An Introduction to Polar Forms

Hans-Peter Seidel *

Universität Erlangen, IMMD IX - Graphische Datenverarbeitung
Am Weichselgarten 9, D-8520 Erlangen

Abstract

Polar forms simplify the construction of polynomial and piecewise polynomial curves and surfaces and lead to new surface representations and algorithms. This paper provides an introduction to polar forms and shows how polar forms yield closed form solutions to various recursive algorithms that are used in Computer Aided Geometric Design. As a consequence, we obtain a simple new labeling scheme for Bézier and B-spline curves and surfaces that allows us to label control points in a consistent and meaningful way. The presentation concludes with a survey of some recent new results that were obtained using polar forms.

1 Introduction

The main idea behind polar forms is best explained by a picture: Fig.1 shows a cubic Bézier curve F over the unit interval $[0, 1]$ together with its Bézier points and all the intermediate points that come up during evaluation of the curve at a parameter t using the *de Casteljau Algorithm*. New in this figure is the labeling scheme. While most standard texts use labels such as b_j^l for the intermediate points of the de Casteljau Algorithm the labels in Fig.1 are of the form $f(\cdot, \cdot, \cdot)$ where f is the *polar form* of the polynomial F. Since F is of degree three, its polar form has three arguments. Furthermore, the polar form is *symmetric*, i.e., its three arguments can be written in any order without changing the value of f, and f is related to F by the identity $F(u) = f(u, u, u)$. Finally, the incidence structure of the points and lines in Fig.1 is reflected in the labels: All points whose labels share at least two arguments lie on the same line. The exact position of a point on this line is determined by the remaining third label: As t moves with constant speed between 0 and 1, the point $f(0, t, 1)$, e.g., moves with constant speed between $f(0, 0, 1)$ and $f(0, 1, 1)$. The point $f(0, t, 1)$ lies t of the way from $f(0, 0, 1)$ to $f(0, 1, 1)$. Moving on a line with constant speed means that the polar form is *affine* in each argument, or

simply *multiaffine.* Thus the polar form f of a cubic polynomial curve F is a symmetric triaffine map that satisfies $F(u) = f(u, u, u)$.
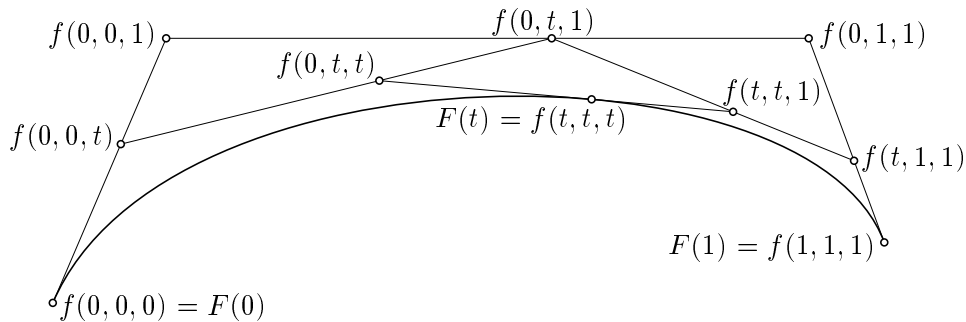

Figure 1: A cubic Bézier curve and the de Casteljau Algorithm.

Note that the above properties allow to reconstruct the point $F(t)$ from the Bézier points $f(0, 0, 0)$, $f(0, 0, 1)$, $f(0, 1, 1)$, and $f(1, 1, 1)$ as follows: First we interpolate linearly along the edges of the control polygon to obtain the points $f(0, 0, t)$, $f(0, t, 1)$, and $f(t, 1, 1)$. Then we interpolate linearly between these points to obtain $f(0, t, t)$ and $f(t, t, 1)$. Finally, the last step of interpolation between these two points yields the point $F(t) = f(t, t, t)$ on the curve. This is exactly the de Casteljau Algorithm.

2   The polar form of a polynomial curve

We now wish to generalize our introductory discussion from cubics to polynomial curves of arbitrary degree. In particular, we wish to establish the so-called *Blossoming Principle* [5, 4, 11, 12, 13] which essentially states that every polynomial has a unique polar form.

We first need a little bit of notation. Recall that a map $f : \mathbb{R} \to \mathbb{R}^t$ is *affine* if it preserves affine combinations, i.e., if f satisfies $f(\sum_j \alpha_j u_j) = \sum_j \alpha_j f(u_j)$ for all scalars $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$ with $\sum_j \alpha_j = 1$. A map $f : \mathbb{R}^n \to \mathbb{R}^t$ is *n-affine* (or just *multiaffine*) if it is an affine map in each argument when the others are held fixed. Thus f is multiaffine if

$$f(u_1, \ldots, \sum_j \alpha_j u_{i_j}, \ldots, u_n) = \sum_j \alpha_j f(u_1, \ldots, u_{i_j}, \ldots, u_n)$$

for all $i = 1, \ldots, n$ and $\alpha_1, \ldots, \alpha_m \in \mathbb{R}$ with $\sum_j \alpha_j = 1$. Finally, $f : \mathbb{R}^n \to \mathbb{R}^t$ is called *symmetric* if it keeps its values under any permutation of its arguments.

It is possible to extend the domain of a symmetric multiaffine map f to vectors: Let $\hat{\xi}_i = w_i - v_i$ be a vector. We can then define $f(u_1, \ldots, u_{n-q}, \hat{\xi}_1, \ldots, \hat{\xi}_q)$ recursively as

$$f(u_1, \ldots, u_{n-q}, \hat{\xi}_1, \ldots, \hat{\xi}_q) =$$
$$f(u_1, \ldots, u_{n-q}, w_1, \hat{\xi}_2 \ldots, \hat{\xi}_q) - f(u_1, \ldots, u_{n-q}, v_1, \hat{\xi}_2 \ldots, \hat{\xi}_q).$$

Note that this definition is in fact well-defined, i.e., only depends on the vectors $\hat{\xi}_i$, not on their starting nor end points w_i and v_i.

With this notation in place we are then able to state the following *Blossoming Principle* [11, 12, 13]:

Theorem 2.1 (Blossoming Principle) *Polynomials $F : \mathbb{R} \to \mathbb{R}^t$ of degree n and symmetric multiaffine maps $f : (\mathbb{R})^n \to \mathbb{R}^t$ are equivalent to each other. In particular, given a map of either type, a unique map of the other type exists that satisfies the identity $F(u) = f(u, \ldots, u)$. In this situation f is called the multiaffine polar form or blossom of F, while F is called the diagonal of f. Furthermore, the q-th derivative of F is given as*

$$F^{(q)}(u) = \frac{n!}{(n-q)!} f(\underbrace{u, \ldots, u}_{n-q}, \underbrace{\hat{1}, \ldots, \hat{1}}_{q}), \tag{2.1}$$

where $\hat{1} = 1 - 0 \in \mathbb{R}$ is the standard unit vector and $f(u, \ldots, u, \hat{1}, \ldots, \hat{1})$ is defined as above. ♣

Explicit formulas for the polar form f become particularly simple for monomials

$$F(u) = \sum_{i=0}^{n} \mathbf{a}_i u^i.$$

In this case the polar form f is given by the formula

$$f(u_1, \ldots, u_n) = \sum_{i=0}^{n} \mathbf{a}_i \binom{n}{i}^{-1} \sum_{\substack{S \subseteq \{1, \ldots, n\} \\ |S| = i}} \prod_{j \in S} u_j,$$

and the coefficients $\mathbf{a}_i$ satisfy

$$\mathbf{a}_i = \frac{F^{(q)}(0)}{q!} = \binom{n}{q} f(\underbrace{0, \ldots, 0}_{n-q}, \underbrace{\hat{1}, \ldots, \hat{1}}_{q}).$$

For the cubic polynomial

$$F(u) = \mathbf{a}_0 + \mathbf{a}_1 u + \mathbf{a}_2 u^2 + \mathbf{a}_3 u^3$$

we obtain, e.g.,

$$f(u_1, u_2, u_3) = \mathbf{a}_0 + \frac{\mathbf{a}_1}{3}(u_1 + u_2 + u_3) + \frac{\mathbf{a}_2}{3}(u_1 u_2 + u_2 u_3 + u_3 u_1) + \mathbf{a}_3 u_1 u_2 u_3.$$

A coordinate-free formula for the polar form is [11]

$$f(u_1, \ldots, u_n) = \frac{1}{n!} \sum_{\substack{S \subseteq \{1, \ldots, n\} \\ |S| = i}} (-1)^{n-i} i^n F(\frac{1}{i} \sum_{j \in S} u_j).$$

We conclude this section with a discussion of the continuity conditions between two polynomials in terms of polar forms. Essentially by rephrasing (2.1) we obtain the following theorem:

3

Theorem 2.2 (C^q-Conditions) *Let $F : \mathbb{R} \to \mathbb{R}^t$ and $G : \mathbb{R} \to \mathbb{R}^t$ be two polynomials of degree n, and let $u \in \mathbb{R}$. Then the following two statements are equivalent:*

- *F and G are C^q-continuous at u.*

- *$f(u, \ldots, u, u_1, \ldots, u_q) = g(u, \ldots, u, u_1, \ldots, u_q)$ for $u_1, \ldots, u_q \in \mathbb{R}$* ♣.

3 Bézier curves

Why are polar forms useful? Let us consider a polynomial curve $F : \mathbb{R} \to \mathbb{R}^t$. Suppose we wish to represent F as a Bézier curve over some given interval $\Delta = [r, s]$. What are the Bézier points? Writing u as an affine combination of r and s,

$$u = \frac{s - u}{s - r} r + \frac{u - r}{s - r} s,$$

we obtain

$$
\begin{aligned}
F(u) &= f(u, \ldots, u) = \frac{s - u}{s - r} \, f(u, \ldots, u, r) + \frac{u - r}{s - r} \, f(u, \ldots, u, s) \\
&= \left(\frac{s - u}{s - r} \right)^2 f(u, \ldots, u, r, r) + 2 \left(\frac{u - r}{s - r} \right) \left(\frac{s - u}{s - r} \right) f(u, \ldots, u, r, s) \\
&\quad + \left(\frac{u - r}{s - r} \right)^2 f(u, \ldots, u, s, s) \\
&= \sum_{j=0}^{n} B_j^{\Delta, n}(u) \, f(\underbrace{r, \ldots, r}_{n-j}, \underbrace{s, \ldots, s}_{j}),
\end{aligned}
$$

where

$$B_j^{\Delta, n}(u) = \binom{n}{j} \left(\frac{u - r}{s - r} \right)^j \left(\frac{s - u}{s - r} \right)^{n-j}, \quad j = 0, \ldots, n,$$

are the Bernstein polynomials w.r.t. $\Delta = [r, s]$. Thus we have [5, 4, 11, 12]:

Theorem 3.1 (Bézier Points) *Let $\Delta = [r, s]$ be an arbitrary interval. Every polynomial $F : \mathbb{R} \to \mathbb{R}^t$ can be represented as a Bézier polynomial w.r.t. Δ. The Bézier points are given as*

$$\mathbf{b}_j = f(\underbrace{r, \ldots, r}_{n-j}, \underbrace{s, \ldots, s}_{j}), \tag{3.1}$$

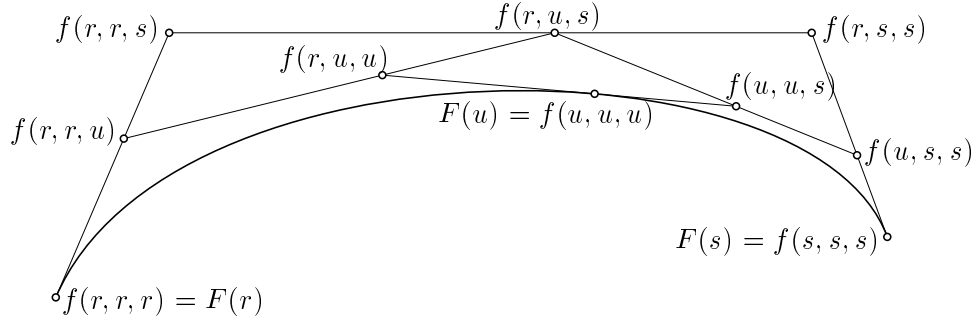*where f is the polar form of F.* ♣

4

Figure 2: The de Casteljau Algorithm in the case $n = 3$

Equation (3.1) immediately leads to an evaluation algorithm that recursively computes the values

$$
\begin{aligned}
\mathbf{b}_j^l(u) &= f(\underbrace{r, \ldots, r}_{n-l-j}, \underbrace{u, \ldots, u}_{l}, \underbrace{s, \ldots, s}_{j}) \\
&= \frac{s-u}{s-r} f(\underbrace{r, \ldots, r}_{n-l-j+1}, \underbrace{u, \ldots, u}_{l-1}, \underbrace{s, \ldots, s}_{j}) + \frac{u-r}{s-r} f(\underbrace{r, \ldots, r}_{n-l-j}, \underbrace{u, \ldots, u}_{l-1}, \underbrace{s, \ldots, s}_{j+1}) \\
&= \frac{s-u}{s-r} \mathbf{b}_j^{l-1}(u) + \frac{u-r}{s-r} \mathbf{b}_{j+1}^{l-1}(u),
\end{aligned}
$$

from the given control points. For $l = n$ we finally compute $\mathbf{b}_0^n(u) = f(u, \ldots, u) = F(u)$ which is the desired point on the curve. The resulting computational scheme is illustrated by Fig.2 and Fig.3. This algorithm was first studied by Paul de Faget de Casteljau [4, 5] and is therefore called *de Casteljau Algorithm*.

Formula (3.1) also shows that the de Casteljau Algorithm offers much more than just evaluation: Suppose that we wish to subdivide a Bézier curve F over a given interval $\Delta = [s, t]$ at an arbitrary parameter $u \in \Delta$. What are the new Bézier points of the left and right segments $\mathbf{F}_l$ and $\mathbf{F}_r$ with respect to the subintervals $\Delta_l = [r, u]$ and $\Delta_r = [u, s]$? Equation (3.1) tells us that the new Bézier points after subdivision are given as

$$
\mathbf{b}_0^l = f(r, \ldots, r), \quad \mathbf{b}_1^l = f(r, \ldots, r, u), \quad \ldots, \quad \mathbf{b}_n^l = f(u, \ldots, u)
$$

and

$$
\mathbf{b}_0^r = f(u, \ldots, u), \quad \mathbf{b}_1^r = f(u, \ldots, u, s), \quad \ldots, \quad \mathbf{b}_n^r = f(s, \ldots, s). \tag{3.2}
$$

Inspection shows that these points are automatically computed during the de Casteljau Algorithm and are stored along the left and right diagonals.

Finally, a slight modification of the de Casteljau Algorithm can be used to compute arbitrary polar values $f(u_1, \ldots, u_n)$ by recursively computing the values
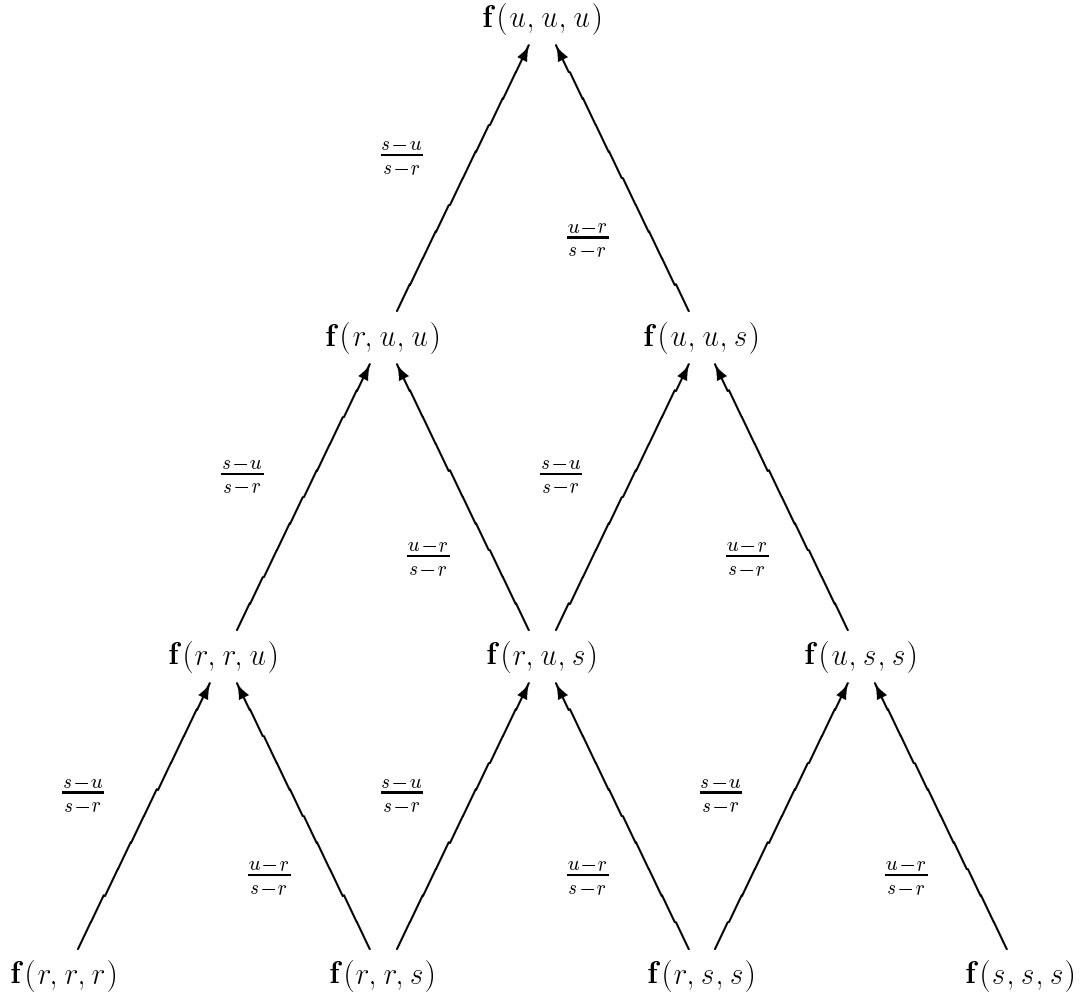
5

Figure 3: The de Casteljau Algorithm for the case $n = 3$.

$$
\begin{aligned}
\mathbf{b}_j^l(u_1, \ldots, u_l) &= f(\underbrace{r, \ldots, r}_{n-l-j}, \underbrace{u_1, \ldots, u_l}_{l}, \underbrace{s, \ldots, s}_{j}) \\
&= \frac{s - u_l}{s - r} \, f(\underbrace{r, \ldots, r}_{n-l-j+1}, \underbrace{u_1, \ldots, u_{l-1}}_{l-1}, \underbrace{s, \ldots, s}_{j}) \\
&\quad + \frac{u_l - r}{s - r} \, f(\underbrace{r, \ldots, r}_{n-l-j}, \underbrace{u_1, \ldots, u_{l-1}}_{l-1}, \underbrace{s, \ldots, s}_{j+1}) \\
&= \frac{s - u_l}{s - r} \, \mathbf{b}_j^{l-1}(u_1, \ldots, u_{l-1}) + \frac{u_l - r}{s - r} \, \mathbf{b}_{j+1}^{l-1}(u_1, \ldots, u_{l-1}).
\end{aligned}
$$

For $l = n$ we finally compute $\mathbf{b}_0^n(u_1, \ldots, u_n) = f(u_1, \ldots, u_n)$. This algorithm is called *multiaffine de Casteljau Algorithm*.

How about derivatives? After writing $\hat{1} = \frac{1}{s-r}(s-r)$, Formula (2.1) implies

$$F'(r) \;\; = \;\; \frac{n}{s-r}(f(r,\ldots,r,s) - f(r,\ldots,r)) = \frac{n}{s-r}(\mathbf{b}_1 - \mathbf{b}_0)$$

and similarly

$$F''(r) = \frac{n(n-1)}{(s-r)^2}(\mathbf{b}_2 - 2\mathbf{b}_1 + \mathbf{b}_0), \;\; \text{etc.}$$

These are the well-known derivative formulas for Bézier curves.

We conclude this section with a brief remark on affine invariance. Let $\tilde{F} = \phi \circ F$ be the image of F under an affine map (e.g., translation, scaling, rotation) ϕ. The uniqueness part of Theorem 2.1 implies that the polar form $\tilde{f}$ of $\tilde{F}$ is given as $\tilde{f} = \phi \circ f$, and it follows that the Bézier points $\tilde{\mathbf{b}}_j$ of $\tilde{F}$ satisfy

$$\tilde{\mathbf{b}}_j = \tilde{f}(r,\ldots,r,s,\ldots,s) = \phi(f(r,\ldots,r,s,\ldots,s)) = \phi(\mathbf{b}_j).$$

This means that the relationship between the curve F and its Bézier control polygon is invariant under affine maps.

4   B-Spline Curves

How can we extend the results of the preceding section from Bézier curves to B-splines? We have seen that a polynomial curve F and its polar form f is completely defined by its Bézier points $\mathbf{b}_j = f(r,\ldots,r,s,\ldots,s)$. Let

$$r_n \leq \ldots \leq r_1 < s_1 \leq \ldots \leq s_n$$

be a non-decreasing sequence of real numbers. We wish to show that F can equally well be defined by its *de Boor points* $\mathbf{d}_j = f(r_1,\ldots,r_{n-j},s_1,\ldots,s_j)$. Since $r_i \neq s_j$, we can express u as an affine combination w.r.t. r_i and s_j,

$$u = \frac{s_j - u}{s_j - r_i}\, r_i + \frac{u - r_i}{s_j - r_i}\, s_j,$$

and by successively expanding

$$
\begin{aligned}
\mathbf{d}_j^l(u) \;\; &= \;\; f(r_1,\ldots,r_{n-l-j}, u,\ldots,u, s_1,\ldots,s_j) \\
&= \;\; \frac{s_{j+1} - u}{s_{j+1} - r_{n-l-j+1}}\, f(r_1,\ldots,r_{n-l-j+1}, u,\ldots,u, s_1,\ldots,s_j) \\
&\quad + \frac{u - r_{n-l-j+1}}{s_{j+1} - r_{n-l-j+1}}\, f(r_1,\ldots,r_{n-l-j}, u,\ldots,u, s_1,\ldots,s_{j+1}) \\
&= \;\; \frac{s_{j+1} - u}{s_{j+1} - r_{n-l-j+1}}\, \mathbf{d}_j^{l-1}(u) \;+\; \frac{u - r_{n-l-j+1}}{s_{j+1} - r_{n-l-j+1}}\, \mathbf{d}_{j+1}^{l-1}(u)
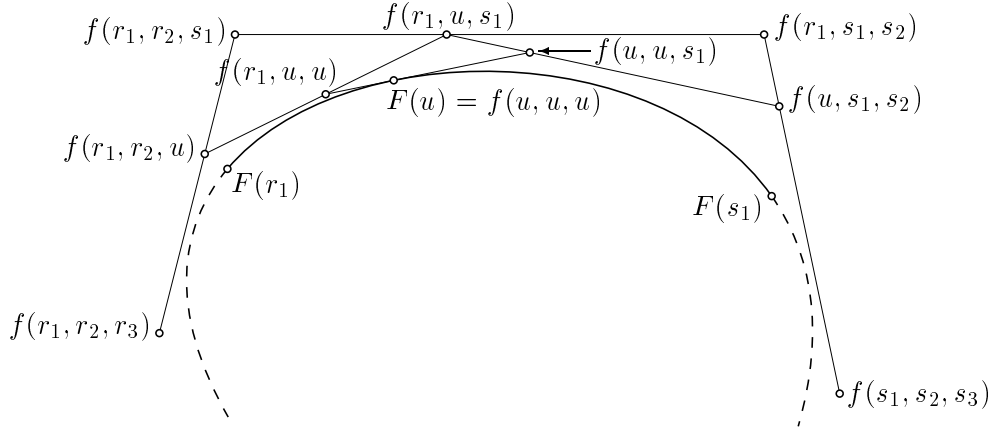\end{aligned}
$$

Figure 4: The de Boor Algorithm for the case $n = 3$.

we see that $F(u) = \mathbf{d}_0^n(u)$ is in fact completely determined by the points $\mathbf{d}_j = f(r_1, \ldots, s_j)$.

Conversely, suppose that the points $\mathbf{d}_j = f(r_1, \ldots, r_{n-j}, s_1, \ldots, s_j)$ are given. We can then use the above recurrence to evaluate the curve F at an arbitrary parameter value u. Inspection shows that the resulting algorithm is identical to the *de Boor Algorithm* for the evaluation of a B-spline segment from its end points. We thus have the following [12, 13, 16]:

Theorem 4.1 (de Boor points) *Every polynomial $F : \mathbb{R} \to \mathbb{R}^t$ can be represented as B-spline segment over a non-decreasing knot sequence $r_n \le \ldots \le r_1 < s_1 \le \ldots \le s_n$. The de Boor points are given as*

$$\mathbf{d}_j = f(r_1, \ldots, r_{n-j}, s_1, \ldots, s_j), \tag{4.1}$$

where f is the polar form of F. ♣

Theorem 4.1 and the de Boor Algorithm are illustrated by Fig.4 and Fig.5. Again, the multiaffine version of the algorithm can be used to compute arbitrary polar values $f(u_1, \ldots, u_n)$ from the given control points.

Even more important than evaluation is *knot insertion*. Suppose that the knot sequence $r_n \le \ldots \le r_1 < s_1 \le \ldots \le s_n$ is given and that we wish to insert a new knot t with $r_1 \le t < s_1$. Equation (4.1) tells us that the new control points $\mathbf{d}_j^*$ after knot insertion are given as

$$
\begin{aligned}
\mathbf{d}_j^* &= f(t, r_1, \ldots, r_{n-j-1}, s_1, \ldots, s_j) \\
&= \frac{s_{j+1} - t}{s_{j+1} - r_{n-j}} \, f(r_1, \ldots, r_{n-j}, s_1, \ldots, s_j) \\
&\quad + \frac{t - r_{n-j}}{s_{j+1} - r_{n-j}} \, f(r_1, \ldots, r_{n-j-1}, s_1, \ldots, s_{j+1}) \\
&= \frac{s_{j+1} - t}{s_{j+1} - r_{n-j}} \, \mathbf{d}_j \; + \; \frac{t - r_{n-j}}{s_{j+1} - r_{n-j}} \, \mathbf{d}_{j+1}.
\end{aligned}
$$

8

$$\mathbf{f}(u,u,u)$$

$$\tfrac{s_1-u}{s_1-r_1} \qquad \tfrac{u-r_1}{s_1-r_1}$$

$$\mathbf{f}(r_1,u,u) \qquad \mathbf{f}(u,u,s_1)$$

$$\tfrac{s_1-u}{s_1-r_2} \qquad \tfrac{s_2-u}{s_2-r_1}$$

$$\tfrac{u-r_2}{s_1-r_2} \qquad \tfrac{u-r_1}{s_2-r_1}$$

$$\mathbf{f}(r_1,r_2,u) \qquad \mathbf{f}(r_1,u,s_1) \qquad \mathbf{f}(u,s_1,s_2)$$

$$\tfrac{s_1-u}{s_1-r_3} \qquad \tfrac{s_2-u}{s_2-r_2} \qquad \tfrac{s_3-u}{s_3-r_1}$$

$$\tfrac{u-r_3}{s_1-r_3} \qquad \tfrac{u-r_2}{s_2-r_2} \qquad \tfrac{u-r_1}{s_3-r_1}$$

$$\mathbf{f}(r_1,r_2,r_3) \qquad \mathbf{f}(r_1,r_2,s_1) \qquad \mathbf{f}(r_1,s_1,s_2) \qquad \mathbf{f}(s_1,s_2,s_3)$$
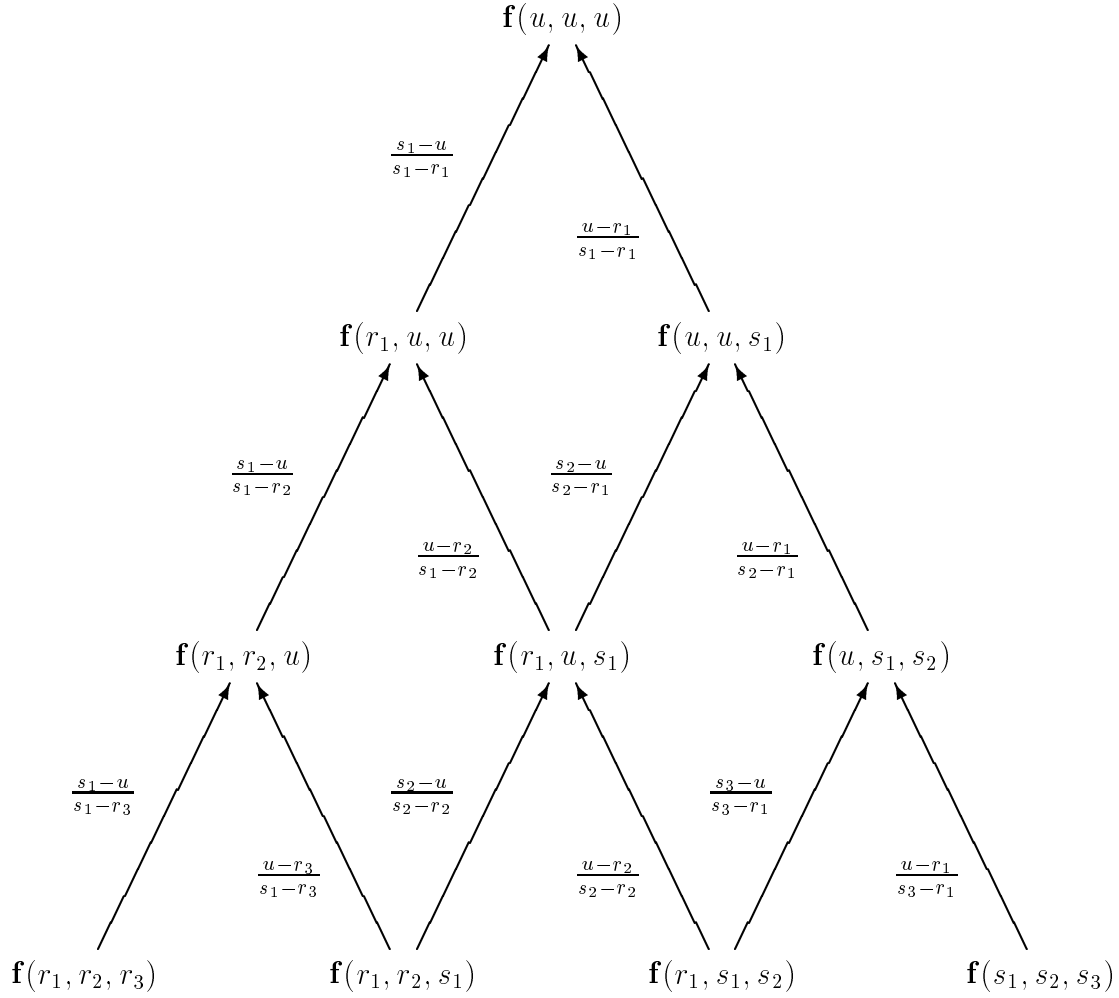
Figure 5: The de Boor Algorithm for a cubic B-spline segment.

This is exactly the *Boehm Algorithm*. Note that the Boehm Algorithm is identical to the first step of the de Boor Algorithm. Other knot insertion algorithms, as, e.g., the OSLO Algorithm, have been studied in great detail in a sequence of papers by P.J. Barry and R.N. Goldman [1, 9]. A thorough treatment of this material can be found in [2].

We conclude this section with a brief discussion of the miracle that B-spline curves are C^{n-q}-continuous at a knot of multiplicity q. In order to keep our discussion as simple as possible we only consider the case where $\{t_i\}_i$ is a sequence of simple knots. Let $F_i : [t_i, t_{i+1}] \to \mathbb{R}^t$ and $F_{i+1} : [t_{i+1}, t_{i+2}] \to \mathbb{R}^t$ be two adjacent B-spline segments that join at the knot t_{i+1}. Since $f_i(t_{i-n+j+1}, \ldots, t_{i+j}) = f_{i+1}(t_{i-n+j+1}, \ldots, t_{i+j})$, for $j = 1, \ldots, n$, successive expansion shows that

$$f_i(t_{i+1}, u, \ldots, u) = f_{i+1}(t_{i+1}, u, \ldots, u).$$

Then (2.1) implies that F_i and F_{i+1} are in fact C^{n-1}-continuous at t_{i+1}. The overlapping de Boor schemes of two adjacent cubic B-spline segments are shown in Fig.6.
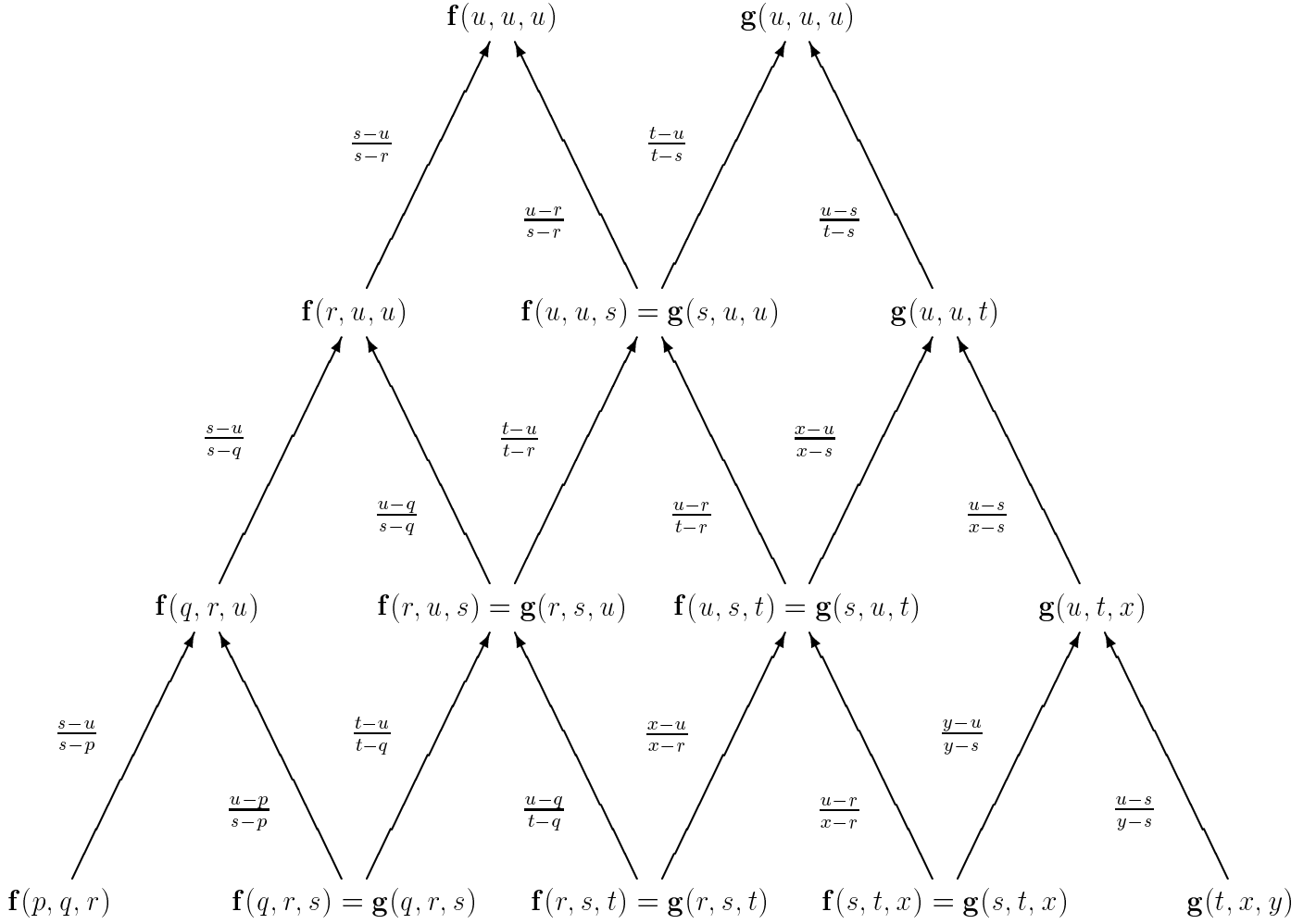

Figure 6: Overlapping de Boor schemes for two adjacent cubic B-spline segments $F : [r, s] \to \mathbb{R}^t$ and $G : [s, t] \to \mathbb{R}^t$ over the knot sequence $\dots, p, q, r, s, t, x, y, \dots$

5 Tensor product surfaces

By far the most popular surfaces in Computer Aided Geometric Design and computer graphics are tensor product surfaces: Given a curve scheme $F(u) = \sum_{i=0}^{n} B_i(u)\mathbf{b}_i$, $\mathbf{b}_i \in \mathbb{R}^t$, the corresponding tensor product scheme is defined as

$$F(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i(u) \, B_j(v) \, \mathbf{b}_{ij}, \quad \mathbf{b}_{ij} \in \mathbb{R}^t,$$

which can also be written as

$$F(u,v) = \sum_{i=0}^{n} B_i(u) \, \mathbf{b}_{i_v} \ \text{ with } \mathbf{b}_{i_v} = \mathbf{b}_i(v) = \sum_{j=0}^{m} B_j(v)\mathbf{b}_{ij}.$$

The last equation demonstrates that tensor product surfaces may be considered as curves of curves, and thus explains that we first have to understand curves in order to understand tensor product surfaces.

Let us now consider the polar form of a polynomial tensor product surface. Let

$$F : \mathbb{R} \times \mathbb{R} \to \mathbb{R} : (u,v) \mapsto F(u,v)$$

be a polynomial tensor product surface of degree n in u and of degree m in v. In order to compute the corresponding polar form f_{TP} of F we simply have to polarize both independent variables u and v separately. The resulting map

$$f_{TP} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^t : (u_1, \ldots, u_n; v_1, \ldots, v_m) \mapsto f_{TP}(u_1, \ldots, u_n; v_1, \ldots, v_m)$$

is then characterized by the following properties:

- Symmetry: f_{TP} is symmetric in the variables u_i and v_j separately, i.e., we get

$$f_{TP}(u_1, \ldots, u_n; v_1, \ldots, v_m) = f_{TP}(u_{\pi(1)}, \ldots, u_{\pi(n)}; v_{\sigma(1)}, \ldots, v_{\sigma(m)})$$

  for all permutations $\pi \in \Sigma_n$ and $\sigma \in \Sigma_m$.

- Multiaffine Property: f_{TP} is affine in each of the variables u_i and v_j separately.

- Diagonal Property: $f_{TP}(u, \ldots, u; v, \ldots, v) = F(u,v)$.

In generalization of the curve case, the Bézier points $\mathbf{b}_{ij}$ of F in the representation $F(u,v) = \sum_{i=0}^{n} B_i^n(u)B_j^m(v)\mathbf{b}_{ij}$ as a tensor product Bézier surface over $[p,q] \times [r,s]$ are given as

$$\mathbf{b}_{ij} = f_{TP}(\underbrace{p, \ldots, p}_{n-i}, \underbrace{q, \ldots, q}_{i}; \underbrace{r, \ldots, r}_{m-j}, \underbrace{s, \ldots, s}_{j})$$

while the de Boor points $\mathbf{d}_{ij}$ of F in the representation $F(u,v) = \sum_i \sum_j N_i^n(u)N_j^m(v)\mathbf{d}_{ij}$ as segment of a tensor product B-spline surface over the knot vectors $S = \{s_i\}$ and $T = \{t_j\}$ are given as

$$\mathbf{d}_{ij} = f_{TP}(s_{i+1}, \ldots, s_{i+n}; t_{j+1}, \ldots, t_{j+m}).$$

Many algorithms that have been discussed in the previous sections can then be generalized from Bézier and B-spline curves to Bézier and B-spline tensor product surfaces.

6 The polar form of a polynomial surface

From now on we wish to discuss 'true' surfaces. We start with the *Blossoming Principle* which generalizes almost word-by-word from curves to surfaces:

Theorem 6.1 (Blossoming Principle) *Polynomials $F : \mathbb{R}^2 \to \mathbb{R}^t$ of degree n and symmetric multiaffine maps $f : (\mathbb{R}^2)^n \to \mathbb{R}^t$ are equivalent to each other. In particular, given a map of either type, a unique map of the other type exists that satisfies the identity $F(\mathbf{u}) = f(\mathbf{u}, \ldots, \mathbf{u})$. In this situation f is called the multiaffine polar form or blossom of F, while F is called the diagonal of f. Furthermore, the q-th directional derivative of F with respect to vectors $\hat{\xi}_1, \ldots, \hat{\xi}_q \in \mathbb{R}^2$ is given as*

$$D_{\hat{\xi}_1, \ldots, \hat{\xi}_q} F(\mathbf{u}) = \frac{n!}{(n-q)!} f(\mathbf{u}, \ldots, \mathbf{u}, \hat{\xi}_1, \ldots, \hat{\xi}_q). \tag{6.1}$$

where $f(\mathbf{u}, \ldots, \mathbf{u}, \hat{\xi}_1, \ldots, \hat{\xi}_q)$ is defined as in Section 2. ♣

Again, things are particularly simple for monomials. For the quadratic polynomial

$$F(\mathbf{u}) = \mathbf{a}_{00} + \mathbf{a}_{10} \, u + \mathbf{a}_{01} \, v + + \mathbf{a}_{20} \, u^2 + \mathbf{a}_{11} \, uv + \mathbf{a}_{02} \, v^2$$

we obtain, e.g.,

$$f(\mathbf{u}_1, \mathbf{u}_2) = \mathbf{a}_{00} + \frac{\mathbf{a}_{10}}{2} \, (u_1 + u_2) + \frac{\mathbf{a}_{01}}{2} \, (v_1 + v_2) + \mathbf{a}_{20} \, u_1 u_2 + \frac{\mathbf{a}_{11}}{2} \, (u_1 v_2 + u_2 v_1) + \mathbf{a}_{02} \, v_1 v_2$$

The coordinate-free formula for the polar form becomes [11]

$$f(\mathbf{u}_1, \ldots, \mathbf{u}_n) = \frac{1}{n!} \sum_{\substack{S \subseteq \{1, \ldots, n\} \\ |S| = i}} (-1)^{n-i} i^n F(\frac{1}{i} \sum_{j \in S} \mathbf{u}_j),$$

and the continuity conditions translate into

Theorem 6.2 (C^q-Conditions) *Let $F : \mathbb{R}^2 \to \mathbb{R}^t$ and $G : \mathbb{R}^2 \to \mathbb{R}^t$ be two polynomials of degree n, and let $\mathbf{u} \in \mathbb{R}^2$. Then the following two statements are equivalent:*

- *F and G are C^q-continuous at $\mathbf{u}$.*

- *$f(\mathbf{u}, \ldots, \mathbf{u}, \mathbf{u}_1, \ldots, \mathbf{u}_q) = g(\mathbf{u}, \ldots, \mathbf{u}, \mathbf{u}_1, \ldots, \mathbf{u}_q)$ for $\mathbf{u}_1, \ldots, \mathbf{u}_q \in \mathbb{R}$* ♣*.*
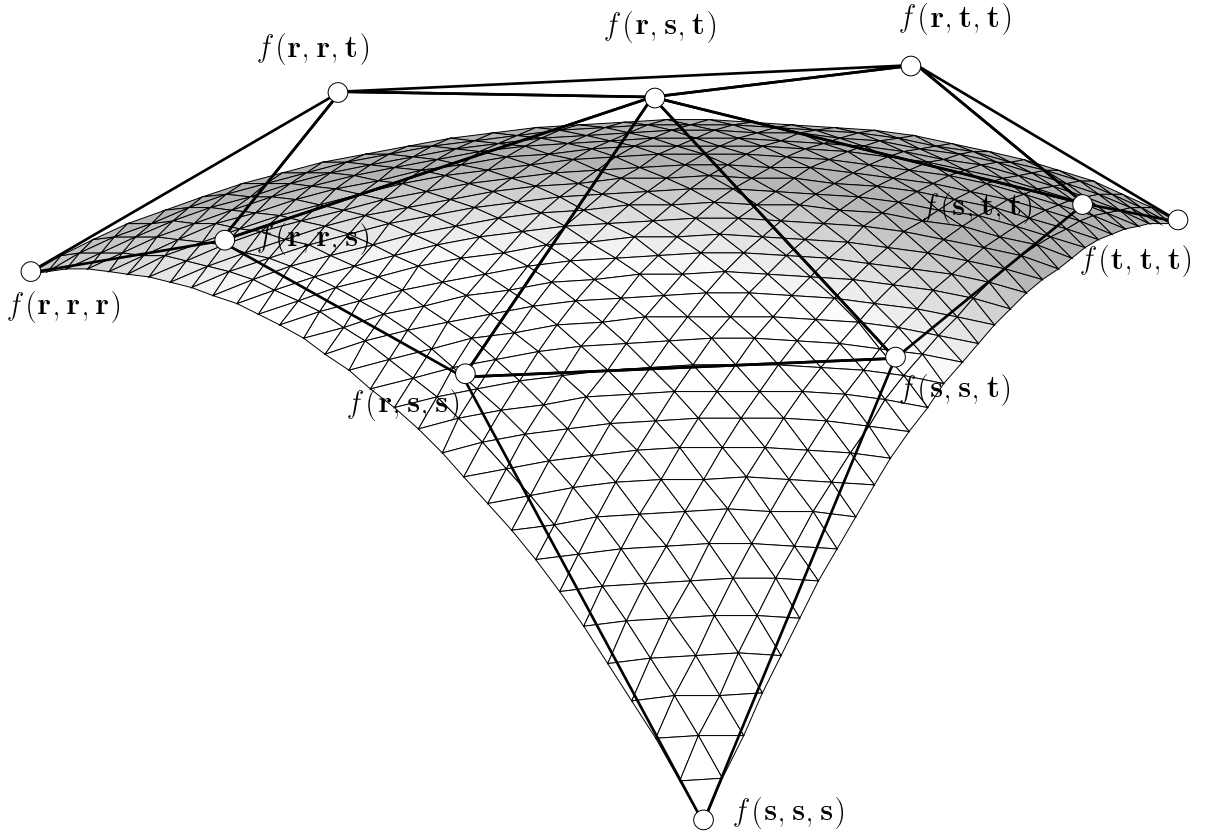
Figure 7: A cubic Bézier patch.

7 Bézier triangles

The straightforward analogue to Bézier curves are triangular Bézier patches. Consider a polynomial surface $F : \mathbb{R}^2 \to \mathbb{R}^t$. Suppose we wish to represent F as a triangular Bézier patch over some given domain triangle $\Delta = \Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$. Representing $\mathbf{u} \in \mathbb{R}^2$ in barycentric coordinates w.r.t. Δ,

$$\mathbf{u} = r(\mathbf{u}) \, \mathbf{r} + s(\mathbf{u}) \, \mathbf{s} + t(\mathbf{u}) \, \mathbf{t}, \quad r + s + t = 1,$$

we obtain

$$
\begin{aligned}
F(\mathbf{u}) &= f(\mathbf{u}, \ldots, \mathbf{u}) \\
&= r(\mathbf{u}) \, f(\mathbf{u}, \ldots, \mathbf{u}, \mathbf{r}) + s(\mathbf{u}) \, f(\mathbf{u}, \ldots, \mathbf{u}, \mathbf{s}) \\
&\quad + t(\mathbf{u}) \, f(\mathbf{u}, \ldots, \mathbf{u}, \mathbf{t}) \\
&= \sum_{i+j+k=n} B_{ijk}^{\Delta,n}(\mathbf{u}) \, f(\underbrace{\mathbf{r}, \ldots, \mathbf{r}}_{i}, \underbrace{\mathbf{s}, \ldots, \mathbf{s}}_{j}, \underbrace{\mathbf{t}, \ldots, \mathbf{t}}_{k})
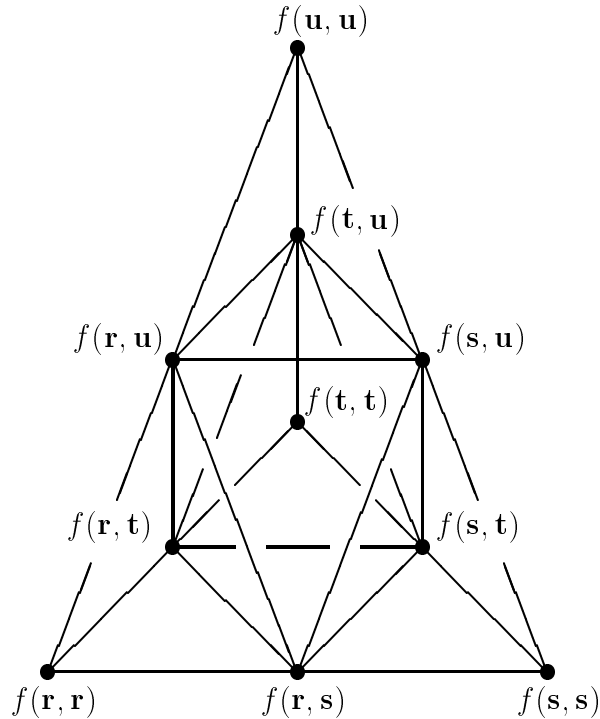\end{aligned}
$$

Figure 8: The de Casteljau Algorithm for a quadratic Bézier patch.

where

$$B_{ijk}^{\Delta,n}(\mathbf{u}) = \binom{n}{ijk} \; r(\mathbf{u})^i \; s(\mathbf{u})^j \; t(\mathbf{u})^k$$

are the Bernstein polynomials w.r.t. $\Delta = \Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$. We have shown:

Theorem 7.1 (Bézier Points) *Let $\Delta = \Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$ be an arbitrary triangle. Every polynomial $F : \mathbb{R}^2 \to \mathbb{R}^t$ can be represented as a Bézier triangle w.r.t. Δ. The Bézier points are given as*

$$\mathbf{b}_{ijk} = f(\underbrace{\mathbf{r}, \dots, \mathbf{r}}_{i}, \underbrace{\mathbf{s}, \dots, \mathbf{s}}_{j}, \underbrace{\mathbf{t}, \dots, \mathbf{t}}_{k}), \qquad (7.1)$$

where f is the polar form of F. ♣

Similar to the curve case, Theorem 7.1 leads directly to the de Casteljau Algorithm for evaluation, subdivision, and computation of the polar form. The resulting computational scheme is illustrated in Fig. 8. The well-known derivative formulas and continuity conditions for Bézier triangles follow directly from (6.1) and (7.1). For $\hat{\xi} = \mathbf{s} - \mathbf{r}$, e.g., we obtain

$$D_{\hat{\xi}} F(\mathbf{r}) = n \; (f(\mathbf{r}, \dots, \mathbf{r}, \mathbf{s}) - \mathbf{f}(\mathbf{r}, \dots, \mathbf{r}, \mathbf{r}) = n \; (\mathbf{b}_{n-1,1,0} - \mathbf{b}_{n,0,0}).$$

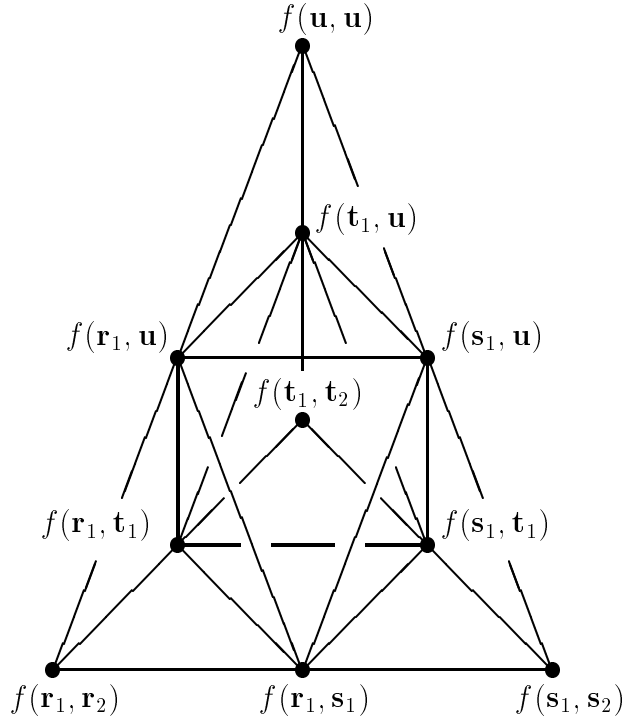Finally, affine invariance also follows in exactly the same way as in the curve case.

Figure 9: The de Boor Algorithm for a quadratic B-patch.

8 B-patches

In the previous section we have seen that a polar form f is uniquely defined by its values $f(\mathbf{r}, \ldots, \mathbf{r}, \mathbf{s}, \ldots, \mathbf{s}, \mathbf{t}, \ldots, \mathbf{t})$ on the vertices of a triangle $\Delta = \Delta(\mathbf{r}, \mathbf{s}, \mathbf{t})$. This definition can be generalized by assigning a family of - usually different - knots to each vertex of the triangle Δ. The resulting surface representation is called a *B-patch* [17, 18].

We say that $\mathcal{A} = \{\mathbf{r}_1, \ldots, \mathbf{r}_n, \mathbf{s}_1, \ldots, \mathbf{s}_n, \mathbf{t}_1, \ldots, \mathbf{t}_n\}$ is a *knot arrangement* if all triangles $\Delta_{ijk} = \Delta(\mathbf{r}_i, \mathbf{s}_j, \mathbf{t}_k)$ are non-degenerate. In this situation we can represent $\mathbf{u}$ in barycentric coordinates w.r.t. $\Delta(\mathbf{r}_i, \mathbf{s}_j, \mathbf{t}_k)$,

$$\mathbf{u} = r_{ijk}(\mathbf{u})\ \mathbf{r}_{i+1} + s_{ijk}(\mathbf{u})\ \mathbf{s}_j + t_{ijk}(\mathbf{u})\ \mathbf{t}_k, \quad r_{ijk} + s_{ijk} + t_{ijk} = 1,$$

and by successively expanding

$$
\begin{aligned}
\mathbf{d}^l_{ijk}(\mathbf{u}) &= f(\mathbf{r}_1, \ldots, \mathbf{r}_i, \mathbf{s}_1, \ldots, \mathbf{s}_j, \mathbf{t}_1, \ldots, \mathbf{t}_k, \mathbf{u}, \ldots, \mathbf{u}) \\
&= r_{i+1,j+1,k+1}(\mathbf{u})\ f(\mathbf{r}_1, \ldots, \mathbf{r}_{i+1}, \mathbf{s}_1, \ldots, \mathbf{s}_j, \mathbf{t}_1, \ldots, \mathbf{t}_k, \mathbf{u}, \ldots, \mathbf{u}) \\
&\quad + s_{i+1,j+1,k+1}(\mathbf{u})\ f(\mathbf{r}_1, \ldots, \mathbf{r}_i, \mathbf{s}_1, \ldots, \mathbf{s}_{j+1}, \mathbf{t}_1, \ldots, \mathbf{t}_k, \mathbf{u}, \ldots, \mathbf{u}) \\
&\quad + t_{i+1,j+1,k+1}(\mathbf{u})\ f(\mathbf{r}_1, \ldots, \mathbf{r}_i, \mathbf{s}_1, \ldots, \mathbf{s}_j, \mathbf{t}_1, \ldots, \mathbf{t}_{k+1}, \mathbf{u}, \ldots, \mathbf{u}),
\end{aligned}
$$

we see that $F(\mathbf{u}) = \mathbf{d}^n_{000}(\mathbf{u})$ is in fact completely determined by the points $\mathbf{d}_{ijk} = f(\mathbf{r}_1, \ldots, \mathbf{t}_k)$. We thus obtain:

15

Theorem 8.1 (B-patch control points) *Let a knot arrangement $\mathcal{A} = \{\mathbf{r}_1, \ldots, \mathbf{t}_n\}$ be given as above. Every polynomial $F : \mathbb{R}^2 \to \mathbb{R}^t$ can be represented as a B-patch over $\mathcal{A}$ with control points*

$$\mathbf{d}_{ijk} = f(\mathbf{r}_1, \ldots, \mathbf{r}_i, \mathbf{s}_1, \ldots, \mathbf{s}_j, \mathbf{t}_1, \ldots, \mathbf{t}_k) \tag{8.1}$$

where f is the polar form of F. ♣

Theorem 8.1 shows that B-patches are the analogue to B-spline curve segments for surfaces. In particular, B-patches have a de Boor like evaluation algorithm that computes a point $F(\mathbf{u})$ on the surface from the given control points through successive linear interpolation. Again, the multiaffine version of this algorithm can be used to compute an arbitrary polar value $f(\mathbf{u}_1, \ldots, \mathbf{u}_n)$. The resulting computational scheme is illustrated by Fig.9.

9   A new multivariate B-spline scheme

By combining B-patches and simplex splines, a new multivariate B-spline scheme has recently been developed in [3]. The new surface scheme is based on blending functions and control points, and allows to construct smooth piecewise polynomial surfaces over arbitrary triangulations of the parameter plane. Due to the given space limitations it is impossible to discuss the surface scheme in glory detail. Some of its main features are summarized in the following theorem:

Theorem 9.1 (Multivariate B-splines) *Let $F(\mathbf{u}) = \sum_I N_{ijk}^I(\mathbf{u}) \mathbf{c}_{ijk}^I$ be a multivariate B-spline surface. Then this surface has the following properties:*

- **Piecewise Polynomial:** *$F(\mathbf{u})$ is a piecewise polynomial of degree n.*

- **Locality:** *Movement of a single control point $\mathbf{c}_{ijk}^I$ only influences the surface on the triangle $\Delta(I)$ and on the triangles directly surrounding $\Delta(I)$.*

- **Convex Hull Property:** *$F(\mathbf{u})$ lies inside the convex hull of its control net.*

- **Smoothness:** *The surface $F(\mathbf{u})$ is generically C^{n-1}-continuous everywhere.*

- **Affine Invariance:** *The relationship between the surface F and its control net is affinely invariant.* ♣

A first implementation has succeeded in demonstrating the practical feasibility of the fundamental algorithms underlying the new surface scheme [8, 18]. Quadratic and cubic surfaces over arbitrary triangulations can be edited and manipulated in real-time.
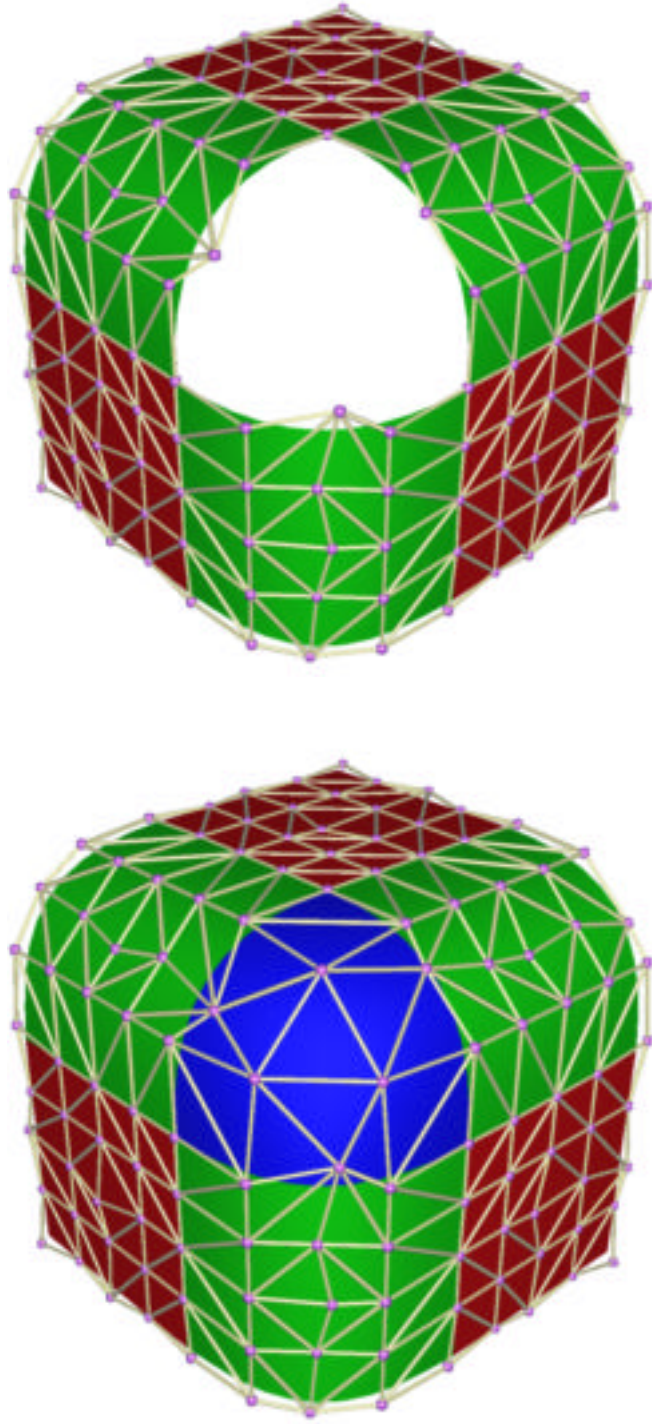
Figure 10: Solving the polygonal hole problem using triangular B-splines: First, the piecewise polynomial surface around the hole is represented as linear combination of B-splines (top). This B-spline surface can then be extended to produce an overall C^{n-1}-continuous fill of the hole (bottom). Note that this method can achieve C^1-continuity with piecewise quadratics, and C^2-continuity with piecewise cubics.

17

In order for a surface scheme to be useful in practice, one must be able to represent as many surfaces as possible by the new scheme. The following theorem [18] is rather remarkable:

Theorem 9.2 (Polynomial and piecewise polynomial representation) *Any polynomial or piecewise polynomial surface F can be represented by the new B-spline scheme. In this situation the control points are obtained as*

$$\mathbf{c}^I_{ijk} = f_i(\mathbf{r}^I_1, \ldots, \mathbf{r}^I_i, \mathbf{s}^I_1, \ldots, \mathbf{s}^I_j, \mathbf{t}^I_1, \ldots, \mathbf{t}^I_k), \qquad (9.1)$$

where f_I is the polar form of the restriction of F to the triangle Δ_I. ♣

Among other applications, Theorem 9.2 allows to derive a solution to the polygonal hole problem, see Figure 10.

10 A few historical remarks

Polar forms are a classical mathematical tool for the study of polynomials. In the context of Computer Graphics and Computer Aided Geometric Design they have first been considered by Paul de Faget de Casteljau at Citroen [4, 5] and by Lyle Ramshaw [11, 12, 13]. The focus of de Casteljau's original work has been on Bézier curves and triangular Bézier patches, and especially on the construction of quasi-interpolants [6]. Ramshaw's treatment is much more algebraic and uses techniques such as homogenizing and tensoring.

More recently, the polar approach to splines has been expanded and applied by various researchers. Seidel [16] applies polar forms directly to the B-spline blending functions and gives a simple development of B-splines from scratch. He also discusses the relationship between polar forms and knot insertion. Barry and Goldman [1, 9] relate polar forms to other B-spline approaches and use polar forms for a thorough discussion of knot insertion for B-splines. Lee [10] and Strom [21] also contribute to this area. Seidel [20] uses the geometry behind polar forms to extend polar forms to geometrically continuous spline curves. Extensions of this geometric approach to surfaces are discussed by Schmeltz [15]. DeRose implements polar forms as an abstract data type and uses them as the basis of a software library [7]. Among other things, he uses polar forms for curvature computations and for composing polynomials.

Polar forms have also been helpful in the development of new surface schemes. Seidel [17] introduces a new surface representation, the B-patch, which may be considered the analogue to a B-spline segment for surfaces. Dahmen, Micchelli, and Seidel [3] combine B-patches with simplex splines and develop the surface scheme of the preceding section. An implementation of this scheme is discussed in [8, 18]. The scheme allows to model smooth piecewise polynomial surfaces over arbitrary triangulations.

References

[1] P.J. Barry and R.N. Goldman. Algorithms for progressive curves: extending B-spline and blossoming techniques to the monomial, power, and newton dual bases. In R.N. Goldman and T. Lyche, editors, *Knot Insertion and Deletion Algorithms for B-Spline Modeling*. SIAM, 1992.

[2] P.J. Barry, R.N. Goldman, L. Ramshaw, and H.-P. Seidel. *Blossoming: The New Polar-Form Approach to Spline Curves and Surfaces, SIGGRAPH '91 Course Notes #26*. ACM SIGGRAPH, 1991.

[3] W. Dahmen, C.A. Micchelli, and H.-P. Seidel. Blossoming begets B-splines built better by B-patches. *Math. Comp.*, 59:97–115, 1992.

[4] P. de Casteljau. Outillages méthodes calcul. Technical report, Andre Citroen, Paris, 1959.

[5] P. de Casteljau. *Formes à Pôles*. Hermes, Paris, 1985.

[6] P. de Casteljau. *Le Lissage*. Hermes, Paris, 1990.

[7] T. DeRose, R.N. Goldman, and M. Lounsbery. A tutorial introduction to blossoming. In H. Hagen and D. Roller, editors, *Geometric Modelling, Methods and Applications*. Springer Verlag, 1991.

[8] P. Fong. Shape control for B-splines over arbitrary triangulations. Master's thesis, University of Waterloo, Waterloo, Canada, 1992.

[9] R.N. Goldman. Blossoming and knot insertion algorithms for B-spline curves. *Computer-Aided Geom. Design*, 7:69–81, 1990.

[10] E.T.Y. Lee. A note on blossoming. *Computer-Aided Geom. Design*, 6:359–362, 1989.

[11] L. Ramshaw. Blossoming: A connect-the-dots approach to splines. Technical report, Digital Systems Research Center, Palo Alto, 1987.

[12] L. Ramshaw. Béziers and B-splines as multiaffine maps. In *Theoretical Foundations of Computer Graphics and CAD*, pages 757–776. Springer, 1988.

[13] L. Ramshaw. Blossoms are polar forms. *Computer-Aided Geom. Design*, 6:323–358, 1989.

[14] A. Rockwood. A brief introduction to blossoming. In *Curve and Surface Design: From Geometry to Applications, SIGGRAPH'92 Course Notes #15*, pages 34–45. ACM SIGGRAPH, 1992.

[15] G. Schmeltz. *Variationsreduzierende Kurvendarstellungen und Krümmungskriterien für Bézierflächen*. PhD thesis, TH Darmstadt, Germany, 1992.

[16] H.-P. Seidel. A new multiaffine approach to B-splines. *Computer-Aided Geom. Design*, 6:23–32, 1989.

[17] H.-P. Seidel. Symmetric recursive algorithms for surfaces: B-patches and the de Boor algorithm for polynomials over triangles. *Constr. Approx.*, 7:257–279, 1991.

[18] H.-P. Seidel. Polar forms and triangular B-Spline surfaces. In *Euclidean Geometry and Computers*. World Scientific Publishing Co., 1992.

[19] H.-P. Seidel. Representing piecewise polynomials as linear combinations of multivariate B-splines. In T. Lyche and L. L. Schumaker, editors, *Curves and Surfaces*, pages 559–566. Academic Press, 1992.

[20] H.-P. Seidel. Polar forms for geometrically continuous spline curves of arbitrary degree. *ACM Trans. Graph.*, 12:1–34, 1993.

[21] K. Strom. *Splines, Polynomials and Polar Forms*. PhD thesis, University of Oslo, Oslo, Norway, 1992.