Multi-sided generalizations of the Coons patch

Péter Salvi

Budapest University of Technology and Economics

Abstract. The Coons patch is one of the simplest formulations for transfinite interpolation: given four C^0 or C^1 boundary constraints, it generates a smooth surface based on a Boolean sum logic.
Over the years there have been many generalizations of this construct to an arbitrary number of sides; here we give an overview of some of the methods in a common framework, highlighting their similarities and differences.

1 Introduction

Most of the objects around us – even such simple household objects as a pitcher (Fig. 1) – have such a structure that a natural partitioning results in non-four-sided regions. This is particularly common in curve network based design, where the model is created by its feature curves.

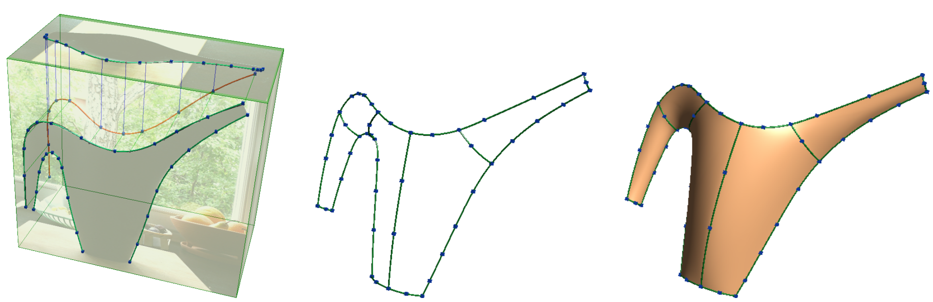


Fig. 1: Designing with a curve network.

In most CAD systems surfaces like these are either trimmed or split (Fig. 2). In trimming, a larger quadrilateral patch is created, then the parts not needed are cut off. A drawback of this method is that the boundaries are not exact, and there is an inherent asymmetry. The central split approach creates quadrilateral subpatches, but then maintaining continutiy between them becomes a problem.

In many applications it is also beneficial if we can create a surface just by boundary constraints, i.e., we need *genuine* multi-sided transfinite interpolation surfaces.

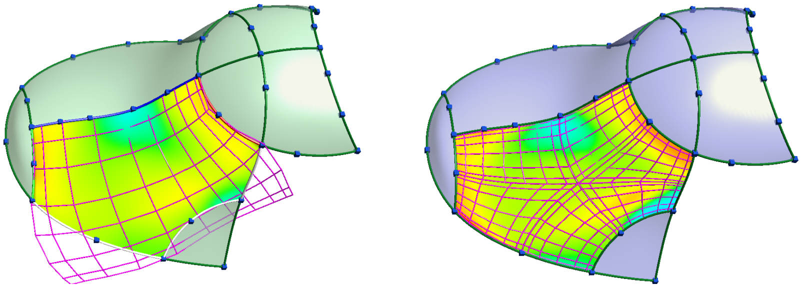


Fig. 2: Conventional handling of multi-sided loops: trimming and splitting.

Our search starts with the Coons patch [2]. In its simplest form, we create two ruled surfaces, $\mathbf{S}_1$ and $\mathbf{S}_2$, connecting opposite sides, and subtract a bilinear surface ($\mathbf{S}_{12}$) interpolating the corners (see also Fig. 3):

$$\mathbf{S} = \mathbf{S}_1 + \mathbf{S}_2 - \mathbf{S}_{12}. \tag{1}$$

The resulting patch naturally interpolates all boundaries.

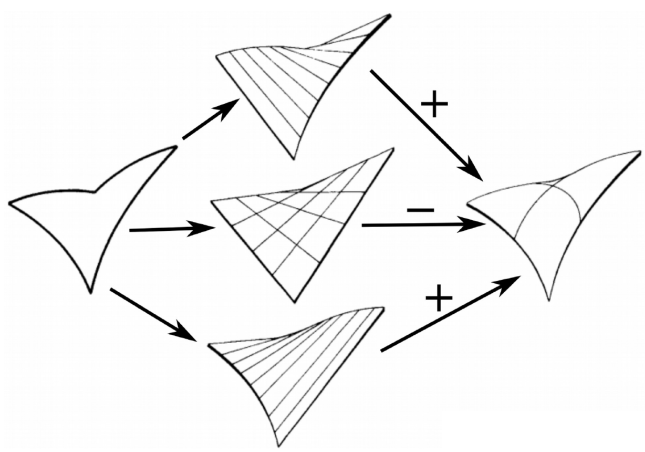


Fig. 3: Coons patch construction (after [3]).

This construction is easy to extend to satisfy C^1 boundary constraints, as we will see below. In both cases, when the constraints are polynomial, the same result can be achieved by solving a linear system on the control points of a Bézier surface; the equations coming from a 3×3 (C^0) or 5×5 (C^1) mask [4]. This approach can be generalized to multi-sided control–point-based surfaces, such as the S-patch [7] or the generalized Bézier patch [14], see e.g. a similar technique in [8].

Here, however, we are going to deal with the general case of arbitrary boundary constraints. Also, the reader should be advised that while the most important equations will be given, the focus is on the ideas of the various generalizations, and for the details one should refer to the original publications.

2 Ribbon-based constructions

The general approach we are going to take is to regard the boundary constraints as *ribbons* – four-sided surfaces that need to be interpolated along one of their boundary curves, possibly with G^1 continuity. These ribbons themselves interpolate one or more of the curves that comprise the boundary loop of the surface to be created: (i) *trilateral* ribbons interpolate three consecutive boundary curves, (ii) *bilateral* ribbons interpolate two, while (iii) *unilateral* ribbons interpolate only one.

2.1 Patches with trilateral ribbons

First we are going to look at patches using trilateral ribbons, i.e., four-sided surfaces interpolating three of the original boundaries.

Generalized C^0 Coons patch [9] The basic idea is to create C^0 Coons patches interpolating three sides, and blend these together (Fig. 4).

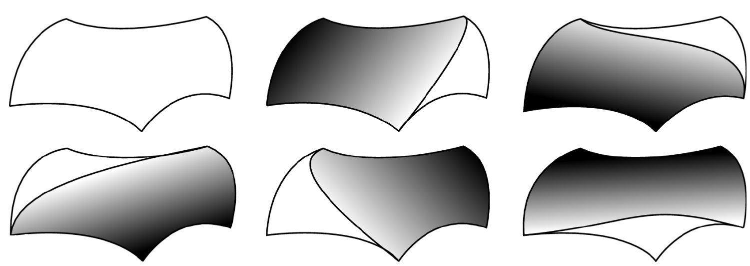


Fig. 4: Ribbons of a generalized C^0 Coons patch.

There are several open questions:

1. How do we define the fourth side of the ribbons?
2. What is the domain of the multi-sided patch?
3. How can we parameterize this domain? In other words, how should we map points of the domain to the local domain of each ribbon surface?
4. What blending function should we use?

A simple answer to the first question is to take the first derivatives of the adjacent curves and create a cubic Bézier curve (Fig. 5).

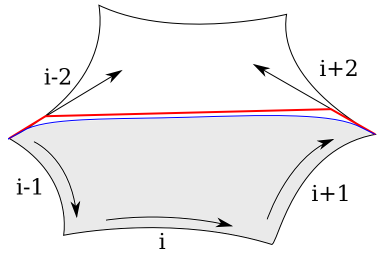


Fig. 5: Ribbon construction for the generalized C^0 Coons patch.

As for the domain, we can use regular n-sided polygons, and map their points using generalized barycentric coordinates, e.g. Wachspress coordinates [5]. Each ribbon has two local parameters: a side parameter (s_i) that goes from 0 to 1 as we go along the base side, and a distance parameter (d_i) that measures the distance from the base side, increasing to 1 as it reaches the distant sides:

$$s_i = \lambda_i \,/\, (\lambda_{i-1} + \lambda_i), \qquad\qquad d_i = 1 - (\lambda_{i-1} + \lambda_i). \tag{2}$$

Here λ_i is the generalized barycentric coordinate associated with the i-th side (assuming cyclic indexing). Figure 6 shows constant parameter lines on a five-sided domain.

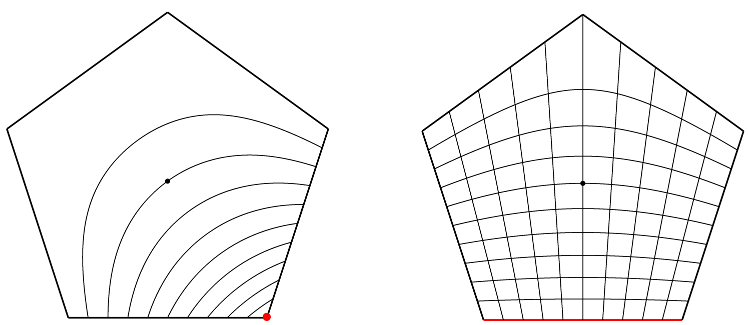


Fig. 6: Parameterization: Constant parameter lines of the Wachspress coordinates (left) and the side/distance parameters (right). The red point shows the base corner, and the red line the base side.

Finally, the blending function can be defined simply as

$$B_i^0 = 1 - d_i, \tag{3}$$

so the whole patch equation becomes

$$\mathbf{S} = \frac{1}{2} \sum_i \mathbf{C}_i^0 B_i^0, \tag{4}$$

where $\mathbf{C}_i^0$ $(i = 1 \dots n)$ denote the ribbon surfaces.

Figure 7 shows a schematic representation of the blending function. The red lines show the blending function associated with the bottom side. It gives 1 at the base side, diminishing on the adjacent sides, and vanishing on all other sides. If we look at the contribution of all ribbons at the bottom side, we see that there is 1 from the red one, and the green and blue ones also sum to 1, which is why we need a multiplier of $\frac{1}{2}$ in Eq. (4).

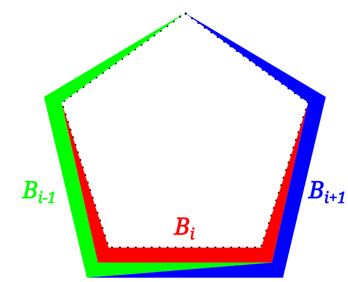


Fig. 7: Blending function for the generalized C^0 Coons patch.

An example with mean curvature and contouring is shown on Fig. 8.

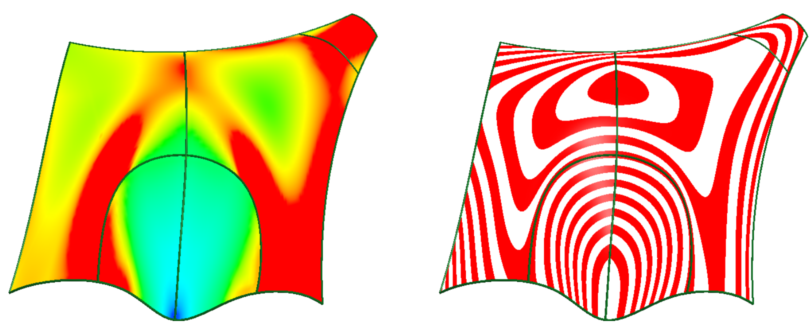


Fig. 8: Generalized C^0 Coons patch showing mean curvature and contouring.

Composite ribbon patch [13] While the above construction gives nice surfaces, it is often important to satisfy G^1 continuity with adjacent patches. Let us review the classic C^1 Coons patch first, but with a ribbon-based interpretation.

The equation is the same as in Eq. (1), but now instead of ruled surfaces, we have Hermite blends of linear side interpolants $\mathbf{R}_i$ in both $\mathbf{S}_1$ and $\mathbf{S}_2$, and Hermite blends of bilinear corner correction patches $\mathbf{Q}_{i,i+1}$ in $\mathbf{S}_{12}$ (Fig. 9).

Once again, we would like to create ribbons interpolating three sides. This time we leave the fourth side floating, and define the ribbons as the Hermite

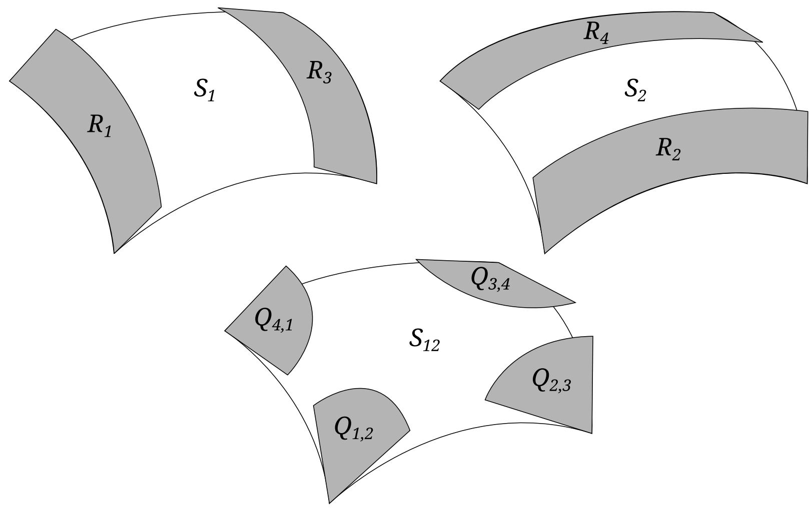


Fig. 9: C^1 Coons patch construction with ribbons.

blend of three side interpolants and two corner correction patches (Fig. 10a):

$$\mathbf{C}_i = \mathbf{R}_i^l \alpha_0(s) + \mathbf{R}_i \alpha_0(d) + \mathbf{R}_i^r \alpha_1(s) - \mathbf{Q}_i^l \alpha_0(s)\alpha_0(d) - \mathbf{Q}_i^r \alpha_1(s)\alpha_0(d). \tag{5}$$

Here $\alpha_0(x) = 1 - \alpha_1(x) = 2x^3 - 3x^2 + 1$ is a Hermite blend function.

We also need a different blending function that satisfies some derivative constraints; for this we use a variant of Shepard interpolation:

$$B_{i-1,i} = \frac{\prod_{k \notin \{i,i-1\}} d_k^2}{\sum_j \prod_{k \notin \{j,j-1\}} d_k^2}, \qquad B_i = B_{i-1,i} + B_{i,i+1}. \tag{6}$$

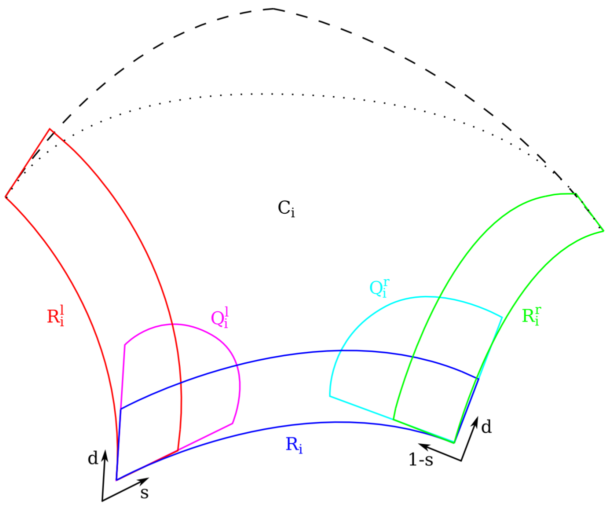
The function $B_{i-1,i}$ gives 1 at the base corner, and diminishes on the adjacent sides, vanishing on all other sides. The sum of two such blending functions fits the bill perfectly (see the schematic representation in Fig. 10b). The patch equation now becomes

$$\mathbf{S} = \frac{1}{2} \sum_i \mathbf{C}_i B_i. \tag{7}$$

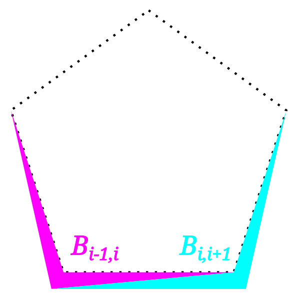
An example with mean curvature and isophote lines is shown on Fig. 11. The 5- and 6-sided patches are symmetric except for a corner cut off on the right side. From the curvature and isophote lines it can be seen that the surface is quite stable.

2.2 Patches with bilateral ribbons

Next we are going to look at constructions using *corner interpolants*, i.e., ribbons interpolating two consecutive sides.



(a) Ribbon construction



(b) Blending function

Fig. 10: Composite ribbon patch.

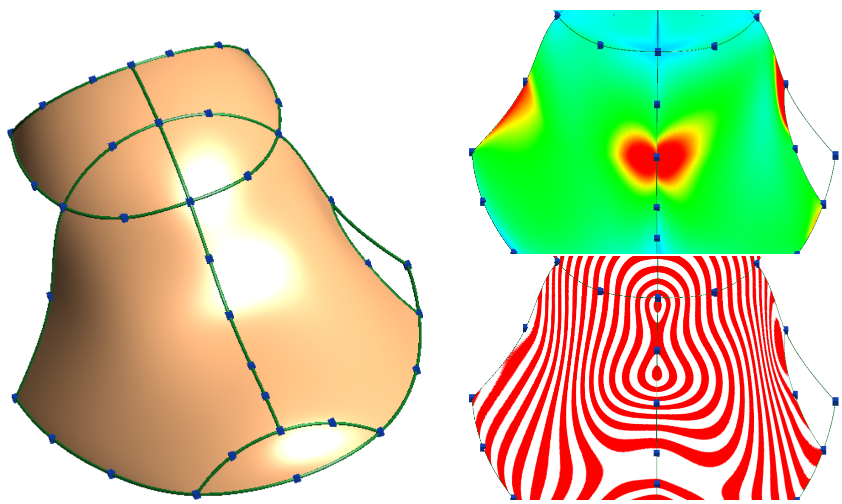


Fig. 11: Composite ribbon patch example showing mean curvature and isophotes.

Charrot–Gregory patch [1] Such a corner interpolant can be created as a partial Coons patch (see also Fig. 12):

$$\mathbf{R}_{i-1,i} = \mathbf{R}_{i-1} + \mathbf{R}_i - \mathbf{Q}_{i-1,i}, \tag{8}$$

and then the surface equation is simply

$$\mathbf{S} = \sum_i \mathbf{R}_{i-1,i} B_{i-1,i}. \tag{9}$$

Here we apply the blending function defined above in Eq. (6).

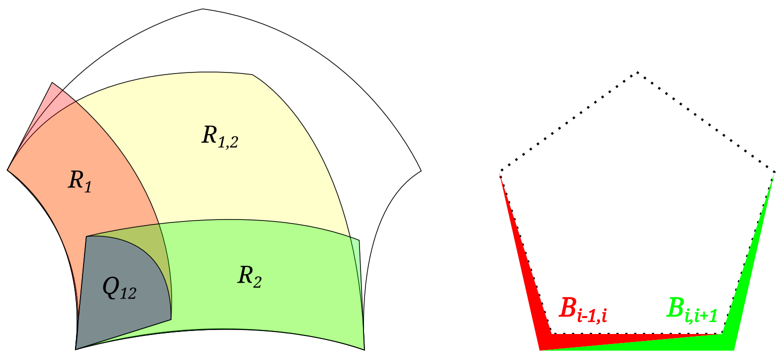


Fig. 12: Construction of the Charrot–Gregory patch.

Midpoint patch [12] There is an alternative blending function using the Hermite blends:

$$B^M_{i-1,i} = \frac{d_{i-1}\alpha_0(s_i)\alpha_0(d_i) + d_i\alpha_1(s_{i-1})\alpha_0(d_{i-1})}{d_{i-1} + d_i} \tag{10}$$

Since these functions do not sum to 1, there is a *weight deficiency*, so we get an extra degree of freedom. We can use this to control the surface interior, for example by moving the center point (Fig. 13).

2.3 Patches with unilateral ribbons

Finally, we move on to surfaces using *side interpolants*, i.e., ribbons interpolating only one boundary.

Katō's patch [6] Probably the simplest construction is to use a singular blending function (Fig. 14a) that gives 1 on the base side and vanishes on all other sides, so the interpolation property is trivially satisfied:

$$B^*_i = \frac{\prod_{k\neq i} d_k^2}{\sum_j \prod_{k\neq j} d_k^2}. \tag{11}$$

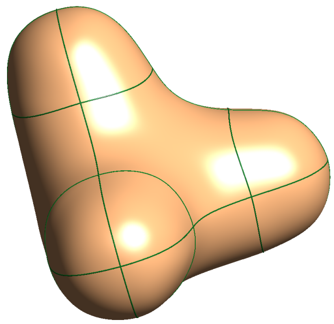
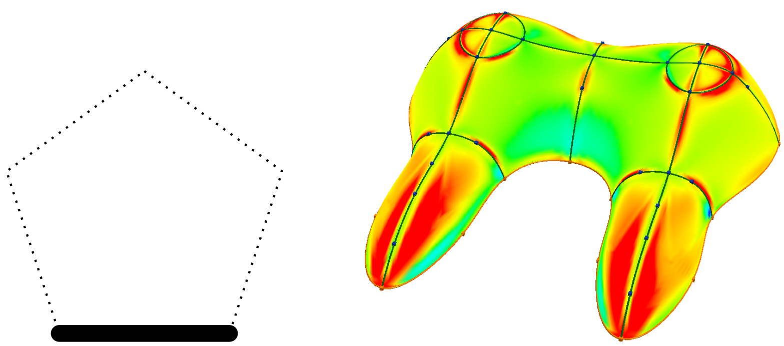


Fig. 13: Modifying the surface center.

Now the surface is just the blended sum of ribbons:

$$\mathbf{S} = \sum_i \mathbf{R}_i B_i^*. \tag{12}$$

Unfortunately most curvature variation tends to be concentrated near the boundaries (see Fig. 14b), so this formulation is not optimal in terms of quality.



(a) Blending function (b) Example showing mean curvature

Fig. 14: Katō's patch.

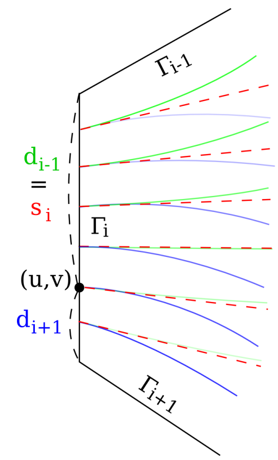
Generalized Coons patch [13] Another approach is to generalize the ribbon-based formulation of the Coons patch shown in Fig. 9:

$$\mathbf{S} = \sum_i \mathbf{R}_i B_i - \sum_i \mathbf{Q}_i B_{i-1,i}. \tag{13}$$

Note that here we employ both of the blends defined in Eq. (6).

This is probably the most natural generalization of the Coons patch, as it retains its basic idea: the final surface is constructed as the sum of interpolants, from which the unnecessary data is subtracted via correction patches.

However, there are further constraints on the parameterization (see Figure on the right). The green lines show distance parameters associated with the top side; the blue lines are distance parameters associated with the bottom side. The constraint says that these should have the same tangents along the left side.



It is easy to show that this is actually a 1-dimensional hole filling problem on its own, that can be solved by some other representation, for example by Katō's patch (Fig. 15). An example of a constrained parameterization is shown in Fig. 16, and a complex model with contouring in Fig. 17.

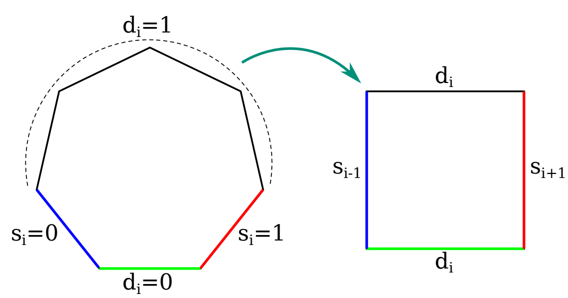


Fig. 15: Constrained parameterization as hole filling.

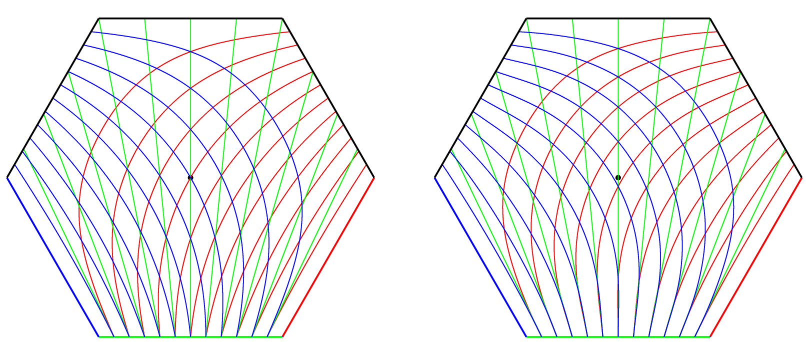


Fig. 16: Parameterization without and with constraints on a 6-sided domain.

Midpoint Coons patch [10] Replacing the blending function with the weight-deficient one gives us a variant of the midpoint patch:

$$\mathbf{S} = \sum_i \mathbf{R}_i B_i^M - \sum_i \mathbf{Q}_i B_{i-1,i}^M, \tag{14}$$

where $B_i^M = B_{i-1,i}^M + B_{i,i+1}^M$.

Summary

Table 1 summarizes all the methods outlined above, showing the type of ribbon and blending function needed, as well as the constraints on the parameterization (here *full* means that d should take the value 1 on the far sides).

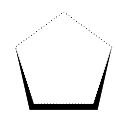
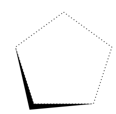
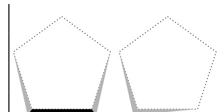
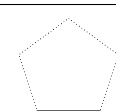
Patch type	Ribbon	Parameterization	Blending function
Generalized C^0 Coons [9]	trilat.	full	
Composite ribbon [13]		$d \in [0, 1]$	
Midpoint [12]	bilat.	full	
Charrot–Gregory [1]		simple	
Generalized Coons [13]	unilat.	constrained	
Midpoint Coons [10]		constrained, full	
Katō [6]		simple	

Table 1: Overview of different generalizations of the Coons patch.

The authors prefer the first three in the list. The first one only interpolates the boundary curves, and no derivatives, but it is a very simple scheme that works well in practice. The second one is very good when the number of sides is relatively low; otherwise some control of the interior is needed, and that is where we prefer the midpoint patch. Note also that the Generalized Coons and Midpoint Coons patches are very similar to the Charrot–Gregory and Midpoint patches, respectively [11], but they need constrained parameterization.

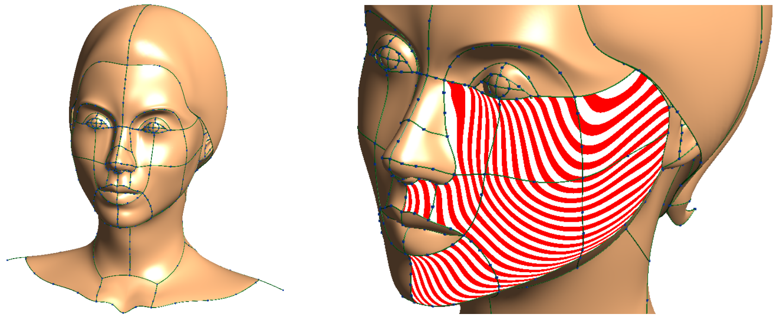


Fig. 17: A complex model showing contouring.

Acknowledgements

Most of the related research was conducted with Tamás Várady, to whom I am grateful for the many conversations which helped forming my view on the subject.

References

1. Peter Charrot and John A. Gregory. A pentagonal surface patch for computer aided geometric design. *Computer Aided Geometric Design*, 1(1):87–94, 1984.
2. Steven Anson Coons. Surfaces for computer aided design. Technical report, MIT, 1964.
3. Gerald Farin. *Curves and surfaces for CAGD: A practical guide*. Morgan Kaufmann, 5th edition, 2002.
4. Gerald Farin and Dianne Hansford. Discrete Coons patches. *Computer Aided Geometric Design*, 16(7):691–700, 1999.
5. Kai Hormann and Michael S. Floater. Mean value coordinates for arbitrary planar polygons. *Transactions on Graphics*, 25(4):1424–1441, 2006.
6. Kiyokata Katō. Generation of n-sided surface patches with holes. *Computer-Aided Design*, 23(10):676–683, 1991.
7. Charles T. Loop and Tony D. DeRose. A multisided generalization of Bézier surfaces. *ACM Transactions on Graphics*, 8(3):204–234, 1989.
8. Péter Salvi. G^1 hole filling with S-patches made easy. In *Proceedings of the 12th Conference of the Hungarian Association for Image Processing and Pattern Recognition*. KÉPAF, 2019.
9. Péter Salvi. A multi-sided generalization of the C^0 Coons patch. In *Proceedings of the Workshop on the Advances of Information Technology*, pages 110–111. BME, 2020.
10. Péter Salvi, István Kovács, and Tamás Várady. Computationally efficient transfinite patches with fullness control. In *Proceedings of the Workshop on the Advances of Information Technology*, pages 96–100. BME, 2017.
11. Péter Salvi and Tamás Várady. Comparison of two n-patch representations in curve network-based design. In *Proceedings of the 10th Conference of the Hungarian Association for Image Processing and Pattern Recognition*, pages 612–624. KÉPAF, 2015.

12. Péter Salvi and Tamás Várady. Multi-sided surfaces with fullness control. In *Proceedings of the Eighth Hungarian Conference on Computer Graphics and Geometry*, pages 61–69. NJSZT, 2016.
13. Péter Salvi, Tamás Várady, and Alyn Rockwood. Ribbon-based transfinite surfaces. *Computer Aided Geometric Design*, 31(9):613–630, 2014.
14. Tamás Várady, Péter Salvi, and György Karikó. A multi-sided Bézier patch with a simple control structure. *Computer Graphics Forum*, 35(2):307–317, 2016.