.



(a) A patch and its ribbons



(b) Mean curvature and spider lines

Fig. 14. Two-sided patch example.

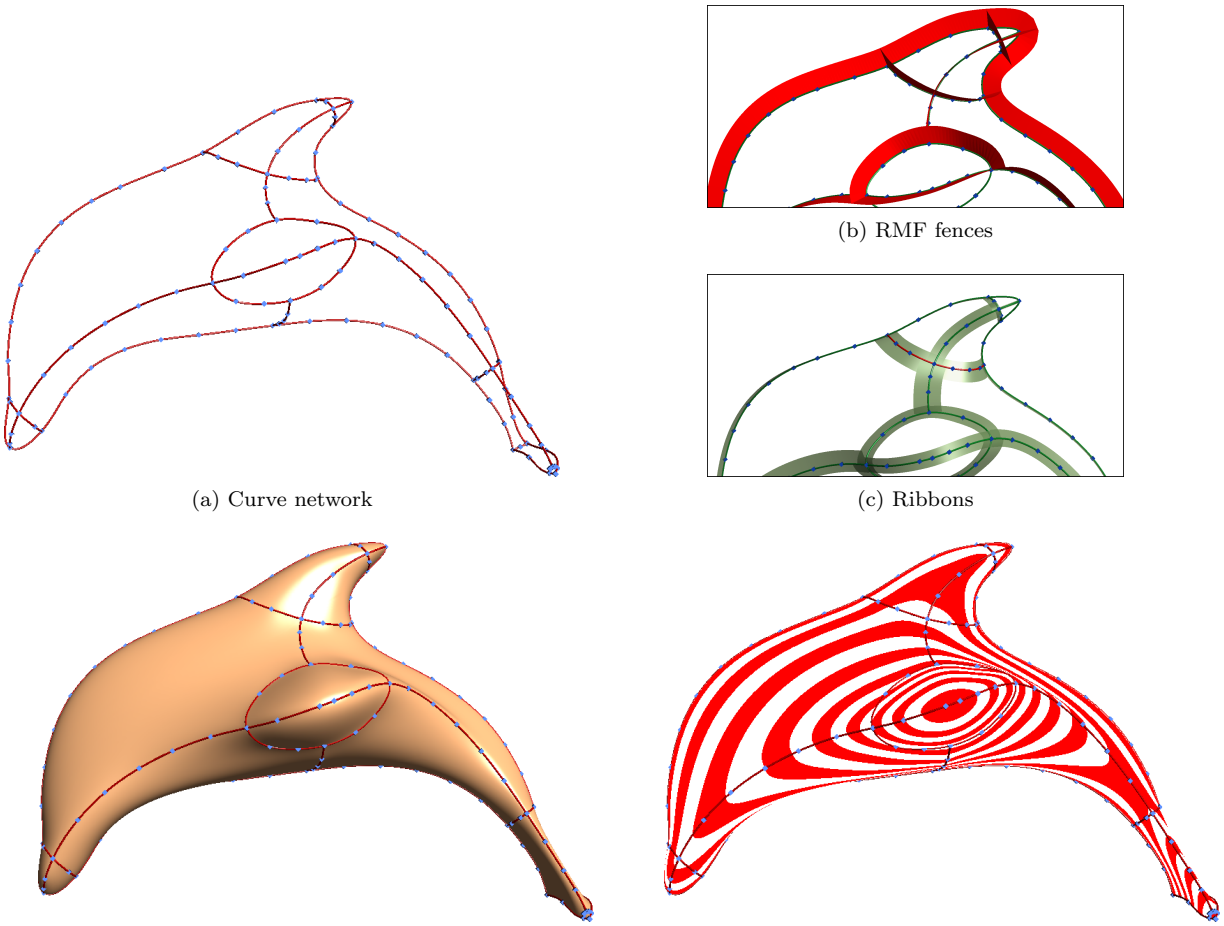
6. Curve network based design, an example

The following example illustrates the process of curve network based design. The original input curves were provided by Cindy Grimm (Washington University, see [7]); the transfinite surface patches were generated by a prototype system called Sketches, developed by ShapEx Ltd., Budapest.

Figure 15a shows the 3D network which is a collection of cubic B-splines. The network is dominated by T-nodes and X-nodes. Figure 15b illustrates RMF-based normal fences that interpolate the normal vectors computed at the vertices and provide a smooth normal vector function between them. The ribbons are perpendicular to the corresponding fences, thus smooth connection between the adjacent patches is guaranteed (15c). In this model there are two 2-sided, four 3-sided, four 4-sided, two 5-sided and two 6-sided patches (15d). In Figure 15c the red curve represents an auxiliary curve that helped to perfect the interior of the two 6-sided patches on the top part. The last picture with contouring (15e) illustrates G^1 continuity between the adjacent patches.

Conclusion, future work

In this work we revitalize curve network-based design. In contrast to control polyhedra-based approximating techniques, here the emphasis was placed on interpolation, using network of curves, ribbons associated with these curves and various techniques that support perfecting and fairing



(a) Curve network (b) RMF fences (c) Ribbons (d) The final model (e) Slicing map showing G^1 connections

Fig. 15. Creation of a dolphin model.

the interior of shapes. In addition to adjusting the widths of ribbons, additional entities — auxiliary vertices, curves and interior surfaces — were combined, applying variations of distance-based blending functions borrowed from transfinite patch interpolation.

Curve network-based design is a challenging paradigm, but there is plenty of space for future research. The automatic generation of compatible G^2 ribbons for transfinite surfaces is a necessary step forward; non-convex domains would also be useful. Although local surface interpolation has its fundamental advantages, it is a deficiency that there is no global energy to affect the overall smoothness of the surface models. The approach is highly sensitive to the fairness of the constituting curves and ribbons, so fairing is particularly important. Optimizing the widths of the ribbons for balanced curvature distributions and approximating measured data within a loop of curves are also interesting topics for the future.

Acknowledgements

Special thanks are due to the anonymous reviewers for their suggestions to enhance the paper. This work was partially supported by the scientific program "Development of quality-oriented and harmonized R+D+I strategy and functional model at the Budapest University of Technology and Economics" (UMFT-TÁMOP-4.2.1/B-09/1/KMR-2010-0002) and a grant by the Hungarian Scientific Research Fund (No. 101845). Some images of this paper were generated by the Sketches system developed by ShapEx Ltd, Budapest; the contribution of György Karikó is highly appreciated. The authors acknowledge support from the Geometric Modeling and Scientific Visualization Research Center of KAUST, Saudi-Arabia.

References

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

[2] S. A. Coons, Surfaces for computer-aided design of space forms, Tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA (1967).

[3] G. Farin, Curves and surfaces for CAGD: a practical guide, 5th ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[4] W. J. Gordon, Spline blended surface interpolation through curve networks, Journal of Mathematics and Mechanics 18 (10) (1969) pp. 931–952.

[5] J. A. Gregory, N-sided surface patches, in: The Mathematics of Surfaces, Oxford University Press, USA, 1986, pp. 217–232.
[6] J. A. Gregory, J. M. Hahn, A C^2 polygonal surface patch, Computer Aided Geometric Design 6 (1989) 69–75.
[7] C. Grimm, P. Joshi, Just draw it! a 3D sketching system, Tech. Rep. 2012-4, Washington University in St. Louis (2012).
[8] K. Kato, Generation of n-sided surface patches with holes, Computer-Aided Design 23 (10) (1991) 676–683.
[9] H. Moreton, C. Séquin, Surface design with minimum energy networks, in: Proceedings of the first ACM symposium on Solid modeling foundations and CAD/CAM applications, SMA '91, ACM, New York, NY, USA, 1991, pp. 291–301.
[10] G. M. Nielson, A method for interpolating scattered data based upon a minimum norm network, Mathematics of Computation 40 (161) (1983) 253–271.
[11] J. Peters, Joining smooth patches around a vertex to form a C^k surface, Comput. Aided Geom. Des. 9 (1992) 387–411.
[12] D. Plowman, P. Charrot, A practical implementation of vertex blend surfaces using an n-sided patch, in: Proceedings of the 6th IMA Conference on the Mathematics of Surfaces, Clarendon Press, New York, NY, USA, 1996, pp. 67–78.
[13] M. Sabin, Non-rectangular surfaces patches suitable for inclusion in a B-spline surface, in: Eurographics '83, North Holland, 1983, pp. 57–70.
[14] M. Sabin, Some negative results in n sided patches, Computer-Aided Design 18 (1) (1986) 38–44.
[15] P. Salvi, Fair curves and surfaces, Ph.D. thesis, Eötvös Loránd University, Budapest (2012).
[16] T. W. Sederberg, J. Zheng, A. Bakenov, A. Nasri, T-splines and T-NURCCs, in: ACM SIGGRAPH 2003 Papers, SIGGRAPH '03, ACM, New York, NY, USA, 2003, pp. 477–484.
[17] T. Varady, Overlap patches: a new scheme for interpolating curve networks with n-sided regions, Computer Aided Geometric Design 8 (1) (1991) 7–27.
[18] T. Várady, T. Hermann, Best fit surface curvature at vertices of topologically irregular curve networks, in: Proceedings of the 6th IMA Conference on the Mathematics of Surfaces, Clarendon Press, New York, NY, USA, 1996, pp. 411–427.
[19] T. Várady, A. Rockwood, P. Salvi, Transfinite surface interpolation over irregular n-sided domains, Computer Aided Design 43 (2011) 1330–1340.
[20] W. Wang, B. Jüttler, D. Zheng, Y. Liu, Computation of rotation minimizing frames, ACM Trans. Graph. 27 (2008) 2:1–2:18.