ON EFFICIENT ALGORITHMS FOR POLYNOMIAL EVALUATION IN CAGD

Jorge Delgado and Juan Manuel Peña

Abstract. For evaluating polynomial curves in computer design the usual algorithm is the de Casteljau algorithm. Although it is simple and stable, this algorithm is not efficient, in the sense that it has not linear complexity. In this paper we discuss and compare the properties of four more efficient algorithms used under some circumstances as alternative to the de Casteljau algorithm.

Keywords: Evaluation algorithm; TP; Linear time complexity.

AMS classification: 65D17.

§1. Introduction

Let $\mathcal{U}$ be a vector space of real functions defined on a real interval I and $(u_0(t), \dots, u_n(t))$ $(t \in I)$ a basis of $\mathcal{U}$. Given a sequence $P_0, \dots, P_n$ of points in $\mathbf{R}^k$ we can define a curve

$$\gamma(t) = \sum_{i=0}^{n} P_i \, u_i(t), \quad t \in I.$$

The points $P_0, \dots, P_n$ are called *control points* and the corresponding polygon $P_0 \cdots P_n$ is called the *control polygon* of γ. In Computer Aided Geometric Design the functions $u_0, \dots, u_n$ are usually nonnegative and $\sum_{i=0}^{n} u_i(t) = 1 \; \forall t \in [a, b]$ (i.e. the system $(u_0, \dots, u_n)$ is normalized) and in this case we say that $(u_0, \dots, u_n)$ is a blending system. The *convex hull property* is an important property for interactive design: for any control polygon, the curve always lies in the convex hull of the control polygon. The convex hull property holds if and only if $(u_0, \dots, u_n)$ is a blending system.

These geometric properties correspond to some properties concerning the collocation matrices of the system of functions. The *collocation matrix* of $(u_0(t), \dots, u_n(t))$ at $t_0 < \cdots < t_m$ in I is given by

$$M \begin{pmatrix} u_0, \dots, u_n \\ t_0, \dots, t_m \end{pmatrix} := (u_j(t_i))_{i=0,\dots,m;j=0,\dots,n}. \tag{1}$$

Clearly, $(u_0, \dots, u_n)$ is blending if and only if all its collocation matrices are stochastic (that is, nonnegative and such that the sum of the elements of each row is 1). In interactive design we also want that the shape of a parametrically defined curve mimics the shape of its control polygon; thus we can predict or manipulate the shape of the curve by choosing or changing the

control polygon suitably. In order to assure this property, we have to require more properties to the collocation matrices (1). A matrix is *totally positive of order* r *(*TP_r*)* if all its $k \times k$ minors, $k = 1, \ldots, r$, are nonnegative. A matrix is *totally positive (*TP*)* if all its minors are nonnegative. Obviously, a $n \times n$ TP matrix is TP_i for all $i \in \{1, \ldots, n\}$. More properties on TP matrices can be seen in [1]. A system of functions is TP_r (respectively TP) when all its collocation matrices (1) are TP_r (respectively TP). In case of a normalized totally positive (NTP) basis one knows that the curve imitates the shape of its control polygon, due to the variation diminishing properties of TP matrices (see [17]). In fact, the shape preserving representations are associated with NTP bases.

In CAGD the usual representation of a polynomial curve is the Bernstein-Bézier form:

$$\gamma(t) = \sum_{i=0}^{n} P_i \, b_i^n(t), \quad t \in [0, 1],$$

where $b_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$ $i = 0, \ldots, n$ is the Bernstein basis (see [15] and [12]). This representation presents optimal shape preserving properties (see [5]). Curves in the Bernstein-Bézier form are usually evaluated by using the de Casteljau algorithm. The de Casteljau algorithm is a *corner cutting algorithm*, that is, an algorithm such that each step consist of obtaining a polygonal arc by cutting corners from another polygonal arc (see [5]). Corner cutting algorithms are useful because they have good stability properties (see [16]).

In spite of the nice properties of the de Casteljau algorithm, its computational cost to evaluate a polynomial curve of degree n is quadratic (that is, of $\mathcal{O}(n^2)$ elementary operations). There are some circumstances, frequently in the surfaces design (see [14] and [20]), where it is convenient to use polynomial evaluation algorithms with less computational cost than the de Casteljau algorithm. This has stimulated in the last two decades the research of new algorithms for the polynomial evaluation which are useful in CAGD (see [10],[18],[19],[23],[21],[6],[7] and [22]). In fact, there are other evaluation algorithms useful in design whose computational cost to evaluate a polynomial curve of degree n is linear (that is, of $\mathcal{O}(n)$ elementary operations). In this paper we will present some of these alternative algorithms. Schumaker and Volk presented in [19] an alternative algorithm to the de Casteljau algorithm for evaluating polynomials in the Bernstein-Bézier form. We show this algorithm in Section 2. In [2], [3] and [4] Ball presented a basis of the space of polynomials of degree less than or equal to 3 on $[0, 1]$. Two different generalized Ball bases of higher degree were obtained by Said and Wang independently in [18] and [23] respectively. Both bases present evaluation algorithms with less computational cost than the de Casteljau algorithm. We present the properties of the Wang-Ball and Said-Ball bases and their evaluation algorithms in sections 3 and 4 respectively. In section 5 we present another basis introduced in [10] whose evaluation algorithm associated has linear time complexity. Finally, in section 6 we present a summary of the properties satisfied by the algorithms considered mentioned along the paper, that is, the computational cost of the algorithm, kind of algorithm and shape preserving properties of the associated basis.

§2. The VS algorithm

In [19] Schumaker and Volk proposed a new algorithm to evaluate multivariate polynomials in the Bernstein-Bézier representation. Here, we particularize this algorithm to the univariate

case. Let $\gamma(t) = \sum_{i=0}^{m} V_i \binom{m}{i} t^i (1-t)^{m-i}$ be a polynomial in the Bernstein-Bezier form. Taking $P_i = \binom{m}{i} V_i$ for $i = 0, \ldots, m$, we can write

$$\gamma(t) = \sum_{i=0}^{m} P_i \, t^i (1-t)^{m-i}, \;\; t \in [0,1].$$

This representation is called Bernstein-Bézier modified representation.

Definition 1. Let $(z_0^m(t), \ldots, z_m^m(t))$, $m \geq 2$, $t \in [0,1]$, be the *VS basis*, defined by $z_i^m(t) = t^i(1-t)^{m-i}$.

The VS basis has the following evaluation algorithm associated which was presented by Schumaker and Volk (see [19]):

Algorithm 1

Let $\gamma(t) = \sum_{i=0}^{m} P_i \, z_i^m(t)$ and $t \in [0,1]$.

1.– If $t \geq 1/2$
  perform step 2
else
  perform step 3
End-If.

2.– $coc = \frac{1-t}{t}$
$A = P_0$
For $i = 1, m \; step \; 1$
  $A = A * coc + P_i$
End-For
$\gamma(t) = A \times t^m$

3.– $coc = \frac{t}{1-t}$.
$A = P_m$
For $i = 1, m \; step \; 1$
  $A = A * coc + P_{m-i}$
End-For
$\gamma(t) = A \times (1-t)^m$.

We can easily checked that the algorithm consists of m sums, $2m$ products and one quotient, so that it has linear complexity. In addition, since the system $(z_0^m, \ldots, z_m^m)$ coincides up to scaling with the Bernstein basis $(b_0^m, \ldots, b_m^m)$ we can conclude that $(z_0^m, \ldots, z_m^m)$ is also TP. However, as we can observe in Algorithm 1, the VS algorithm is not a corner cutting algorithm.

§3. The Wang-Ball algorithm

The Wang-Ball basis was presented by Wang in [23] as a generalization of the Ball basis (see [2], [3] and [4]). In addition, the Wang-Ball basis has been also considered in [21] and [6]. Let us recall the definition of the Wang-Ball basis.

Definition 2. Let $(a_0^m(t),\dots,a_m^m(t))$, $m\geq 2$, $t\in[0,1]$, be the Wang-Ball system, defined by:

$$\begin{array}{ll} a_i^m(t)=2^i t^i(1-t)^{i+2}, & 0\leq i\leq \left\lfloor \frac{m}{2}\right\rfloor -1,\\ a_i^m(t)=2^{m-i}t^{m+2-i}(1-t)^{m-i}, & \left\lfloor \frac{m+1}{2}\right\rfloor +1\leq i\leq m.\end{array}$$

In addition, if m is even,

$$a_{\frac{m}{2}}^m(t)=2^{\frac{m}{2}}t^{\frac{m}{2}}(1-t)^{\frac{m}{2}},$$

and, if m is odd,

$$a_{\frac{m-1}{2}}(t)=2^{\frac{m-1}{2}}t^{\frac{m-1}{2}}(1-t)^{\frac{m+1}{2}},\quad a_{\frac{m+1}{2}}(t)=2^{\frac{m-1}{2}}t^{\frac{m+1}{2}}(1-t)^{\frac{m-1}{2}},$$

where $\lfloor r\rfloor$ ($r>0$) is the greatest positive integer less than or equal to r.

The Wang-Ball basis has the following evaluation algorithm associated, which was presented in [21].

Algorithm 2

Let $\gamma(t)=\sum_{i=0}^m P_i\, a_i^m(t)$ and $t\in[0,1]$.

1.– For $i=0,\dots,m$

$f_i^0(t)=P_i$

End-For.

2.– For $k=m,\dots,3$ step$=-1$

If k is odd

$$\begin{array}{l} f_i^{m+1-k}(t)=f_i^{m-k}(t),\ \ 0\leq i\leq \frac{k-3}{2},\\ f_{\frac{k-1}{2}}^{m+1-k}(t)=(1-t)f_{\frac{k-1}{2}}^{m-k}(t)+tf_{\frac{k+1}{2}}^{m-k}(t),\\ f_i^{m+1-k}(t)=f_{i+1}^{m-k}(t),\ \ \frac{k+1}{2}\leq i\leq k-1,\end{array}$$

else

$$\begin{array}{l} f_i^{m+1-k}(t)=f_i^{m-k}(t),\ \ 0\leq i\leq \frac{k}{2}-2,\\ f_{\frac{k}{2}-1}^{m+1-k}(t)=(1-t)f_{\frac{k}{2}-1}^{m-k}(t)+tf_{\frac{k}{2}}^{m-k}(t),\\ f_{\frac{k}{2}}^{m+1-k}(t)=(1-t)f_{\frac{k}{2}}^{m-k}(t)+tf_{\frac{k}{2}+1}^{m-k}(t),\\ f_i^{m+1-k}(t)=f_{i+1}^{m-k}(t),\ \ \frac{k}{2}+1\leq i\leq k-1,\end{array}$$

End If

End-For.

3.– $f_0^{m-1}(t)=(1-t)f_0^{m-2}(t)+tf_1^{m-2}(t)$

$f_1^{m-1}(t)=(1-t)f_1^{m-2}(t)+tf_2^{m-2}(t)$

$f_0^m(t)=(1-t)f_0^{m-1}(t)+tf_1^{m-1}(t)$

then, $f_0^m(t)=\gamma(t)$.

We can easily check that the previous algorithm consists, if m is even, of $\frac{3}{2}m$ sums and $3m$ products and, if m is odd, of $\frac{1}{2}(3m-1)$ sums and $3m-1$ products. In addition, as we can observe, Algorithm 2 is a corner cutting algorithm and so it will have good stability properties. Besides, the Wang-Ball basis is blending. However, by Theorem 3.7 of [8], the Wang-Ball basis is not TP_2. Therefore we cannot expect that this representation satisfies strong shape preserving properties.

§4. The Said-Ball algorithm

Another generalized Ball basis is the Said-Ball basis. The Said-Ball basis was defined by Said in [18] for the space of polynomials of degree at most m with odd m. Goodman and Said proved in [13] the total positivity of this family of bases. Later, Shi-Min Hu, Guo-Zhao Wang and Tong-Guang Jin suggested in [21] an extension of this family to the case of polynomials of degree at most m with even m. Let us present this complete family of bases:

Definition 3. The Said-Ball basis $(s_0^m(t),\dots,s_m^m(t))$, $m\geq 1$, $t\in[0,1]$, is defined by:

$$\begin{aligned} s_i^m(t) &= \binom{\lfloor \frac{m}{2}\rfloor + i}{i} t^i (1-t)^{\lfloor \frac{m}{2}\rfloor+1}, \qquad 0\leq i\leq \left\lfloor \frac{m-1}{2}\right\rfloor, \\ s_i^m(t) &= \binom{\lfloor \frac{m}{2}\rfloor + m - i}{m-i} t^{\lfloor \frac{m}{2}\rfloor+1} (1-t)^{m-i}, \qquad \left\lfloor \frac{m}{2}\right\rfloor + 1\leq i\leq m, \end{aligned}$$

and, if m is even

$$s_{\frac{m}{2}}^m(t) = \binom{m}{\frac{m}{2}} t^{\frac{m}{2}} (1-t)^{\frac{m}{2}}.$$

The Said-Ball basis has the following evaluation algorithm associated, which was presented in [21].

Algorithm 3

Let $\gamma(t)=\sum_{i=0}^m P_i\, s_i^m(t)$ and $t\in[0,1]$.

1.– For $i=0,\dots,m$
$f_i^0(t)=P_i$
End For.

2.– For $k=m,\dots,3$ step$=-1$
If k is odd
$f_i^{m+1-k}(t)=f_i^{m-k}(t),\ \ 0\leq i\leq \frac{k-3}{2},$
$f_{\frac{k-1}{2}}^{m+1-k}(t)=(1-t)f_{\frac{k-1}{2}}^{m-k}(t)+tf_{\frac{k+1}{2}}^{m-k}(t),$
$f_i^{m+1-k}(t)=f_{i+1}^{m-k}(t),\ \ \frac{k+1}{2}\leq i\leq k-1,$
else
For $i=0,\dots,k$
$g_i^{0,k}(t)=f_i^{m-k}(t)$
End For.

$g_i^{1,k}(t) = g_i^{0,k}(t), \;\; 0 \leq i \leq \frac{k}{2} - 2,$
$g_{\frac{k}{2}-1}^{1,k}(t) = (1-t)g_{\frac{k}{2}-1}^{0,k}(t) + tg_{\frac{k}{2}}^{0,k}(t),$
$g_{\frac{k}{2}}^{1,k}(t) = (1-t)g_{\frac{k}{2}}^{0,k}(t) + tg_{\frac{k}{2}+1}^{0,k}(t),$
$g_i^{1,k}(t) = g_{i+1}^{0,k}(t), \;\; \frac{k}{2} + 1 \leq i \leq k-1,$
For $j = \frac{k}{2} - 1, \ldots, 1$ step$= -1$
$g_i^{\frac{k}{2}+1-j,k}(t) = g_i^{\frac{k}{2}-j,k}(t), \;\; 0 \leq i \leq j-2,$
$g_{j-1}^{\frac{k}{2}+1-j,k}(t) = (1-t)g_{j-1}^{\frac{k}{2}-j,k}(t) + tg_j^{\frac{k}{2}-j,k}(t),$
$g_i^{\frac{k}{2}+1-j,k}(t) = g_i^{\frac{k}{2}-j,k}(t), \;\; j \leq i \leq k-j-1,$
$g_{k-j}^{\frac{k}{2}+1-j,k}(t) = (1-t)g_{k-j-1}^{\frac{k}{2}-j,k}(t) + tg_{k-j}^{\frac{k}{2}-j,k}(t),$
$g_i^{\frac{k}{2}+1-j,k}(t) = g_i^{\frac{k}{2}-j,k}(t), \;\; k-j+1 \leq i \leq k-1,$
End-For
$f_i^{m+1-k}(t) = g_i^{\frac{k}{2},k}(t), \;\; 0 \leq i \leq k-1$
End-If
End-For.

3.– $f_0^{m-1}(t) = (1-t)f_0^{m-2}(t) + tf_1^{m-2}(t)$
$f_1^{m-1}(t) = (1-t)f_1^{m-2}(t) + tf_2^{m-2}(t)$
$f_0^m(t) = (1-t)f_0^{m-1}(t) + tf_1^{m-1}(t)$

then, $f_0^m(t) = \gamma(t)$.

We can easily check that the previous algorithm consists, if m is even, of $\frac{m}{2}\left(\frac{m}{2}+2\right)$ sums and $m\left(\frac{m}{2}+2\right)$ products and, if m is odd, of $\frac{(m+1)^2}{4}$ sums and $\frac{(m+1)^2}{2}$ products. It has less computational cost than the de Casteljau algorithm (see [21]). In addition, as we can observe, Algorithm 3 is a corner cutting algorithm and so it will have good stability properties. Finally, by Proposition 3 of [9], the Said-Ball basis is NTP and so the curves represented in the Said-Ball basis imitate the shape of its corresponding control polygons (recall that this was proved in [13] only for the case when m is odd).

§5. Another basis with a linear complexity evaluation algorithm

In [10] Delgado and Peña introduced another basis with an evaluation algorithm associated of linear complexity. Let us recall the definition of this basis.

Definition 4. Let $(c_0^m(t), \ldots, c_m^m(t))$, $m \geq 2$, be the system of polynomials on $[0,1]$ defined by:

$$\begin{aligned} c_0^m(t) &= (1-t)^m, \\ c_i^m(t) &= t(1-t)^{m-i}, \; 1 \leq i \leq \left\lfloor \frac{m}{2} \right\rfloor - 1, \end{aligned}$$

$$\begin{aligned} c_i^m(t) &= t^i(1-t), \ \left\lfloor \frac{m+1}{2} \right\rfloor + 1 \leq i \leq m-1, \\ c_m^m(t) &= t^m. \end{aligned}$$

In addition, if m is even,

$$c_{\frac{m}{2}}^m(t) = 1 - t^{\frac{m}{2}+1} - (1-t)^{\frac{m}{2}+1},$$

and, if m is odd,

$$\begin{aligned} c_{\frac{m-1}{2}}^m(t) &= t(1-t)^{\frac{m+1}{2}} + \frac{1}{2}\left[1 - t^{\frac{m+1}{2}} - (1-t)^{\frac{m+1}{2}}\right], \\ c_{\frac{m+1}{2}}^m(t) &= \frac{1}{2}\left[1 - t^{\frac{m+1}{2}} - (1-t)^{\frac{m+1}{2}}\right] + t^{\frac{m+1}{2}}(1-t). \end{aligned}$$

Let us observe that, for $m = 2$, the previous system coincides with the Bernstein basis on $[0, 1]$. More properties of this basis can be seen in [10].

In [10] the authors also present the following evaluation algorithm associated to the representation corresponding to this basis.

Algorithm 4

Let $\gamma(t) = \sum_{i=0}^m P_i\, c_i^m(t)$ and $t \in [0, 1]$.

1.– For $i = 0, 1, \ldots, m$

$f_i^0(t) = P_i$

End-For.

2.– If $m \geq 3$ and m is odd

$f_0^1(t) = (1-t)f_0^0(t) + tf_1^0(t)$,
$f_i^1(t) = f_{i+1}^0(t), \ \ 1 \leq i \leq \frac{m-3}{2}$,
$f_{\frac{m-1}{2}}^1(t) = \frac{1}{2}f_{\frac{m-1}{2}}^0(t) + \frac{1}{2}f_{\frac{m+1}{2}}^0(t)$,
$f_i^1(t) = f_i^0(t), \ \ \frac{m+1}{2} \leq i \leq m-2$,
$f_{m-1}^1(t) = (1-t)f_{m-1}^0(t) + tf_m^0(t)$,

else if $m \geq 3$ and m is even

$f_0^1(t) = (1-t)f_0^0(t) + tf_1^0(t)$,
$f_i^1(t) = f_{i+1}^0(t), \ \ 1 \leq i \leq \frac{m}{2} - 1$,
$f_i^1(t) = f_i^0(t), \ \ \frac{m}{2} \leq i \leq m-2$,
$f_{m-1}^1(t) = (1-t)f_{m-1}^0(t) + tf_m^0(t)$,

End-If.

3.– For $k = m-1, \ldots, 3$ step$= -1$

If k is odd

$f_0^{m+1-k}(t) = (1-t)f_0^{m-k}(t) + tf_1^{m-k}(t)$,
$f_i^{m+1-k}(t) = f_{i+1}^{m-k}(t), \ \ 1 \leq i \leq \frac{k-1}{2}$,
$f_i^{m+1-k}(t) = f_i^{m-k}(t), \ \ \frac{k+1}{2} \leq i \leq k-2$,

$$f_{k-1}^{m+1-k}(t) = (1-t)f_{k-1}^{m-k}(t) + tf_k^{m-k}(t),$$

else

$$f_0^{m+1-k}(t) = (1-t)f_0^{m-k}(t) + tf_1^{m-k}(t),$$
$$f_i^{m+1-k}(t) = f_{i+1}^{m-k}(t), \;\; 1 \leq i \leq \tfrac{k}{2} - 1,$$
$$f_i^{m+1-k}(t) = f_i^{m-k}(t), \;\; \tfrac{k}{2} \leq i \leq k-2,$$
$$f_{k-1}^{m+1-k}(t) = (1-t)f_{k-1}^{m-k}(t) + tf_k^{m-k}(t),$$

End-If

End-For.

4.– $f_0^{m-1}(t) = (1-t)f_0^{m-2}(t) + tf_1^{m-2}(t)$

$f_1^{m-1}(t) = (1-t)f_1^{m-2}(t) + tf_2^{m-2}(t)$

$f_0^m(t) = (1-t)f_0^{m-1}(t) + tf_1^{m-1}(t)$

then, $f_0^m(t) = \gamma(t)$.

In Remark 2 of [10] was checked that, when m is odd, the number of sums is $2m$ and the number of multiplications is $4m$ and, when m is even, the number of sums is $2m-1$ and the number of multiplications is $4m-2$. Besides, Algorithm 4 is a corner cutting algorithm and so it will present good stability properties. In addition, by Theorem 5 and Theorem 6 of [10], the system $(c_0^m, \ldots, c_m^m)$ is NTP for all $m \geq 2$. Thus, the curves represented in the system $(c_0^m, \ldots, c_m^m)$ imitate the shape of its corresponding control polygons.

§6. Conclusion

We show in Table 1 a comparison of the results of sections 2, 3, 4 and 5. As we can see, the de Casteljau algorithm is a corner cutting algorithm but it has not linear time complexity. On the other hand, the Bernstein basis, which is the basis associated to the de Casteljau algorithm, is NTP and so, this representation is shape preserving. We can also see in Table 1 that the VS basis is TP (although it is not NTP, a scaling produces the Bernstein basis which is NTP) and the VS algorithm, which is its associated algorithm, has linear time complexity but it is not a corner cutting algorithm. Again, in Table 1 we can see that the Wang-Ball algorithm has linear time complexity and, in addition, it is a corner cutting algorithm. But its associated basis, that is, the Wang-Ball basis, is not TP. The another generalised Ball basis, that is, the Said-Ball basis, is NTP. In addition, its corresponding evaluation algorithm, that is the Said-Ball algorithm, is a corner cutting algorithm, but it has not linear complexity although it has less computational cost than the de Casteljau algorithm. Finally, the basis introduced by the authors satisfies simultaneously all these good properties, that is, it is NTP, its associated evaluation algorithm has linear time complexity and is a corner cutting algorithm. In [11] it has been used for rational curves evaluation and in a future research it will be used for surfaces design.

References

[1] T. ANDO Totally Positive Matrices. *Linear Algebra Appl. 90* (1987), 165–219.

	TP basis	NTP basis	corner cutting algorithm	linear complexity
de Casteljau algorithm	✓	✓	✓	
VS algorithm	✓			✓
Wang-Ball algorithm			✓	✓
Said-Ball algorithm	✓	✓	✓	
Authors algorithm	✓	✓	✓	✓

Table 1: Summary of properties of polynomial bases

[2] BALL A. A. CONSURF, Part one: Introduction to conic lifting title. *Computer-Aided Design 6* (1974), 243–249.

[3] BALL A. A. CONSURF, Part two: Description of the algorithms. *Computer-Aided Design 7* (1975), 237–242.

[4] BALL A. A. CONSURF, Part three: How the program is used. *Computer-Aided Design 9* (1977), 9–12.

[5] CARNICER, J. M. AND PEÑA, J. M. Shape preserving representations and optimality of the Bernstein basis. *Advances in Computational Mathematics 1* (1993), 173–196.

[6] DEJDUMRONG, N. AND PHIEN, H. N. Efficient algorithms for Bezier curves. *Comput. Aided Geom. Design 17* (2000), 247–250.

[7] DEJDUMRONG, N., PHIEN, H. N., TIEN, H. L. AND LAY, K. M. Rational Wang-Ball curves. *Int. J. Math. Educ. Sci. Technol. 32* (2001), 565–584.

[8] DELGADO, J. AND PEÑA, J. M. Monotonicity preservation of some polynomial and rational representations. In *Information Visualisation*, Ebad Banissi and Muhammad Sarfraz, IEEE Computer Society, Los Alamitos (CA), 2002, pp. 57–62.

[9] DELGADO, J. AND PEÑA, J. M. On the generalized Ball bases. *Preprint*.

[10] DELGADO J. AND PEÑA, J. M. A shape preserving representation with an evaluation algorithm of linear complexity. *Comput. Aided Geom. Design 20*, 1 (2003), 1–10.

[11] DELGADO J. AND PEÑA, J. M. A Shape Preserving Representation for Rational Curves with Efficient Evaluation Algorithm. In *Advances in Geometric Modeling*. Muhammad Sarfraz, Chichester West Sussex, John Wiley, 2004.

[12] FARIN, G. *Curves and Surfaces for Computer Aided Geometric Design (4th edition)*. Academic Press, San Diego, 1996.

[13] GOODMAN, T. N. T. AND SAID, H. B. Shape preserving properties of the generalised Ball basis. *Comput. Aided Geom. Design 8* (1991), 115–121.

[14] GUOJIN, W. AND MIN, C. New algorithms for evaluating parametric surface. *Progress in Natural Science 11* (2001), 142–148.

[15] HOSCHECK, J. AND LASSER, D. *Fundamentals of Computer Aided Geometric Design*. Wellesley, AKPeters, 1993.

[16] MAINAR, E. AND PEÑA, J. M. Error analysis of corner cutting algorithms. *Numerical Algorithms 22* (1999), 41–52.

[17] PEÑA, J. M. *Shape preserving representations in Computer Aided-Geometric Design*. Nova Science Publishers, Commack (NY), 1999.

[18] SAID, H. B. Generalized Ball curve and its recursive algorithm. *ACM. Trans. Graph. 8* (1989), 360–371.

[19] SCHUMAKER L. L. AND VOLK, W. Efficient evaluation of multivariate polynomials. *Comput. Aided Geom. Design 3*, 2 (1986), 149–154.

[20] SHI-MIN, H., GUOJIN, W. AND JIAGUANG, S. A Type of Triangular Ball Surface and Its Properties. *J. of Comput. Sci. & Technol. 13* (1998), 63–72.

[21] SHI-MIN, H., GUO-ZHAO, W. AND TONG-GUANG, J. Properties of two types of generalized Ball curves. *Computer-Aided Design 28* (1996), 125–133.

[22] TIEN, H. L., HANSUEBSAI, D. AND PHIEN, H. N. Rational Ball curves. *Int. J. Math. Educ. Sci. Technol. 30* (1999), 243–257.

[23] WANG, G-J. Ball curve of high degree and its geometyic properties. *Appl. Math.: A journal of Chinese Universities 2* (1987), 126–140.