

**A Polynomial Basis with Distinct Advantages  
for Adaptive Forward Differencing**

**Salim S. Abi-Ezzi**

**January 18, 1995**

**3D Technology Group,  
Integrated Media Platform Software,  
SunSoft Incorporated,  
Mt. View, CA 94043**

**(submitted for Eurographics '93)**



# Abstract

We present a new polynomial basis that has distinct advantages for the adaptive forward differencing polynomial evaluation method. The basis belongs to the Polya family of bases and hence it exhibits the convex hull property and it interpolates the first and last control points. In addition, the new basis like the forward differencing basis, exhibits an efficient forward step of linear complexity. This forward property generalizes to arbitrary degree polynomials.

Furthermore, we use the basis for the tessellation of planar cubic curves through adaptive forward differencing. The adaptation is indirectly based on the convex hull property and is used for enforcing a threshold on the chord-to-curve deviation. The result is the fastest known method for tessellating planar cubics with this deviation enforcement feature.

## 1– Introduction

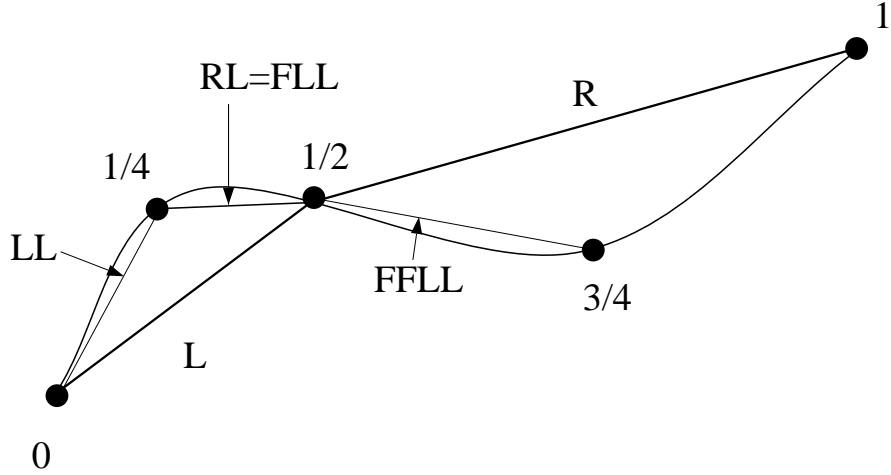
A significant step in the graphical processing of parametric curved primitives is the tessellation of such primitives into a piecewise linear form that meets a certain approximation criterion. The tessellation process results with what we call tessellants, which are chords in the case of curves and triangles in the case of surfaces. The approximation criterion is in the form of a threshold on the size of a tessellant, or alternatively a threshold on the maximum deviation\* of a tessellant from the original primitive; see [PHIGS PLUS91].

Adaptive forward differencing (AFD) is a variation of the forward differencing polynomial evaluation technique, which adapts the step size while forward marching across a primitive [Lien87]. The technique capitalizes on the efficiency of the forward differencing technique, and adapts the step size in order to diminish the number of generated tessellants while still satisfying a certain approximation criterion. Lien et al used the technique for enforcing a threshold on the size of the tessellant. However, experiments showed us that enforcing a deviation threshold is more effective than a size threshold due to the behavior of parametric primitives; see [Abi-Ezzi89].

Without loss of generality we focus on unit curve segments that correspond to  $t \in [0, 1]$ . AFD is a sequential polynomial (curve) sampling technique, which is based on applying efficient linear parameter substitutions of the form  $(t \leftarrow at + b)$  to the polynomial, through the transformation of the polynomial's coefficients via linear transformations. So, for a cubic with four coefficients a linear reparameterization

---

\* In this paper by maximum deviation we mean the maximum distance from a curve segment to the chord.



Recursive Subdivision (L, R)	$L : t \leftarrow t/2$
Forward Differencing ( $L_n, F$ )	$R : t \leftarrow (t+1)/2$
Adaptive F. D. ( $L_n, F, L, L^1$ )	$F : t \leftarrow t+1$
	$L^1 : t \leftarrow 2t$

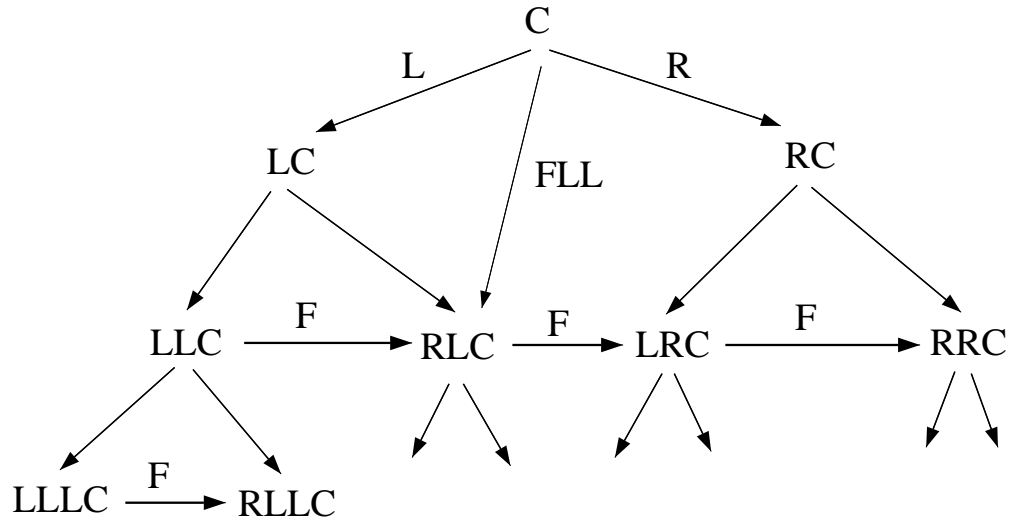


Figure 1: Recursive subdivision, forward differencing, and adaptive F. D. as polynomial reparameterization operations.

is achievable via multiplying the coefficients vector with a  $4 \times 4$  matrix. The specific reparameterization operations that are needed for AFD are as follows:

1. The forward step is accomplished via:  $F : t \leftarrow t + 1$ .
2. The left-half (adjust-down) step is accomplished via:  $L : t \leftarrow t/2$ .
3. The adjust-up step is accomplished via:  $L^{-1} : t \leftarrow 2t$ .

We also introduce an extra operation for explanatory purposes:

4 The right-half step is accomplished via:  $R : t \leftarrow (t + 1)/2$ .

Applying the  $L$  and  $R$  operations recursively to a unit curve, which corresponds to the recursive subdivision of the unit curve, results with dissecting the curve in a binary tree fashion; see Figure 1. The  $L$  operation results with a transition to a lower level in the tree and is referred to as an adapt down operation, while the  $L^{-1}$  operation results with a transition to an upper level of the tree and is referred to as an adapt up operation. Furthermore, the  $F$  operation results with a forward step from one leaf to the right hand adjacent leaf in the tree. After each forward step, the approximation criterion is checked and a decision is made on whether to adapt up or down, or to proceed forward. In practical situations, the rate of adaptation is much lower than that of advancing forward.

The FD polynomial basis, see [Lien87], exhibits a very efficient matrix for the  $F$  operation. This matrix is a two diagonals band matrix, where all band entries are one; hence, the cost of applying the matrix consists of  $d$  additions and no multiplications where  $d$  is the degree of the polynomial. From linear algebra the FD basis can be shown to be optimal for the  $F$  operation; i.e. among all polynomial bases it is the basis that exhibits the most efficient form of the  $F$  operation; see [Abi-Ezzi89].

Applying AFD in the FD basis works well for the size approximation criterion but not for the deviation criterion. The reason is that the FD basis has no provisions for checking the maximum deviation of a primitive from its tessellant. Traditionally, recursive subdivision in the Bézier basis has been the method of choice for handling the deviation criterion because of the Bézier convex hull property. Through this property it becomes possible to insure a threshold on the deviation of the curve from a tessellant. However, both recursive subdivision and FD in the Bézier basis are expensive, since the matrices that correspond to the  $R$ ,  $L$ , and  $F$  operations are upper/lower triangular and hence are at a cost of  $d(d + 1)$  FLOPs per step. The  $F$  matrix in the Bézier basis as well as in most other bases, turns out to be triangular and hence is at a cost of  $d(d + 1)$  operations. For this reason, we pursued a basis that exhibits the convex hull property for enforcing the deviation criterion, and that exhibits an efficient  $F$  matrix. Next we present such a basis.

## 2– A New Basis for Adaptive Forward Differencing

This basis is the uniform knot Polya basis (the Bézier basis is also a Polya basis), see [Goldman83, 84], therefore it exhibits two important properties: 1) it interpolates the first and last control points, and 2) its control points constitute a convex hull for the primitive. Furthermore, this basis has an efficient,

$O(d)$ , forward step. The basis is defined recursively as follows:

$$\begin{aligned}\alpha_0^0(t) &= 1 \\ \alpha_i^d(t) &= \frac{t+i-1}{d}\alpha_{i-1}^{d-1}(t) + \frac{d-i-t}{d}\alpha_i^{d-1}(t)\end{aligned}\quad (1)$$

An alternative closed form expression is as follows:

$$\alpha_i^d(t) = \frac{(-1)^{d-i}}{(d-i)!i!} \prod_{j=1-i}^{d-i} (t-j) \quad (2)$$

It is straight forward to verify that the basis satisfies the properties above.

Given that this basis interpolates the end control points, then a forward step is equivalent to a “next evaluation” of the polynomial\*; basically, one of the new coefficients after the reparameterization operation is the next point on the curve. As we’ve seen, in the FD basis the  $F$  matrix turns out to be very sparse with a cost of  $d$  operations. In the new basis, the  $F$  matrix also turns out to be sparse, as will be shown next. We give the proofs of this section’s results in Appendix A.

The polynomials of the new basis exhibit the following relations which can be proven based on Equations 1 and 2:

$$\begin{aligned}\alpha_i^d(t+1) &= -\frac{i+1}{d-i}\alpha_{i+1}^d(t), \quad i \in \{0, \dots, d-1\}, \\ \alpha_d^d(t+1) &= \sum_{i=0}^d \frac{d+1}{d-i+1}\alpha_i^d(t).\end{aligned}\quad (3)$$

From Equation 3 we can deduce that  $F$  has the following form, where the  $A_i$ s are point coefficients of the polynomial curve:

$$\begin{bmatrix} A'_0 \\ A'_1 \\ A'_2 \\ \vdots \\ A'_{d-2} \\ A'_{d-1} \\ A'_d \end{bmatrix} = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 & 0 & 1 \\ -\frac{1}{d} & 0 & \dots & 0 & 0 & 0 & \frac{d+1}{d} \\ 0 & -\frac{2}{d-1} & \dots & 0 & 0 & 0 & \frac{d+1}{d-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & -\frac{d-2}{3} & 0 & 0 & \frac{d+1}{3} \\ 0 & 0 & \dots & 0 & -\frac{d-1}{2} & 0 & \frac{d+1}{2} \\ 0 & 0 & \dots & 0 & 0 & -d & d+1 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{d-2} \\ A_{d-1} \\ A_d \end{bmatrix}. \quad (4)$$

Notice that each row in  $F$  sums to unity; therefore, for  $i \in \{1, 2, \dots, d\}$ ,  $A'_i \in \overline{A_{i-1}A_d}$ , since any affine combination of two points lie on the line determined by these two points.

---

\* This fact is not true in general, for example in the case of the uniform B-spline basis we have an efficient forward step, but the basis does not interpolate the end points and hence a forward step does not lead to a “next evaluation” of the polynomial.

We convert the  $(d + 1)$  equations of  $F$  to an equivalent set of equations, where point  $A_0$  and scales of the vectors  $\vec{v}_i = (A_i - A_0), i \in \{1, 2, \dots, d\}$  are forwarded instead of the control points  $(A_i)$ ; see Figure 2. This modified version of  $F$  is as follows:

$$\begin{bmatrix} A'_0 \\ \binom{d}{1} \vec{v}'_1 \\ \binom{d}{2} \vec{v}'_2 \\ \vdots \\ \binom{d}{d-2} \vec{v}'_{d-2} \\ \binom{d}{d-1} \vec{v}'_{d-1} \\ \vec{v}'_d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 & \binom{1}{d} \\ 0 & -1 & 0 & \dots & 0 & 0 & \binom{1}{d} \\ 0 & 0 & -1 & \dots & 0 & 0 & \binom{2}{d} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -1 & 0 & \binom{d-2}{d} \\ 0 & 0 & 0 & \dots & 0 & -1 & \binom{d-1}{d} \end{bmatrix} \begin{bmatrix} A_0 \\ \binom{d}{1} \vec{v}_1 \\ \binom{d}{2} \vec{v}_2 \\ \vdots \\ \binom{d}{d-2} \vec{v}_{d-2} \\ \binom{d}{d-1} \vec{v}_{d-1} \\ \vec{v}_d \end{bmatrix}. \quad (5)$$

The modified forward step,  $F'$ , requires  $(2d + \lceil \frac{d-1}{2} \rceil - 1)$  operations that are integer multiplies and adds; i.e., exact operations. Therefore, similarly to the FD basis, a forward step in the modified new basis (Equation 5) does not create new errors in the coefficients, but exaggerates initial ones. Notice that a forward step in the unmodified new basis (Equation 4) does create new errors, since the rational entries in the matrix can only be approximated via floating point numbers.

### 3– Specifics for Planar Cubics

From above, it is clear that the forward properties of the new basis generalize to arbitrary degree and n-dimensional vector valued polynomials; however, in this section we will focus on its practical significance for planar cubic curves. Planar cubics represent a significant case because even in the case of displaying 3D cubics, the deviation criterion is typically intended to control the deviation of tessellants in the 2D screen space; i.e. after the viewing transformation (possibly perspective) has been applied. Therefore, we perform the deviation criterion check based on the  $X$  and  $Y$  coordinates only; the  $Z$ -coordinate is ignored since it is orthogonal to the screen.

In the case of cubics the basis is as follows:

$$\begin{aligned} \alpha_0^3(t) &= -\frac{1}{6}(t-1)(t-2)(t-3), \\ \alpha_1^3(t) &= +\frac{1}{2}t(t-1)(t-2), \\ \alpha_2^3(t) &= -\frac{1}{2}(t+1)t(t-1), \\ \alpha_3^3(t) &= +\frac{1}{6}(t+2)(t+1)t. \end{aligned} \quad (6)$$

The original  $F$  matrix is as follows:

$$\begin{bmatrix} A'_0 \\ A'_1 \\ A'_2 \\ A'_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ -1/3 & 0 & 0 & 4/3 \\ 0 & -1 & 0 & 2 \\ 0 & 0 & -3 & 4 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{bmatrix}.$$

The modified  $F'$  matrix is as follows:

$$\begin{bmatrix} A'_0 \\ 3\vec{v}'_1 \\ 3\vec{v}'_2 \\ \vec{v}'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 3 \\ 0 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{bmatrix}. \quad (7)$$

Figure 2 shows a forward step on a cubic, notice the geometric relationship between the  $A'_i$ s and the  $A_i$ s.

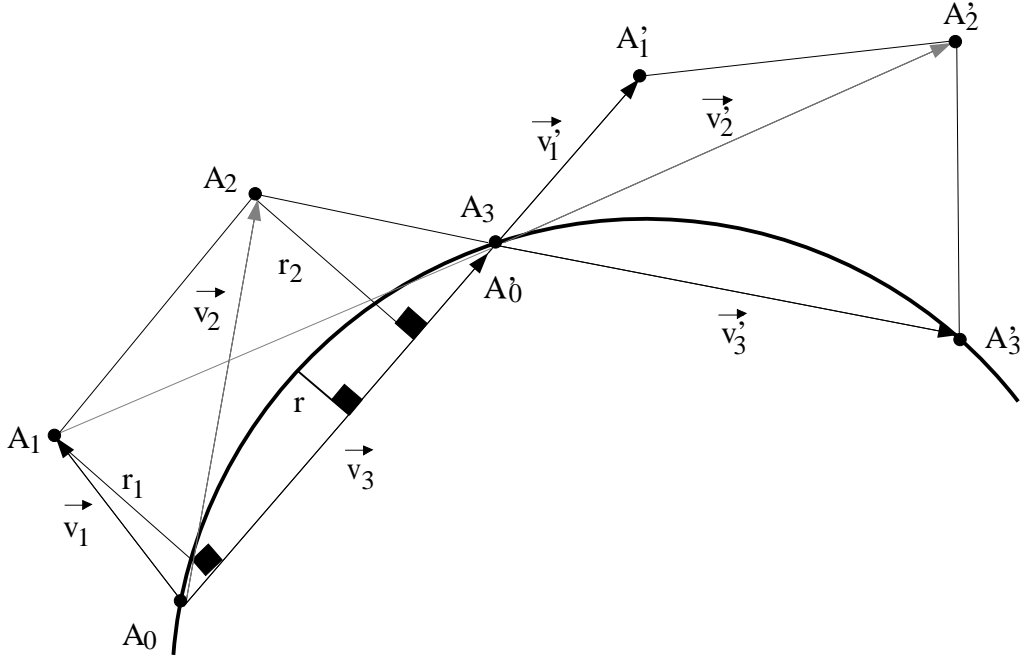


Figure 2: The linear relation between the “next” set of control points to the previous one.

After forwarding into a new portion of the curve, the convex hull of that portion is used to check the deviation criterion and establish a decision on adaptation. The two distances ( $r_1$  and  $r_2$ ), from internal control points ( $A_1$  and  $A_2$ ) to chord  $(\overline{A_0A_3})$ , can be used to estimate the maximum deviation, see Figure 2. Furthermore, it is possible to inflate the tolerance by 3 and then use  $3\vec{v}'_1$  and  $3\vec{v}'_2$  to find  $3r_1$  and  $3r_2$ , respectively, and thus there is no need to find  $\vec{v}_1$  nor  $\vec{v}_2$  explicitly. Therefore, a forward step for cubics can be carried out at a cost of four FLOPs per coordinate; which is pretty close to the 3 FLOPs of ordinary FD.

The adaptation matrices are as follows:

$$L : \begin{bmatrix} A'_0 \\ 3\vec{v}'_1 \\ 3\vec{v}'_2 \\ \vec{v}'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 5/16 & 1/16 & 1/16 \\ 0 & 1/4 & 1/4 & 1/4 \\ 0 & 1/16 & 1/16 & 5/16 \end{bmatrix} \begin{bmatrix} A_0 \\ 3\vec{v}_1 \\ 3\vec{v}_2 \\ \vec{v}_3 \end{bmatrix},$$

and

$$L^{-1} : \begin{bmatrix} A'_0 \\ 3\vec{v}'_1 \\ 3\vec{v}'_2 \\ \vec{v}'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 4 & -1 & 0 \\ 0 & -4 & 6 & -4 \\ 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} A_0 \\ 3\vec{v}_1 \\ 3\vec{v}_2 \\ \vec{v}_3 \end{bmatrix},$$

which can be carried out at a cost of eight and seven FLOPs, respectively. Notice that all entries can be represented exactly in fixed-point binary.

### 3.1– Tightness of the Convex Hull

The effectiveness of the new basis in dealing with the deviation approximation criterion is dependent on the tightness of the convex hull (proximity to the curve) exhibited by this basis\*. Define  $\Theta$  to be the set of cubic polynomial bases that exhibit the convex hull property and that interpolate end points. Define  $S^\theta(C) = \max(r_1^\theta, r_2^\theta)$  to be the size of the convex hull of a given planar unit cubic curve  $C(t)$  under some basis  $\theta \in \Theta$ , where  $r_1^\theta$  and  $r_2^\theta$  are the distances from the two internal control points in basis  $\theta$  to the chord. From approximation theory it is known that the Bézier basis exhibits a fairly tight convex hull around the curve, and we compare the tightness of the convex hull of the new basis to that of the Bézier basis.

To compare the tightness of the new convex hull to that of the Bézier convex hull, we express the actual maximum deviation  $r$  as a function of  $r_1$  and  $r_2$  in these two bases, which leads to the two functions:  $r = f^\beta(r_1^\beta, r_2^\beta) = f^\alpha(r_1^\alpha, r_2^\alpha)$ . By assuming that  $r_1 \geq r_2$ , without loss of generality, it was possible to find  $f^\beta$  and  $f^\alpha$  analytically; see Appendix B. By plotting these functions, they turn out to be almost linear; see Figure 3. Hence,

$$r \approx (b_1 r_1^\beta + b_2 r_2^\beta) \quad \text{and} \quad r \approx (a_1 r_1^\alpha + a_2 r_2^\alpha),$$

for the following constants:  $b_1 = \frac{1}{2}$ ,  $b_2 = \frac{1}{4}$ ,  $a_1 = \frac{1}{5}$ , and  $a_2 = \frac{7}{100}$ . It is likely that this “almost linear behavior” is a general trend across bases  $\in \Theta$ .

We apply this important phenomena to essentially equate between the effectiveness of the new convex hull and that of the Bézier convex hull, although the former is much wider than the former, in determining

\* In relation to the Polya family, the new basis corresponds to tension parameter  $a_1 = 1$ , while the Bézier basis corresponds to tension parameter  $a_1 = 0$ ; see [Goldmann83, 84]. This tension parameter can go all the way to  $+\infty$ , and the larger it is the further the convex hull is from the curve.

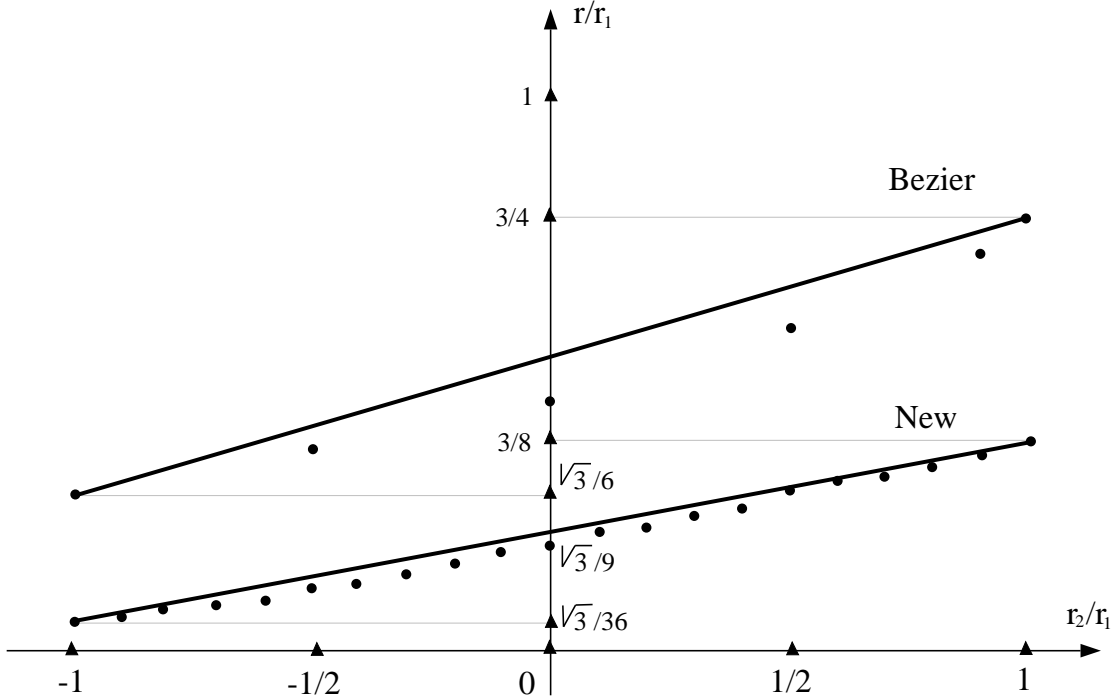


Figure 3: The almost linear relation between  $r$ , and  $r_1$  and  $r_2$ .

an upper bound to the maximum deviation of a curve from a chord. Once  $r_1$  and  $r_2$  are determined, we use  $(a_1 r_1^\beta + a_2 r_2^\beta)$  as a much closer upper bound of the maximum deviation,  $r$ , than what  $\max(r_1^\alpha, r_2^\alpha)$  or  $\max(r_1^\beta, r_2^\beta)$  offers.

Two other interesting findings, which are illustrated in Figure 3, are as follows:

1. For the Bézier basis we have:

$$\forall C, r \leq \frac{3}{4} S^\beta(C).$$

In other words, for unit cubic curves the size of the Bézier convex hull always over estimates the maximum deviation by more than 33%.

2. For the new basis we have:

$$2S^\beta(C) \leq S^\alpha(C) \leq 6S^\beta(C).$$

In other words, for a certain unit cubic curve the size of the convex hull in the new basis is between twice and six times the one in the Bézier basis.

### 3.2– Precision under Floating-Point Computations

Another important issue regarding the new basis is the error behavior under floating-point computations. As in the case of the FD basis, the operations of  $F'$  in Equation 7 are exact; there is no accumulation

of errors. The sole matter of concern is the exaggeration/propagation of the coefficient's initial error; see [Abi-Ezzi89] for a detailed analysis. This can be analyzed by inspecting  $F^n$ , where  $n$  is the number of applications of  $F'$ . This analysis leads to the same worst case results as the analysis for the FD basis. However, in test cases, the new basis exaggerated/propagates errors faster than did the FD basis.

## 4– Conclusion

We have presented a new polynomial basis with distinct properties for the AFD polynomial evaluation technique. The basis exhibits the convex hull property, interpolates end points, and has an efficient forward step. Also this basis produces the fastest method that we know of, for adaptively evaluating cubic curves while enforcing a deviation threshold. The decision to adapt is indirectly based on the convex hull property of the basis. For the purpose of display, the method is effective for tessellating 3D cubics (even if viewed in perspective), since in that case it is sufficient to enforce the deviation threshold in the 2D screen space.

In the future it would be interesting to generalize the results presented here to arbitrary degree curves and to surface primitives.

## References

1. [Abi-Ezzi89] Salim Abi-Ezzi: "The Graphical Processing of B-splines in a Highly Dynamic Environment," RPI Ph.D. dissertation, RDRC-TR-89001, Troy, New York, May 1989.
2. [Goldman83] Ronald Goldman: "An Urnful of Blending Functions," IEEE Computer Graphics & Applications, 4(10), pp 49-54, October 1983.
3. [Goldman85] Ronald Goldman: "Polya's Urn Model and Computer Aided Geometric Design," SIAM J. Alg. Disc. Meth., 6(1), pp 1-28, January 1985.
4. [Lien87] Sheue-Ling Lien, Michael Shantz, and Vaughan Pratt: "Adaptive Forward Differencing for Rendering Curves and Surfaces," Computer Graphics, 21(4), pp 111–117, July 1987.
5. [PHIGS PLUS91] Programmer's Hierarchical Interactive Graphics System- Plus Lumière Und Surfaces (PHIGS PLUS), Functional Description, ISO/IEC 9592–4:199x, Draft International Standard, Feb 1991.

## Appendix A– The Forward Operation In the New Basis

We deduce the sparse forms of the forward ( $F$ ) and modified forward ( $F'$ ) operations in the new basis, for arbitrary degree  $d$  polynomials; see Section 2.

We use Equations 1 and 2 in order to prove Equation 3:

$$\begin{aligned}\alpha_i^d(t+1) &= -\frac{i+1}{d-i}\alpha_{i+1}^d(t), \quad i \in \{0, \dots, d-1\}, \\ \alpha_d^d(t+1) &= \sum_{i=0}^d \frac{d+1}{d-i+1}\alpha_i^d(t),\end{aligned}$$

from which the form of  $F$  in Equation 4 is derived. We start by proving the first component of the above equation:

$$\begin{aligned}\alpha_0^d(t+1) &= \frac{(-1)^{d-i}}{(d-i)!i!} \prod_{j=1-i}^{d-i} (t+1-j) \\ &= \frac{(-1)(-1)^{d-(i+1)}(i+1)}{(d-i)(d-(i+1))!(i+1)!} \prod_{j=(1-(i+1))}^{d-(i+1)} (t+1-(j+1)) \\ &= \frac{-(i+1)}{(d-i)}\alpha_{i+1}^d(t).\end{aligned}$$

For the second equation we use a proof by recursion on  $d$ , as follows:

1. Check the initial case of  $d = 1$ :

$$\begin{aligned}\text{for } d = 1, \quad &\begin{cases} \alpha_0^1(t) = (1-t) \\ \alpha_1^1(t) = t \end{cases} \Rightarrow \\ \alpha_1^1(t+1) &= \frac{1+1}{1+1}(1-t) + \frac{1+1}{1+1-1}t \Rightarrow \\ \alpha_1^1(t+1) &= 1-t+2t = t+1.\end{aligned}$$

2. Assume the relation is correct for the case of degree  $d-1$ ; i.e:

$$\alpha_{d-1}^{d-1}(t+1) = \sum_{i=0}^{d-1} \frac{d}{d-i}\alpha_i^{d-1}(t).$$

3. Show that it is correct for the case of degree  $d$ . We need three results in order to show that:

- a. First we show that

$$\alpha_d^d(t+1) = \alpha_d^d(t) + \alpha_{d-1}^{d-1}(t+1),$$

which is equivalent to

$$\alpha_d^d(t) = \alpha_d^d(t+1) - \alpha_{d-1}^{d-1}(t+1).$$

Expand right hand side:

$$\begin{aligned}
& \alpha_d^d(t+1) - \alpha_{d-1}^{d-1}(t+1) \\
&= \frac{1}{d!} \prod_{j=1-d}^0 (t+1-j) - \frac{1}{(d-1)!} \prod_{j=2-d}^0 (t+1-j) \\
&= \frac{t+d}{d!} \prod_{j=2-d}^0 (t+1-j) - \frac{1}{(d-1)!} \prod_{j=2-d}^0 (t+1-j) \\
&= \frac{t+d-d}{d!} \prod_{j=2-d}^0 (t+1-j) \\
&= \frac{1}{d!} \prod_{j=2-d}^1 (t+1-j) \\
&= \frac{1}{d!} \prod_{j=1-d}^0 (t-j) \\
&= \alpha_d^d(t).
\end{aligned}$$

b. Second, we show that

$$\alpha_i^{d-1}(t) = \frac{d-i}{d} \alpha_i^d(t) + \frac{(i+1)}{d} \alpha_{i+1}^d(t).$$

Expand right hand side

$$\begin{aligned}
& \frac{d-i}{d} \alpha_i^d(t) + \frac{(i+1)}{d} \alpha_{i+1}^d(t) \\
&= \frac{(-1)^{d-i}(d-i)}{d(d-i)!i!} \prod_{j=1-i}^{d-i} (t-j) + \frac{(-1)^{d-i-1}(i+1)}{d(d-i-1)!(i+1)!} \prod_{j=-i}^{d-i-1} (t-j) \\
&= \frac{(-1)^{d-i}(d-i)(t-d+i)}{d(d-i)!i!} \prod_{j=1-i}^{d-i-1} (t-j) + \frac{(-1)^{d-i-1}(i+1)(t+i)}{d(d-i-1)!(i+1)!} \prod_{j=1-i}^{d-i-1} (t-j) \\
&= \left[ \frac{(-1)^{d-i}(t-d+i)}{d(d-i-1)!i!} + \frac{(-1)^{d-i-1}(t+i)}{d(d-i-1)!i!} \right] \prod_{j=1-i}^{d-i-1} (t-j) \\
&= \frac{(t+i) \left( (-1)^{d-i} + (-1)^{d-i+1} \right) - d(-1)^{d-i}}{d(d-i-1)!i!} \prod_{j=1-i}^{d-i-1} (t-j) \\
&= \frac{(-1)^{d-i-1}}{(d-i-1)!i!} \prod_{j=1-i}^{d-i-1} (t-j) \\
&= \alpha_i^{d-1}(t).
\end{aligned}$$

c. Third, we show that

$$\begin{aligned}
& \alpha_d^d + \sum_{i=0}^{d-1} \left( \alpha_i^d + \frac{i+1}{d-i} \alpha_{i+1}^d \right) \\
&= \alpha_d^d + \sum_{i=0}^{d-1} \alpha_i^d + \sum_{i=0}^{d-1} \frac{i+1}{d-i} \alpha_{i+1}^d \\
&= \alpha_d^d + \alpha_0^d + \sum_{i=1}^{d-1} \alpha_i^d + \frac{d}{1} \alpha_d^d + \sum_{i=0}^{d-2} \left( \frac{i+1}{d-i} \right) \alpha_{i+1}^d \\
&= (d+1) \alpha_d^d + \alpha_0^d + \sum_{i=1}^{d-1} \alpha_i^d + \sum_{i=1}^{d-1} \frac{i}{d-i+1} \alpha_i^d \\
&= (d+1) \alpha_d^d + \alpha_0^d + \sum_{i=1}^{d-1} \left( \frac{d+1}{d-i+1} \right) \alpha_i^d \\
&= \sum_{i=0}^d \frac{d+1}{d-i+1} \alpha_i^d.
\end{aligned}$$

Now we are ready for the final step:

$$\begin{aligned}
\alpha_d^d(t+1) &= \alpha_d^d(t) + \alpha_{d-1}^{d-1}(t+1) \dots \text{from (a)} \\
&= \alpha_d^d(t) + \sum_{i=0}^{d-1} \frac{d}{d-i} \alpha_i^{d-1}(t) \dots \text{from (2)} \\
&= \alpha_d^d(t) + \sum_{i=0}^{d-1} \frac{d}{d-i} \left( \frac{d-i}{d} \alpha_i^d(t) + \frac{i+1}{d} \alpha_{i+1}^d(t) \right) \dots \text{from (b)} \\
&= \alpha_d^d(t) + \sum_{i=0}^{d-1} \left( \alpha_i^d + \frac{i+1}{d-i} \alpha_{i+1}^d \right) \dots \text{from (c)} \\
&= \sum_{i=0}^d \frac{d+1}{d+1-i} \alpha_i^d(t),
\end{aligned}$$

The proof is complete, and hence we have the form of  $F$  in Equation 4.

Now we prove the modified form of  $F$ , which is given in Equation 5.

1. Multiply each equation of  $F$  with the denominator of the coefficients of the corresponding row, we get the following equivalent set of equations:

$$\begin{aligned}
A'_0 &= A_d \\
dA'_1 &= (d+1)A_d - A_0 \\
(d-1)A'_2 &= (d+1)A_d - 2A_1 \\
(d-2)A'_3 &= (d+1)A_d - 3A_2 \\
&\vdots \\
2A'_{d-1} &= (d+1)A_d - (d-1)A_{d-2} \\
A'_d &= (d+1)A_d - dA_{d-1}
\end{aligned}$$

2. Multiply each equation  $i, i \in \{2, \dots, d\}$  by a constant  $k_i$ , so that the left hand side of  $(i-1)$  and the right term in the right hand side of  $i$  have equivalent coefficients, (note that for  $i \geq \lfloor \frac{d+1}{2} \rfloor$  the numerators of the coefficients start having common factors with the denominators), we get the following:

$$\begin{aligned}
A'_0 &= A_d \\
dA'_1 &= (d+1)A_d - A_0 \\
\frac{(d-1)d}{2}A'_2 &= \frac{d(d+1)}{2}A_d - dA_1 \\
\frac{(d-2)(d-1)d}{2 \cdot 3}A'_3 &= \frac{(d-1)d(d+1)}{2 \cdot 3}A_d - \frac{(d-1)d}{2}A_2 \\
\frac{(d-3)(d-2)(d-1)d}{2 \cdot 3 \cdot 4}A'_4 &= \frac{(d-2)(d-1)d(d+1)}{2 \cdot 3 \cdot 4}A_d - \frac{(d-2)(d-1)d}{2 \cdot 3}A_3 \\
&\vdots \\
dA'_{d-1} &= \frac{d(d+1)}{2}A_d - \frac{(d-1)d}{2}A_{d-2} \\
A'_d &= (d+1)A_d - dA_{d-1}
\end{aligned}$$

3. For each equation  $i, i \in \{1, \dots, d\}$ , subtract  $\binom{d}{i}A'_0$  from the left-hand-side, and  $\binom{d}{i}A_d$  from the right-hand-side (we can do this since  $A'_0 = A_d$ ), we get:

$$\begin{aligned}
A'_0 &= A_0 + \vec{v}_d \\
\binom{d}{1}\vec{v}'_1 &= d(A'_1 - A'_0) = [(d+1) - d]A_d - A_0 = \vec{v}_d \\
\binom{d}{2}\vec{v}'_2 &= \binom{d}{2}(A'_2 - A'_0) = \left[ \frac{d(d+1)}{2} - \frac{(d-1)d}{2} \right] A_d - dA_1 = \binom{d}{1}(\vec{v}_d - \vec{v}_1) \\
\binom{d}{3}\vec{v}'_3 &= \begin{cases} \left[ \frac{(d-1)d(d+1)}{2 \cdot 3} - \frac{(d-2)(d-1)d}{2 \cdot 3} \right] A_d \\ - \binom{d}{2}A_2 = \binom{d}{2}(\vec{v}_d - \vec{v}_2) \end{cases} \\
&\vdots \\
\binom{d}{d-1}\vec{v}'_{d-1} &= \binom{d}{d-2}(\vec{v}_d - \vec{v}_{d-2}) \\
\vec{v}'_d &= \binom{d}{d-1}(\vec{v}_d - \vec{v}_{d-1})
\end{aligned}$$

where  $\vec{v}_i = (A_i - A_0)$ . This gives us the form of  $F'$  in Equation 5.

## Appendix B– The Exact Maximum Deviation of a Planar Cubic

For the case of planar cubic curves we express the exact maximum deviation as a function of the control points. We give such an expression for both the new and the Bézier bases.

Let  $A_0, A_1, A_2,$  and  $A_3$  be the four control points of a planar cubic in the new basis. Without loss of generality, let  $A_0$  be at the origin,  $A_3$  on the  $X$  axis,  $A_1$  on the positive side of the  $Y$  axis and at a distance of  $r_1$  from the  $X$  axis, and  $A_2$  is at a distance of  $r_2 = kr_1, k \in [-1, +1]$  from the  $X$  axis; see Figure A.1. Let  $r(t)$  be the distance from  $C(t)$  to the  $X$  axis, and  $r_{\max} = \max_{t \in [0,1]}(r(t))$ .

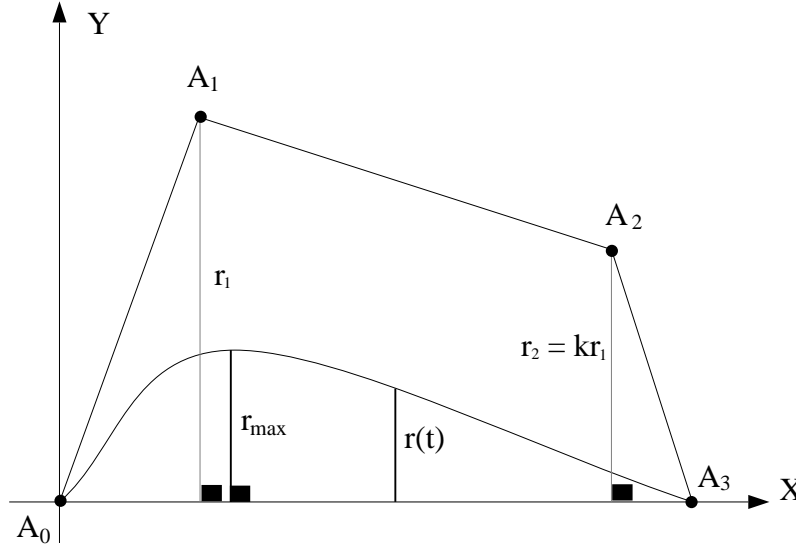


Figure A. 4: The maximum deviation as a function of the distances of the two internal control points to the chord.

It is clear that both  $r(t)$  and  $r_{\max}$  are functions of  $r_1$  and  $r_2$ . In Equation 6 we gave the  $\alpha_i^3$  cubic basis polynomials. First we find  $r(t)$  and then we find  $r_{\max}$ .

$$\begin{aligned}
 r(t) &= r_1\alpha_1^3(t) + kr_1\alpha_2^3(t) \\
 &= \frac{1}{2}r_1t(t-1)[(t-2) - k(t+1)] \\
 &= \frac{1}{2}r_1t(t-1)[(1-k)t - (2+k)] \\
 &= \frac{1}{2}r_1[(1-k)t^3 - 3t^2 + (k+2)t].
 \end{aligned}$$

In order to study the maximum of this function we need to inspect its derivative:

$$\begin{aligned}
 r'(t) &= \frac{1}{2}r_1[3(1-k)t^2 - 6t + k + 2] \Rightarrow \\
 \Delta &= 3(k^2 + k + 1) > 0, \forall k \in [-1, +1] \Rightarrow \\
 t_1 &= \frac{3 - \sqrt{\Delta}}{3(1-k)}, \quad t_2 = \frac{3 + \sqrt{\Delta}}{3(1-k)}.
 \end{aligned}$$

Due to our construction, a maximum will always occur at  $t_1$ ; notice that  $\forall k \in [-1, +1], t_1 \in ]0, 1[$  and  $\exists k \in [-1, +1], \ni t_2 \notin [0, 1]$ . Therefore,

$$\begin{aligned} r_{\max} &= r(t_1) \Rightarrow \\ r_{\max} &= \frac{r_1(2\sqrt{3}\Delta^{\frac{3}{2}} - 9k(k+1))}{18(k-1)^2} \Rightarrow \\ \frac{r_{\max}}{r_1} &= \frac{2\sqrt{3}(3(k^2+k+1))^{\frac{3}{2}} - 9k(k+1)}{18(k-1)^2}, \end{aligned}$$

which when plotted for  $k \in [-1, +1]$  gives the almost linear curve in Figure 3.

We applied the same analysis to the Bézier basis, thus substituting  $\beta_1^3 \rightarrow \alpha_1^3$  and  $\beta_2^3 \rightarrow \alpha_2^3$  in the above analysis. This leads to the following:

$$\begin{aligned} r(t) &= 3(1-k)t^3 + 3(k-2)t^2 + 3t, \\ r'(t) &= 3[3(1-k)t^2 + 2(k-2)t + 1], \\ \Delta &= k^2 - k + 1, \\ t_1 &= \frac{(k-2) + \sqrt{\Delta}}{3(k-1)}, \\ \frac{r_{\max}}{r_1} &= \frac{2(k^2 - k + 1)^{\frac{3}{2}} + (k+1)(k-2)(2k-1)}{9(k-1)^2}. \end{aligned}$$

which also behave almost linearly as shown in Figure 3.

We did carry this analysis to 3D cubics, however, in that case things become more costly and less practical. The added cost arises from the fact that we have an added parameter that influences  $r(t)$ , this parameter being the angular deviation of  $A_0A_2$  from plane  $A_0A_1A_3$ . We have observed that the extreme cases from 2D remain the extremes in 3D (for the new basis at  $k = -1$ ,  $(r_{\max}/r_1) = \frac{\sqrt{3}}{36}$  and at  $k = +1$ ,  $(r_{\max}/r_1) = \frac{3}{8}$ , and for the Bézier basis at  $k = -1$ ,  $(r_{\max}/r_1) = \frac{\sqrt{3}}{6}$  and at  $k = +1$ ,  $(r_{\max}/r_1) = \frac{3}{4}$ ). Therefore, for 3D Bézier cubics the Bézier convex hull always over estimates the actual maximum deviation by at least 33%. Hence, in case of using Bézier recursive subdivision to enforce a deviation threshold, it is recommended to enlarge such a threshold by 33% to achieve better results.