Fast and Simple Methods For Computing Control Points

Jean Gallier* Weiqing Gu**

*Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA

**Mathematics Department
Harvey Mudd College
1250 N. Dartmouth Ave
Claremont, CA 91711

`jean@saul.cis.upenn.edu`
`gu@math.hmc.edu`

June 10, 2006

Abstract. The purpose of this paper is to present simple and fast methods for computing control points for polynomial curves and polynomial surfaces given explicitly in terms of polynomials (written as sums of monomials). We give recurrence formulae w.r.t. arbitrary affine frames. As a corollary, it is amusing that we can also give closed-form expressions in the case of the frame (r, s) for curves, and the frame $((1, 0, 0), (0, 1, 0), (0, 0, 1)$ for surfaces. Our methods have the same low polynomial (time and space) complexity as the other best known algorithms, and are very easy to implement.

1 Introduction

Polynomial curves and surfaces are used extensively in geometric modeling and computer aided geometric design (CAGD) in particular (see Ramshaw [10], Farin [3, 2], Hoschek and Lasser [7], or Piegl and Tiller [9]). One of the main reasons why polynomial curves and surfaces are used so extensively in CAGD, is that there is a very powerful and versatile algorithm to recursively approximate a curve or a surface using repeated affine interpolation, the *de Casteljau algorithm*. However, the de Casteljau algorithm applies to curves and surfaces only if they are defined in terms of *control points*. There are situations where a curve or a surface is defined explicitly in terms of polynomials. For example, the following polynomials define a surface known as the Enneper surface:

$$\begin{aligned} x(u,v) &= u - \frac{u^3}{3} + uv^2 \\ y(u,v) &= v - \frac{v^3}{3} + u^2v \\ z(u,v) &= u^2 - v^2. \end{aligned}$$

Thus, the problem of computing control points from polynomials (defined as sums of monomials) arises. If control points can be computed quickly from polynomials, all the tools available in CAGD for drawing curves and surfaces can be applied. This could be very useful in problems where a curve of a surface is obtained analytically in terms of polynomials or rational functions, for example, problems involving generalizations of Voronoi diagrams, or motion planning problems. When dealing which such problems, it is often necessary to decide whether curves segments or surface patches intersect or not. As is well known (for example, see [3, 2]), there are effective methods based on subdivision (exploiting the fact that a Bézier curve or surface patch is contained within the convex hull of its control points) for deciding whether Bézier curve segments or surface patches intersect. Simple and fast methods for computing control points might also be also useful to teach say, Math students, to learn computational tools for drawing interesting curves and surfaces. Now, it turns out that the problem of computing control points can be viewed as a change of polynomial basis, more specifically as a change of basis from the monomial basis to bases of Bernstein polynomials. Algorithms for performing such changes of basis have been given by Piegl and Tiller [9]. More general algorithms for performing changes of bases between progressive bases and Pólya bases are presented in Goldman and Barry [5] and Lodha and Goldman [8]. These algorithms compute certain triangles or tetrahedras whose nodes are labeled with certain multisets, and are generalizations of the de Casteljau and the de Boor algorithm. In this paper, we present alternate and more direct methods for computing control points from polynomial definitions (in monomial form) that run in the same low time complexity as the above algorithms ($O(m^2)$ for curves of degree m, $O(p^2q^2)$ for rectangular surfaces of bidegree $\langle p, q\rangle$, and $O(m^3)$ for triangular surfaces of total degree m). Our algorithms are not as general as those of Goldman and Barry [5] and Lodha and Goldman [8], but they are more direct and very easy to implement.

The paper is organized as follows. In section 2, we review briefly the relationship between polynomial definitions and control points. We begin with the polarization of polynomials in one or two variables, and then we show how polynomial curves and surfaces are completely determined by sets of control points. In the case of surfaces, depending on the mode of polarization, we get two kinds of surfaces, bipolynomial surfaces (or rectangular patches) and total degree surfaces (or triangular patches). Efficient methods for computing control points are given in the next three section: polynomial curves in section 3, bipolynomial surfaces in section 4, and polynomial total degree surfaces in section 5. Some examples are given in section 6.

2 Control Points

2.1 Polynomial Curves

The deep reason why polynomial curves and surfaces can be handled in terms of control points is that polynomials in one or several variables can be *polarized*. This means that every polynomial function arises from a unique symmetric multiaffine map. A detailed treatment of this approach can be found in Ramshaw [10], Farin [3, 2], Hoschek and Lasser [7], or Gallier [4]. We simply review what is needed to explain our algorithms.

Recall that a map $f\colon \mathbb{R}^d \to \mathbb{R}^n$ is *affine* if

$$f((1-\lambda)a + \lambda b) = (1-\lambda)f(a) + \lambda f(b),$$

for all $a, b \in \mathbb{R}^d$, and all $\lambda \in \mathbb{R}$. A map $f\colon \underbrace{\mathbb{R}^d \times \cdots \times \mathbb{R}^d}_{m} \to \mathbb{R}^n$ is *multiaffine* if it is affine in each of its arguments, and a map $f\colon \underbrace{\mathbb{R}^d \times \cdots \times \mathbb{R}^d}_{m} \to \mathbb{R}^n$ is *symmetric* if it does not depend on the order of its arguments, i.e., $f(a_{\pi(1)}, \ldots, a_{\pi(m)}) = f(a_1, \ldots, a_m)$ for all $a_1, \ldots, a_m$, and all permutations π. We also say that a map $f\colon \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}^d$ is $\langle p, q\rangle$*-symmetric* if it is symmetric separately in its first p arguments and in its last q arguments.

Let us first treat the case of polynomials in one variable, which corresponds to the case of curves. Given a (plane) polynomial curve $F\colon \mathbb{R} \to \mathbb{R}^2$ of degree m,

$$\begin{aligned} x(t) &= F_1(t), \\ y(t) &= F_2(t), \end{aligned}$$

where $F_1(t)$ and $F_2(t)$ are polynomials of degree $\leq m$, it turns out that $F\colon \mathbb{R} \to \mathbb{R}^2$ comes from a unique symmetric multiaffine map $f\colon \mathbb{R}^m \to \mathbb{R}^2$, the *polar form of* F, such that

$$F(t) = f(\underbrace{t, \ldots, t}_{m}), \quad \text{for all } t \in \mathbb{R}.$$

Furthermore, given any interval (r, s) (affine frame), the map $f\colon \mathbb{R}^m \to \mathbb{R}^2$ is determined by the sequence $(b_0, \ldots, b_m)$ of $m+1$ *control points*

$$b_i = f(\underbrace{r, \ldots, r}_{m-i}, \underbrace{s, \ldots, s}_{i}),$$

where $0 \leq i \leq m$. Using linearity, in order to polarize a polynomial of one variable t, it is enough to polarize a monomial t^k. Since there are $\binom{m}{k}$ terms in the sum

$$\sum_{\substack{I \subseteq \{1,\ldots,m\} \\ |I|=k}} \Big(\prod_{i \in I} t_i\Big),$$

the polar form $f_k^m(t_1, \ldots, t_m)$ of the monomial t^k with respect to the degree m (where $k \leq m$) is given by

$$f_k^m(t_1, \ldots, t_m) = \frac{1}{\binom{m}{k}} \sum_{\substack{I \subseteq \{1,\ldots,m\} \\ |I|=k}} \Big(\prod_{i \in I} t_i\Big).$$

2.2 Polynomial Surfaces Polarization

Given a polynomial surface $F\colon \mathbb{R}^2 \to \mathbb{R}^3$, there are two natural ways to polarize the polynomials defining F.

The first way is to polarize *separately* in u and v. If p is the highest degree in u and q is the highest degree in v, we get a unique $\langle p, q\rangle$-symmetric degree $(p+q)$ multiaffine map

$$f\colon \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}^3,$$

such that

$$F(u, v) = f(\underbrace{u, \ldots, u}_{p}; \underbrace{v, \ldots, v}_{q}).$$

We get what is traditionally called a *tensor product surface*, or as we prefer to call it, a *bipolynomial surface of bidegree* $\langle p, q\rangle$ (or a *rectangular surface patch*).

The second way to polarize is to treat the variables u and v *as a whole*. This way, if F is a polynomial surface such that the maximum total degree of the monomials is m, we get a unique symmetric degree m multiaffine map

$$f\colon (\mathbb{R}^2)^m \to \mathbb{R}^3,$$

such that

$$F(u, v) = f(\underbrace{(u, v), \ldots, (u, v)}_{m}).$$

We get what is called a *total degree surface* (or a *triangular surface patch*).

Using linearity, it is clear that all we have to do is to polarize a monomial $u^h v^k$.

It is easily verified that the unique $\langle p, q\rangle$-symmetric multiaffine polar form of degree $p+q$

$$f^{p,q}_{h,k}(u_1, \ldots, u_p; v_1, \ldots, v_q)$$

of the monomial $u^h v^k$ is given by

$$f^{p,q}_{h,k}(u_1, \ldots, u_p; v_1, \ldots, v_q) = \frac{1}{\binom{p}{h}\binom{q}{k}} \sum_{\substack{I\subseteq\{1,\ldots,p\},|I|=h\\ J\subseteq\{1,\ldots,q\},|J|=k}} \left(\prod_{i\in I} u_i\right)\left(\prod_{j\in J} v_j\right).$$

The denominator $\binom{p}{h}\binom{q}{k}$ is the number of terms in the above sum.

It is also easily verified that the unique symmetric multiaffine polar form of degree m

$$f^{m}_{h,k}((u_1, v_1), \ldots, (u_m, v_m))$$

of the monomial $u^h v^k$ is given by

$$f^{m}_{h,k}((u_1, v_1), \ldots, (u_m, v_m)) = \frac{1}{\binom{m}{h}\binom{m-h}{k}} \sum_{\substack{I\cup J\subseteq\{1,\ldots,m\}\\ |I|=h,|J|=k,I\cap J=\emptyset}} \left(\prod_{i\in I} u_i\right)\left(\prod_{j\in J} v_j\right).$$

The denominator $\binom{m}{h}\binom{m-h}{k} = \binom{m}{h\,k\,(m-h-k)}$ is the number of terms in the above sum.

2.3 Control Points For Polynomial Surfaces

Let $\Delta_m = \{(i,\, j,\, k) \in \mathbb{N}^3 \mid i + j + k = m\}$. Given an affine frame Δrst in the plane (where $r, s, t \in \mathbb{R}^2$ are affinely independent points), a polynomial surface $F\colon \mathbb{R}^2 \to \mathbb{R}^3$ of total degree m specified by the symmetric multiaffine map

$$f\colon (\mathbb{R}^2)^m \to \mathbb{R}^3$$

is completely determined by the family of $\frac{(m+1)(m+2)}{2}$ points in $\mathbb{R}^3$

$$b_{i,\,j,\,k} = f(\underbrace{r, \ldots, r}_{i}, \underbrace{s, \ldots, s}_{j}, \underbrace{t, \ldots, t}_{k}),$$

where $(i, j, k) \in \Delta_m$.

These points are called *control points*, and the family $\{b_{i,j,k} \mid (i,j,k) \in \Delta_m\}$ is called a *triangular control net*.

Let (r_1, s_1) and (r_2, s_2) be any two affine frames for the affine line $\mathbb{R}$. A bipolynomial surface $F\colon \mathbb{R}^2 \to \mathbb{R}^3$ of bidegree $\langle p, q\rangle$ specified by the $\langle p, q\rangle$-symmetric multiaffine map

$$f\colon \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}^3,$$

is completely determined by the family of $(p+1)(q+1)$ points in $\mathbb{R}^3$

$$b_{i,j} = f(\underbrace{r_1, \ldots, r_1}_{p-i}, \underbrace{s_1, \ldots, s_1}_{i}; \underbrace{r_2, \ldots, r_2}_{q-j}, \underbrace{s_2, \ldots, s_2}_{j}),$$

where $0 \leq i \leq p$ and $0 \leq j \leq q$.

These points are called *control points*, and the family $\{b_{i,j} \mid 0 \leq i \leq p,\, 0 \leq j \leq q\}$ is called a *rectangular control net*.

Thus, to compute control points, in principle, we need to compute the polar forms of polynomials. However, this method requires polarization, which is very expensive. In the following sections, we give recurrence formulae for computing control points efficiently. As a corollary, in the case of any affine frame (r, s) or of the affine frame $((1,0), (0,1), (0,0))$, it is possible to give closed-form formulae for calculating control points in terms of binomial coefficients.

3 Computing Control Points For Curves

We saw in section 2 that the polar form of a monomial t^k with respect to the degree m is

$$f_k^m(t_1, \ldots, t_m) = \frac{1}{\begin{pmatrix} m \\ k \end{pmatrix}} \sum_{\substack{I \subseteq \{1,\ldots,m\} \\ |I| = k}} \Bigl(\prod_{i\in I} t_i\Bigr).$$

Letting $\sigma_k^m = \begin{pmatrix} m \\ k \end{pmatrix} f_k^m$, it is easily verified that we have the following recurrence equations:

$$\sigma_k^m = \begin{cases} \sigma_k^{m-1} + t_m \sigma_{k-1}^{m-1} & \text{if } 1 \leq k \leq m; \\ 1 & \text{if } k = 0 \text{ and } m \geq 0; \\ 0 & \text{otherwise.} \end{cases}$$

The above formulae can be used to compute inductively the polar values

$$f_k^m(t_1, \ldots, t_m) = \frac{1}{\begin{pmatrix} m \\ k \end{pmatrix}} \sigma_k^m(t_1, \ldots, t_m).$$

The computation is reminiscent of the Pascal triangle. Alternatively, we can compute f_k^m directly using the recurrence formula

$$f_k^m = \frac{(m-k)}{m}\, f_k^{m-1} + \frac{k}{m} t_m\, f_{k-1}^{m-1},$$

where $1 \leq k \leq m$. When writing computer programs implementing these recurrence equations, we observed that computing σ_k^m and dividing by $\binom{m}{k}$ is faster than computing f_k^m directly using the above formula. This is because the second method requires more divisions.

Given $(t_1, \ldots, t_m)$, computing all the scaled polar values $\sigma_k^i(t_1, \ldots, t_i)$, where $1 \leq k \leq i$ and $1 \leq i \leq m$, requires time $O(m^2)$. The naive method using polarization requires computing $\sum_{i=0}^m 2^i = 2^{m+1} - 1$ terms. To compute the coordinates of control points, we simply combine the σ_k^m. Specifically, the coordinate value of the control point b_j contributed by the polynomial $\sum_{k=0}^n a_k t^k$ (where $n \leq m$) is

$$\sum_{k=0}^{n} a_k \binom{m}{k}^{-1} \sigma_k^m(\underbrace{r, \ldots, r}_{m-j}, \underbrace{s, \ldots, s}_{j}),$$

where (r, s) is an affine frame. Given $(t_1, \ldots, t_m)$, our algorithm computes the table of values $\sigma_k^i(t_1, \ldots, t_i)$, where $1 \leq k \leq i$ and $1 \leq i \leq m$, and thus, it is very cheap to compute these sums.

Remark: Given a polynomial curve F of degree m specified by the sequence of control points $(b_0, \ldots, b_m)$ over $(0, 1)$, it is well known (see Farin [3, 2], Hoschek and Lasser [7], or Piegl and Tiller [9]) that $F(t)$ can be expressed in terms of the Bernstein polynomials $B_k^m(t) = \binom{m}{k}(1-t)^{m-k}t^k$ as

$$F(t) = B_0^m(t)\, b_0 + \cdots + B_m^m(t)\, b_m.$$

It is also well known that the Bernstein polynomials $B_0^m(t), \ldots, B_m^m(t)$ form a basis of the vector space of polynomials of degree $\leq m$. Thus, it is also possible to compute the control points of F by expressing the polynomials involved in the explicit polynomial definition of F in term of the basis Bernstein polynomials. Such algorithms were given by Goldman and Barry [5]. Our algorithm has the same complexity and is more direct.

It is also easy to derive closed-form formulae for any affine frame (r, s).

Theorem 3.1 *If the total degree is m and there are r occurrences where $t = p$ and s occurrences where $t = q$, then*

$$f_k^m = \frac{\binom{p}{k} r^k + \binom{p}{k-1}\binom{q}{1} r^{k-1}s + \binom{p}{k-2}\binom{q}{2} r^{k-2}s^2 +, \ldots, + \binom{q}{k} s^k}{\binom{m}{k}}.$$

Proof. It can be shown by induction using the above recurrence equations. □

4 Computing Rectangular Control Nets

As we saw in section 2, the polar form of the monomial $u^h v^k$ with respect to (p, q) is

$$f_{h,k}^{p,q}(u_1, \ldots, u_p; v_1, \ldots, v_q) = \frac{1}{\left(\begin{array}{c} p \\ h \end{array}\right)\left(\begin{array}{c} q \\ k \end{array}\right)} \sum_{\substack{I \subseteq \{1,\ldots,p\}, |I|=h \\ J \subseteq \{1,\ldots,q\}, |J|=k}} \left(\prod_{i \in I} u_i\right)\left(\prod_{j \in J} v_j\right).$$

Letting $\sigma_{h,k}^{p,q} = \left(\begin{array}{c} p \\ h \end{array}\right)\left(\begin{array}{c} q \\ k \end{array}\right) f_{h,k}^{p,q}$, it is easily verified that we have the following recurrence equations:

$$\sigma_{h,k}^{p,q} = \begin{cases} \sigma_{h,k}^{p-1,q-1} + u_p \sigma_{h-1,k}^{p-1,q-1} + v_q \sigma_{h,k-1}^{p-1,q-1} + u_p v_q \sigma_{h-1,k-1}^{p-1,q-1} & \text{if } 1 \leq h \leq p \text{ and } 1 \leq k \leq q, \\ \sigma_{0,k}^{p,q-1} + v_q \sigma_{0,k-1}^{p,q-1} & \text{if } h = 0 \leq p \text{ and } 1 \leq k \leq q, \\ \sigma_{h,0}^{p-1,q} + u_p \sigma_{h-1,0}^{p-1,q} & \text{if } 1 \leq h \leq p \text{ and } k = 0 \leq q, \\ 1 & \text{if } h = k = 0,\ p \geq 0, \text{ and } q \geq 0; \\ 0 & \text{otherwise.} \end{cases}$$

Observe that the recurrence formula is a sort of generalization of the Pascal triangle. Alternatively, prove that $f_{h,k}^{p,q}$ can be computed directly using the recurrence formula

$$f_{h,k}^{p,q} = \frac{(p-h)(q-k)}{pq}\, f_{h,k}^{p-1,q-1} + \frac{h(q-k)}{pq} u_p\, f_{h-1,k}^{p-1,q-1} + \frac{(p-h)k}{pq} v_q\, f_{h,k-1}^{p-1,q-1} + \frac{hk}{pq} u_p v_q\, f_{h-1,k-1}^{p-1,q-1},$$

where $1 \leq h \leq p$ and $1 \leq k \leq q$,

$$f_{0,k}^{p,q} = \frac{(q-k)}{q}\, f_{0,k}^{p,q-1} + \frac{k}{q} v_q\, f_{0,k-1}^{p,q-1},$$

where $h = 0 \leq p$ and $1 \leq k \leq q$, and

$$f_{h,0}^{p,q} = \frac{(p-h)}{p}\, f_{h,0}^{p-1,q} + \frac{h}{p} u_p\, f_{h-1,0}^{p-1,q},$$

where $1 \leq h \leq p$ and $k = 0 \leq q$. As in section 3, we found that computing $\sigma_{h,k}^{p,q}$ and dividing by $\left(\begin{array}{c} p \\ h \end{array}\right)\left(\begin{array}{c} q \\ k \end{array}\right)$ is faster than computing $f_{h,k}^{p,q}$ directly.

Given $(u_1, \ldots, u_p; v_1, \ldots, v_q)$, using the recurrence equations, computing all the scaled polar values

$$\sigma_{h,k}^{i,j}(u_1, \ldots, u_i; v_1, \ldots, v_j),$$

where $1 \leq h \leq i$, $1 \leq k \leq j$, $1 \leq i \leq p$, and $1 \leq j \leq q$, can be done in time $O(p^2 q^2)$. The naive method using polarization requires computing $\sum_{i=0}^{p} \sum_{j=0}^{q} 2^{i+j} = 2^{p+q+1} - 1$ terms.

To compute the coordinates of control points, we combine the $\sigma_{h,k}^{p,q}$. Specifically, the coordinate value of the control point $b_{i,j}$ contributed by the polynomial $\sum_{h=0}^{m}\sum_{k=0}^{n} a_{h,k}u^h v^k$ (where $m \leq p$ and $n \leq q$) is

$$\sum_{h=0}^{m}\sum_{k=0}^{n} a_{h,k}\binom{p}{h}^{-1}\binom{q}{k}^{-1}\sigma_{h,k}^{p,q}(\underbrace{r_1,\ldots,r_1}_{p-i},\underbrace{s_1,\ldots,s_1}_{i};\underbrace{r_2,\ldots,r_2}_{q-j},\underbrace{s_2,\ldots,s_2}_{j}),$$

where (r_1, s_1) and (r_2, s_2) are affine frames. Our algorithm computes the table of values $\sigma_{h,k}^{m,n}(u_1,\ldots,u_m;v_1,\ldots,v_n)$, and thus, it is very cheap to compute these sums.

When the affine frames $(0, 1)$ are used, the following theorem gives closed-form formulae for the polar values with respect to the bidegree (p, q).

Theorem 4.1 *If there are s occurrences where $u = 0$, r occurrences where $v = 0$, and all the other occurrences of u and v have the value 1, then*

$$f_{h,k}^{p,q}(u_1,u_2,\ldots,u_p;v_1,v_2,\ldots,v_q) = \frac{\binom{p-s}{h}\binom{q-r}{k}}{\binom{p}{h}\binom{q}{k}}$$

.

Proof. It can be shown by induction using the above recurrence equations. □

It is well known that F can be expressed in terms of control points and (products of) Bernstein polynomials (see Farin [3, 2], Hoschek and Lasser [7], or Piegl and Tiller [9]). As in the case of curves, the methods of Lodha and Goldman [8] have the same complexity as ours, but our method is more direct.

5 Computing Triangular Control Nets

As we saw in section 2, the polar form of the monomial $u^h v^k$ with respect to the total degree m is

$$f_{h,k}^{m}((u_1,v_1),\ldots,(u_m,v_m)) = \frac{1}{\binom{m}{h}\binom{m-h}{k}}\sum_{\substack{I\cup J\subseteq\{1,\ldots,m\}\\ |I|=h,|J|=k,I\cap J=\emptyset}}\left(\prod_{i\in I}u_i\right)\left(\prod_{j\in J}v_j\right).$$

Letting $\sigma_{h,k}^{m} = \binom{m}{h}\binom{m-h}{k} f_{h,k}^{m}$, it is easily verified that we have the following recurrence equations:

$$\sigma_{h,k}^{m} = \begin{cases} \sigma_{h,k}^{m-1} + u_m\sigma_{h-1,k}^{m-1} + v_m\sigma_{h,k-1}^{m-1} & \text{if } h,k\geq 0 \text{ and } 1\leq h+k\leq m,\\ 1 & \text{if } h=k=0 \text{ and } m\geq 0;\\ 0 & \text{otherwise.}\end{cases}$$

The above formulae can be used to compute inductively the polar values

$$\frac{1}{\binom{m}{h}\binom{m-h}{k}}\sigma_{h,k}^{m}((u_1,v_1),\ldots,(u_m,v_m)).$$

The computation consists in building a tetrahedron of values reminiscent of the Pascal triangle (but 3-dimensional). Alternatively, we can compute $f_{h,k}^{m}$ directly using the recurrence formula

$$f_{h,k}^{m}=\frac{(m-h-k)}{m}\,f_{h,k}^{m-1}+\frac{h}{m}u_m\,f_{h-1,k}^{m-1}+\frac{k}{m}v_m\,f_{h,k-1}^{m-1},$$

where $h,k\geq 0$ and $1\leq h+k\leq m$. Again, we found that computing $\sigma_{h,k}^{m}$ and dividing by $\binom{m}{h}\binom{m-h}{k}$ is faster than computing $f_{h,k}^{m}$ directly.

Given $((u_1,v_1),\ldots,(u_m,v_m))$, using the recurrence equations, computing all the scaled polar values

$$\sigma_{h,k}^{i}((u_1,v_1),\ldots,(u_i,v_i)),$$

where $h,k\geq 0$, $1\leq h+k\leq i$, and $1\leq i\leq m$, can be done in time $O(m^3)$. The naive method using polarization requires computing $\sum_{i=0}^{m}3^i=(3^{m+1}-1)/2$ terms. To compute the coordinates of control points, we simply combine the $\sigma_{h,k}^{m}$. Specifically, the coordinate value of the control point $b_{i,j,k}$ (where $i+j+k=m$) contributed by the polynomial $\sum_{h+l\leq n}a_{h,l}u^hv^l$ (where $n\leq m$) is

$$\sum_{h+l\leq n}a_{h,l}\begin{pmatrix} m\\ h\;l\;d\end{pmatrix}^{-1}\sigma_{h,l}^{m}(\underbrace{r,\ldots,r}_{i},\underbrace{s,\ldots,s}_{j};\underbrace{t,\ldots,t}_{k}),$$

where $d=m-k-l$ and $r=(r_1,r_2)$, $s=(s_1,s_2)$ and $t=(t_1,t_2)$ are the vertices of the affine frame. Our algorithm computes the table of values $\sigma_{h,l}^{i}((u_1,v_1),\ldots,(u_i,v_i))$, and thus, it is very cheap to compute these sums.

When the affine frame $((1,0),(0,1),(0,0))$ is used, the following theorem gives closed-form formulae for the polar values with respect to the total degree m.

Theorem 5.1 *Assume that $m=r\;+\;s\;+\;t$, with r occurrences of $(1,0)$, s occurrences of $(0,1)$, and t occurrences of $(0,0)$ (and no occurrences of $(1,1)$). Then*

$$f_{h,k}^{m}((u_1,v_1),\ldots,(u_m,v_m))=\frac{\binom{r}{h}\binom{s}{k}}{\binom{m}{h}\binom{m-h}{k}}$$

.

Proof. It can be shown by induction using the above recurrence equations. □

As in the previous case, it is well known that F can be expressed in terms of control points and (trivariate) Bernstein polynomials (see Farin [3, 2], Hoschek and Lasser [7], or Piegl and Tiller [9]). The methods of Lodha and Goldman [8] have the same complexity as ours, but our method is more direct and very easy to implement.

6 Examples

We wrote an implementation in *Mathematica* of a program computing polar values for curves, using the recurrence equations of section 3. It works for an arbitrary affine frame (r, s).

As an example, consider the curve of degree 10 given by

$$\begin{aligned} x &= \frac{4t(1-t^2)^2(1-14t^2+t^4)}{(1+t^2)^5}, \\ y &= \frac{8t^2(1-t^2)(3-10t^2+3t^4)}{(1+t^2)^5}. \end{aligned}$$

Using the above program, the following control polygon w.r.t. $[0, 1]$ is obtained:

```
rcpoly =  {{0, 0, 1}, {2/5, 0, 1}, {18/25, 12/25, 10/9}, {1/2, 6/5, 4/3},
   {-14/45, 71/45, 12/7}, {-45/37, 45/37, 148/63}, {-71/45, 14/45, 24/7},
   {-6/5, -1/2, 16/3}, {-12/25, -18/25, 80/9}, {0, -2/5, 16}, {0, 0, 32}};
```

Note that the control points also contain weights, since there are denominators (see Ramshaw [10], Farin [3, 2], Hoschek and Lasser [7], or Gallier [4]). Here is the rational curve.

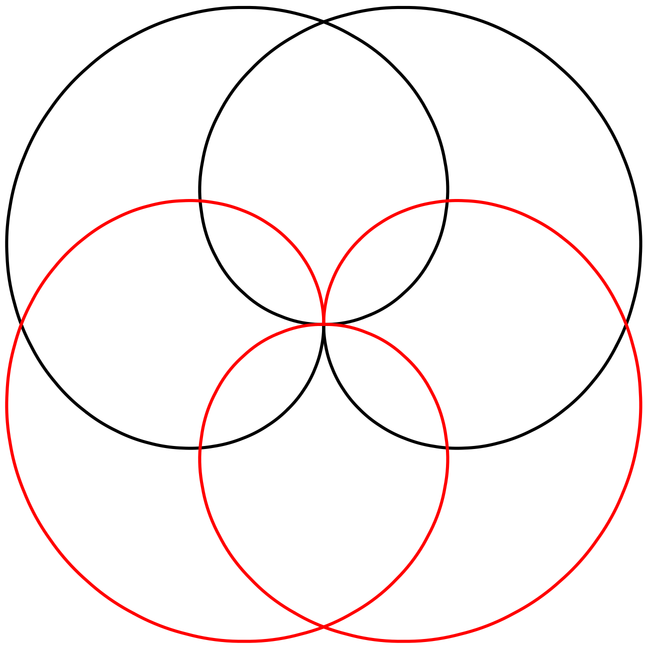

Figure 1: A rose

We also wrote an implementation in *Mathematica* of a program computing polar values for triangular surface patches, using the recurrence equations of section 5. This algorithm works for any affine frame (r, s, t).

In Hilbert and Cohn-Vossen [6] (and also do Carmo [1]), an interesting map $\mathcal{H}$ from $\mathbb{R}^3$ to $\mathbb{R}^4$ is defined as

$$(x, y, z) \mapsto (xy, yz, xz, x^2 - y^2).$$

This map has the remarkable property that when restricted to the sphere S^2, we have $\mathcal{H}(x, y, z) = \mathcal{H}(x', y', z')$ iff $(x', y', z') = (x, y, z)$ or $(x', y', z') = (-x, -y, -z)$. In other words, the inverse image of every point in $\mathcal{H}(S^2)$ consists of two antipodal points. Thus, the map $\mathcal{H}$ induces an injective map from the projective plane onto $\mathcal{H}(S^2)$, which is obviously continuous, and since the projective plane is compact, it is a homeomorphism. Thus, the map $\mathcal{H}$ allows us to realize concretely the projective plane in $\mathbb{R}^4$, by choosing any parameterization of the sphere S^2, and applying the map $\mathcal{H}$ to it. For example, the following parametric definition specifies the entire projective plane over $[-1, 1] \times [-1, 1]$:

$$x = \frac{16uv^2(1 - u^2)}{(u^2 + 1)^2(v^2 + 1)^2},$$

$$y = \frac{8uv(u^2 + 1)(v^2 - 1)}{(u^2 + 1)^2(v^2 + 1)^2},$$

$$z = \frac{4v(1-u^4)(v^2-1)}{(u^2+1)^2(v^2+1)^2},$$
$$t = \frac{4v^2(u^4-6u^2+1)}{(u^2+1)^2(v^2+1)^2}.$$

Using our algorithm, the following net of degree 8 over the affine frame $((1,0,0), (0,1,0), (0,0,1))$ is obtained:

```
proj8net =
   {{0, 0, 0, 0, 1}, {0, 0, -1/2, 0, 1}, {0, 0, -14/15, 2/15, 15/14},
   {0, 0, -20/17, 6/17, 17/14}, {0, 0, -120/101, 60/101, 101/70},
   {0, 0, -1, 4/5, 25/14}, {0, 0, -11/16, 15/16, 16/7}, {0, 0, -1/3, 1, 3},
   {0, 0, 0, 1, 4}, {0, 0, 0, 0, 1}, {0, -1/7, -1/2, 0, 1},
   {4/45, -4/15, -14/15, 2/15, 15/14}, {4/17, -28/85, -20/17, 6/17, 17/14},
   {40/101, -32/101, -120/101, 60/101, 101/70},
   {8/15, -6/25, -1, 4/5, 25/14}, {5/8, -1/8, -11/16, 15/16, 16/7},
   {2/3, 0, -1/3, 1, 3}, {0, 0, 0, 0, 15/14}, {0, -4/15, -7/15, 0, 15/14},
   {20/121, -60/121, -105/121, 9/121, 121/105},
   {10/23, -14/23, -25/23, 9/46, 46/35},
   {240/331, -192/331, -360/331, 108/331, 331/210},
   {80/83, -36/83, -75/83, 36/83, 83/42}, {10/9, -2/9, -11/18, 1/2, 18/7},
   {0, 0, 0, 0, 17/14}, {0, -32/85, -7/17, 0, 17/14},
   {9/46, -16/23, -35/46, -1/46, 46/35},
   {27/53, -89/106, -50/53, -3/53, 53/35},
   {36/43, -100/129, -40/43, -4/43, 129/70},
   {12/11, -6/11, -25/33, -4/33, 33/14}, {0, 0, 0, 0, 101/70},
   {0, -48/101, -34/101, 0, 101/70},
   {56/331, -288/331, -204/331, -40/331, 331/210},
   {56/129, -44/43, -98/129, -40/129, 129/70},
   {16/23, -144/161, -120/161, -80/161, 23/10}, {0, 0, 0, 0, 25/14},
   {0, -14/25, -6/25, 0, 25/14}, {8/83, -84/83, -36/83, -16/83, 83/42},
   {8/33, -38/33, -6/11, -16/33, 33/14}, {0, 0, 0, 0, 16/7},
   {0, -5/8, -1/8, 0, 16/7}, {0, -10/9, -2/9, -2/9, 18/7}, {0, 0, 0, 0, 3},
   {0, -2/3, 0, 0, 3}, {0, 0, 0, 0, 4}};
```

Note that the control points also contain weights, since there are denominators. If we project the real projective plane onto a hyperplane in $\mathbb{R}^4$, either from a center or parallel to a direction, we can see a “3D shadow” of the real projective plane in $\mathbb{R}^3$. For example, one of the projections is the cross-cap, whose control net is

```
projnet4 =
   {{0, 0, 0, 1}, {0, 0, 0, 1}, {0, 0, 2/15, 15/14}, {0, 0, 6/17, 17/14},
   {0, 0, 60/101, 101/70}, {0, 0, 4/5, 25/14}, {0, 0, 15/16, 16/7},
   {0, 0, 1, 3}, {0, 0, 1, 4}, {0, 0, 0, 1}, {0, -1/7, 0, 1},
   {4/45, -4/15, 2/15, 15/14}, {4/17, -28/85, 6/17, 17/14},
   {40/101, -32/101, 60/101, 101/70}, {8/15, -6/25, 4/5, 25/14},
   {5/8, -1/8, 15/16, 16/7}, {2/3, 0, 1, 3}, {0, 0, 0, 15/14},
   {0, -4/15, 0, 15/14}, {20/121, -60/121, 9/121, 121/105},
   {10/23, -14/23, 9/46, 46/35}, {240/331, -192/331, 108/331, 331/210},
   {80/83, -36/83, 36/83, 83/42}, {10/9, -2/9, 1/2, 18/7}, {0, 0, 0, 17/14},
   {0, -32/85, 0, 17/14}, {9/46, -16/23, -1/46, 46/35},
   {27/53, -89/106, -3/53, 53/35}, {36/43, -100/129, -4/43, 129/70},
   {12/11, -6/11, -4/33, 33/14}, {0, 0, 0, 101/70}, {0, -48/101, 0, 101/70},
   {56/331, -288/331, -40/331, 331/210}, {56/129, -44/43, -40/129, 129/70},
   {16/23, -144/161, -80/161, 23/10}, {0, 0, 0, 25/14},
   {0, -14/25, 0, 25/14}, {8/83, -84/83, -16/83, 83/42},
   {8/33, -38/33, -16/33, 33/14}, {0, 0, 0, 16/7}, {0, -5/8, 0, 16/7},
   {0, -10/9, -2/9, 18/7}, {0, 0, 0, 3}, {0, -2/3, 0, 3}, {0, 0, 0, 4}}
```

The cross-cap is shown below.

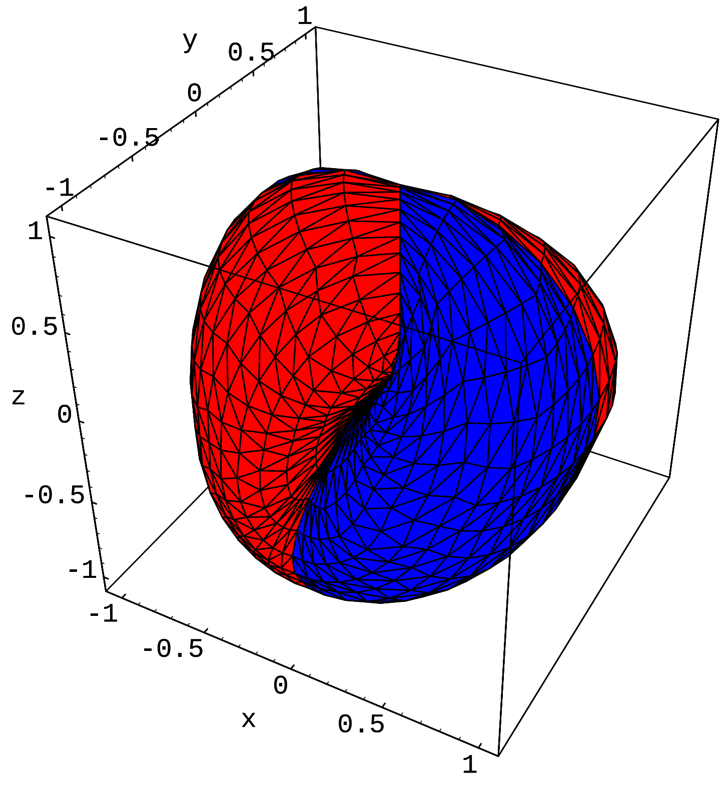

Figure 2: The cross-cap surface

Acknowledgement. I wish to thank Deepak Tolani for very helpful comments, in

particular on the use of Bernstein polynomials.

References

[1] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces.* Prentice Hall, 1976.

[2] Gerald Farin. *NURB Curves and Surfaces, from Projective Geometry to practical use.* AK Peters, first edition, 1995.

[3] Gerald Farin. *Curves and Surfaces for CAGD.* Academic Press, fourth edition, 1998.

[4] Jean H. Gallier. *Curves and Surfaces In Geometric Modeling: Theory And Algorithms.* Morgan Kaufmann, first edition, 1999.

[5] Ronald Goldman and P.J. Barry. Wonderful triangle: a simple, unified, algorithmic approach to change of basis procedures in computer aided geometric design. In T. Lyche and L.L Schumaker, editors, *Mathematical Methods in CAGD II*, pages 297–320. Academic Press, 1992.

[6] D. Hilbert and S. Cohn-Vossen. *Geometry and the Imagination.* Chelsea Publishing Co., 1952.

[7] J. Hoschek and D. Lasser. *Computer Aided Geometric Design.* AK Peters, first edition, 1993.

[8] S.K. Lodha and Ronald Goldman. Change of basis algorithms for surfaces in CAGD. *Computer Aided Geometric Design*, 12:801–824, 1995.

[9] Les Piegl and Wayne Tiller. *The NURBS Book.* Monograph in Visual Communications. Springer Verlag, first edition, 1995.

[10] Lyle Ramshaw. Blossoming: A connect-the-dots approach to splines. Technical report, Digital SRC, Palo Alto, CA 94301, 1987. Report No. 19.