Offset Approximation Improvment
by
Control Points Perturbation *

Gershon Elber[†] and Elaine Cohen
Department of Computer Science
University of Utah
Salt Lake City, UT 84112 USA

Abstract

There does not usually exist a closed general NURBs representation of the offset curve (or surface) to a NURBs curve (surface). In a related paper [7] a method was developed to determine a sequence of approximations to the offset curve (surface) of a given curve (surface) with the properties that the global error of each approximation from the true offset can be bounded and the sequence converges to the true offset. In this paper we take the next step and develop a method using the analysis of the offset error function that perturbs the curve or surface control points of a specified NURBs approximation to an offset of a NURBs curve (surface) so a better approximation to the offset results.

1 Introduction

Offset surfaces are very important in manufacturing, and computation so finding approximations of offset curves and surfaces have undergone extensive research. Creating an offset to a curve is an intuitive operation and has been mathematically known for more than a hundred years [2, 15, 16]. The offset operator is closed for arcs and lines, i.e. an offset of an arc and a line are an arc and a line, respectively. This is not so for Bézier and NURB curves, in general, so approximations are usually derived.

Two methods for finding approximations to offset curves are commonly used. The first approximates the curve using piecewise lines and arcs and then finds the representation of the exact offset to the approximation. That approach was introduced

*This work was supported in part by DARPA (N00014-88-K-0689). All opinions, findings, conclusions or recommendations expressed in this document are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

[†]Appreciation is expressed to IBM for partial fellowship support of the first author.

in [12] and used successfully in [10]. The second method attempts to approximate the offset by directly transforming the curve representation, in particular the control points [13, 3, 14, 8, 9, 5]. To improve the accuracy of the approximation in the second method, the original curve is subdivided [13, 8, 9] or manually refined [3] when the error is above prespecified tolerance level and the same technique is applied to each of the subdivided pieces. The original curve is usually subdivided in the middle of its parametric domain [13, 8, 9], although that is usually not the optimal location. Curve inflection points have also been considered as splitting points for planar offsets [9].

Both approaches do not bound the offset error globally. In [7] the error of the approximation to the offset is computed as a function, and analyzed to provide a global tolerance bound. Furthermore this error function is also used to automatically identify the regions with larger error. An automatic iterative process is used to refine these regions so that an approximated offset representation is generated whose global error is to within a prespecfied tolerance.

Section 2 develops the necessary background for this paper. In section 3 we show how perturbing control points by analyzing the error function results in better offset approximation. Section 4 provides several results for curves and surfaces. Section 5 concludes.

2 Background

Let $C(t)$ be a planar regular parameterized curve, which without loss of generality, is assumed to be in the $x-y$ plane. An offset curve for $C(t)$ by an amount d is defined mathematically as

$$\hat{\mathcal{C}}_d(t) = C(t) + N(t)d \tag{1}$$

where $N(t)$ is the unit normal to the curve at t.

Similarly for surfaces, an offset surface for surface $S(u,v)$ by an amount d is mathematically defines as

$$\mathcal{S}_d(u,v) = S(u,v) + n(u,v)d \tag{2}$$

where $n(u,v)$ is the surface unit normal to the surface at parameter values (u,v).

In the paper we will concentrate on characterizing methods for the NURBs representation since the Bézier representation is a subset of it. Given two freeform NURBs curves $C_1(t)$ and $C_2(t)$ (surfaces $S_1(u,v)$ and $S_2(u,v)$), their sum, difference and product is also a NURBs curve (surface) [6, 4, 11, 7]. Derivatives of NURBs curves (surfaces) are also NURB curves (surfaces) [4].

Therefore, if $N(t)$ $(n(u,v))$ could be computed and represented as a NURBs, so could $\mathcal{C}_d(t)$ $(\mathcal{S}_d(u,v))$, respectively. Unfortunately, however, the general form of a unil normal requires the square root of a function which is usually not representable as either a polynomial or a piecewise polynomial. Thus, offsets of freeform curves and surfaces used in computation will, in general, be approximations.

Let $\mathcal{C}_d^a(t)$ be an approximation to the offset curve of $C(t)$ by an amount d (equation 1), and let $\delta(t) = \mathcal{C}_d^a(t) - C(t)$ be the difference curve. Ideally, if $\mathcal{C}_d^a(t) \equiv \mathcal{C}_d(t)$, then $\delta(t) \equiv dN(t)$.

As shown in [7] the magnitude of $\delta(t)$ can be efficiently computed to determine the accuracy of $\mathcal{C}_d^a(t)$. Current offset techniques usually test this accuracy by evaluating this magnitude on a presepcified collection of sampled points. Direct representation of $\|\delta(t)\|$ would require the representation of a square root, so instead [7] uses $\psi(t) = \|\delta(t)\|^2$ and compares to d^2.

$$\psi(t) = \|\delta(t)\|^2 = \delta_x(t)^2 + \delta_y(t)^2 + \delta_z(t)^2, \tag{3}$$

where $\delta_x(t)$, $\delta_y(t)$ and $\delta_z(t)$ are the components of $\delta(t)$.

Equation 3 can be represented in closed form using multiplication and addition which are computable for rationals and piecewise rationals. Hereafter, assume that the coefficients of $\psi(t)$ can be computed and $\psi(t)$ can be represented as a scalar NURBs curve. If $\mathcal{C}_d^a(t)$ were exact , ψ would be a constant curve equal to d^2, so by subtracting d^2 from ψ one could find the error difference curve for a particular approximation:

$$\epsilon(t) = \psi(t) - d^2. \tag{4}$$

The extremal values of the coefficients of ϵ provide a global error measure. It is important to examine the consequences for computing $\epsilon(t)$ instead of $\varepsilon(t) = \|\delta(t)\| - d$, the real error between the exact offset curve and its approximation:

$$\epsilon(t) = \psi(t) - d^2 = \|\delta(t)\|^2 - d^2 = (\varepsilon(t) + d)^2 - d^2 = \varepsilon(t)^2 + 2d\varepsilon(t) \approx 2d\varepsilon(t) \tag{5}$$

In other words, by computing the differences of the squared magnitude, the resulting error bound is scaled by the magnitude of twice the offset distance, $2d$, which is a constant and therefore easy to control. $\varepsilon(t)^2$ has been ignored since it is much smaller than $2d\varepsilon(t)$, as the error converges to zero.

The problem of finding the global offset error has been reduced to a problem of finding the extrema of a freeform explicit curve. Since the values of a scalar B-spline curve over an interval lie between the maximum and minimum values of the coefficients of the non-zero B-spline functions, a simple and computationally efficient way of locally bounding the curve is immediately available.

An iterative process in which each step uses the direct polygon transformation method [3] to compute offset approximations is used and the error function, $\epsilon(t)$ is computed for it. the curves (or surface) is then refined at the regions with error larger than can be tolerated and the process repeats itself. The process terminates when the magnitudes of the extrema of $\epsilon(t)$ are within the tolerance.

The above method can be applied equally well to finding errors of offset surfaces. $\delta(t)$, $\psi(t)$ and $\epsilon(t)$ would be simply explicit surfaces, $\delta(u,v)$, $\psi(u,v)$ and $\epsilon(u,v)$, instead of explicit curves.

3 Getting A Better Approximation of Offsets

In [7] a technique was developed to provide a global bound using a global error function and used with refinement to reduce maximum error. Here we show how to reduce the error by using this bound and perturbing control points. For each control point, one could compute the gradient direction that maximize the local change in the error function and move the control point in that direction. However, such a computation would be extremely expensive and slow. By refining and offsetting in the normal direction, it is known that the sequence of approximations to the offset converges to the exact offset [7]. A simple candidate for a preferred direction in which to offset is the normal direction. We also show that perturbing in the normal direction results in an exact representation of offset of circular curves in a rational quadratic representation with no refinement at all. Therefore, the curve or surface normal direction at the node value associated with the respective control point is used as a first order approximation for the gradient to minimize the offset approximation error. The amount that the control point is moved is determined from the amount of error in the offset approximation, as is evaluated from ϵ at that location. The iterative algorithmic process for curves follows:

Algorithm 1

Input:
    τ, required offset curve tolerance.
    $C(t)$, input curve.
    $C_d^a(t)$, offset approximation to input curve.
    d, offset distance.

Output:
    $\hat{C}_d^a(t)$, improved curve approximation.

Algorithm:
    $C_0^a(t) \Leftarrow C_d^a(t)$
    $i \Leftarrow 0$
    $MaxErr \Leftarrow Infinity$
    Do
        Compute offset error $\epsilon(t)$ for $C(t)$, $C_i^a(t)$ (equations 3,4)
        $C_{i+1}(t) \Leftarrow C_i(t)$ perturbed according to $\epsilon(t)$ at node values.
        $LastMaxErr \Leftarrow MaxErr$

$MaxErr \Leftarrow \mathbf{min}(LastMaxErr, \epsilon(t)$ `highest error`$)$
$i \Leftarrow i + 1$
`While` $(MaxErr > \tau$ `and` $MaxErr < LastMaxErr)$

The error function is computed in each iteration and each control point is moved in the normal direction by the offset error amount at the node parameter value associated with this control point. This process repeats itself until no improvement is gained in the maximum error (i.e. no convergence) or the required tolerance is achieved. This iterative algorithm is interleaved with refinement to guarantee convergence [7]. In each step algorithm 1 is invoked. If it improves the approximation to the required tolerance the whole process stops. Otherwise refinement takes place in the regions with high offset error and algorithm 1 is invoked again. Algorithm 1 directly extends to surfaces. The process is exactly the same for surfaces where ϵ is a scalar surface, i.e. $\epsilon(u, v)$, and the control mesh is pertubed in the direction of the surface normal.

In section 4 several examples on the affect of perturbing control points are being provided.

4 Examples

Figure 1 shows a unit circle composed of four 90 degree quadratic arcs. The first offset is obviously underestimated, but it converges quite quickly to the exact offset by moving only the corner points. These points have non zero error, as can be seen from figure 2 which also shows the respective error function as the process converges. The points in the error function in figure 2 in which the error is always zero correspond to the points the circular curve interpolate the control points (the end points of the quadratic Bézier segments). Since the normals for the corner control point node values are in the direction (vectors $(\pm a, \pm a)$) pointing to the corner control points of a larger similarly represented circle, this process converges to an exact offset circle, with no refinement.

In Figure 3 the quadratic curve consists of three arcs of 120 degrees and three lines. The offset error along the line is zero and no improvment is applied there. The arcs can be improved to the exact representation. The required tolerance of 0.01 terminated this process at that accuracy as can be seen in table 1.

Figure 4 is a case in which exact representation of the offset in the NURBs does not exist. Control points perturbation can only improve the result, but refinement is still necessary to meet the required tolerance of 0.04 as can be seen from table 2.

Figure 5 shows the same process applied to a unit sphere. This time the process does not converge to the exact representation since the normals at the node values of the corner points are not in the exact direction (vectors $(\pm a, \pm a, \pm a)$). Even so the improvement gained is quite significant. The right side of figure 5 is the regular offset while the left side shows the same surface (and same number of control points)

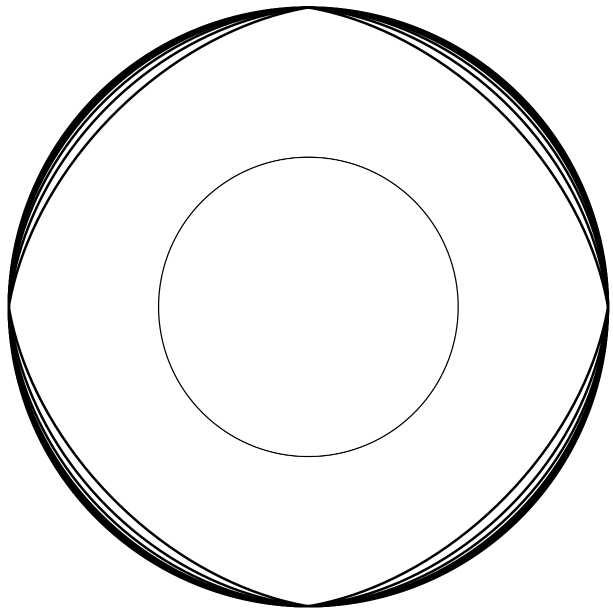

Figure 1: Control points perturbation converges to exact offset circle.

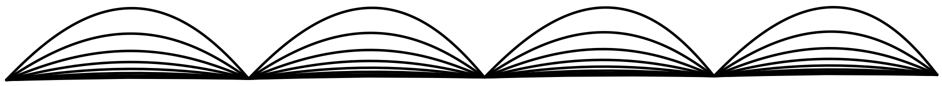

Figure 2: The error function convergence to zero, of the circle in figure 1.

Step	Error	Comments
1	0.49	
2	0.387	
3	0.291	
4	0.211	
5	0.149	
6	0.104	
7	0.071	
8	0.048	
9	0.033	
10	0.022	
11	0.015	
12	< 0.01	Tolerance is met.

Table 1: Errors in convergence of the sphere offset sequence using control points perturbation.

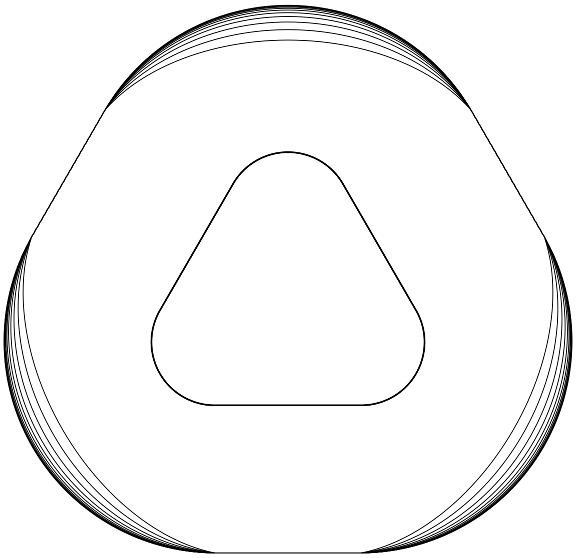

Figure 3: Error function convergence to zero, for three 120 degrees arcs in the curve

Step	Error	Comments
1	0.22	
2	0.197	
3	0.187	
4	0.182	
5	0.179	
⋮	⋮	
16	0.178	No improvement - refinement stage
21	0.083	
22	0.040	
23	< 0.04	Tolerance is met.

Table 2: Errors in convergence of sphere offset using control points perturbation.

after perturbing it. Table 3 provides the convergence steps for this case, up to the prespecified tolerance of 0.01.

5 Conclusions

References

[1] S. Aomura and T. Uehara. Self-intersection of an offset surface. Computer Aided Design, vol. 22, no. 7, pp 417-422, september 1990

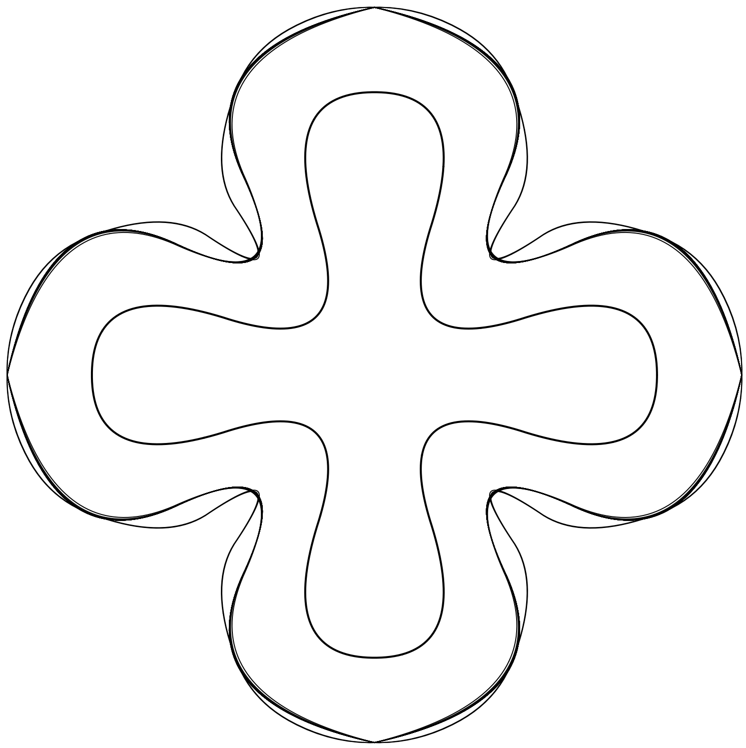

Figure 4: The error function does not convergence to zero, for general curves.

Step	Error	Comments
1	2.574	
2	1.43	
3	0.964	
4	0.804	
5	0.733	
6	0.691	
⋮	⋮	
20	0.616	No improvement - refinement stage
21	0.296	
22	0.238	
23	0.186	
24	0.146	
25	0.115	
26	< 0.01	Tolerance is met.

Table 3: Errors in convergence of sphere offset using control points perturbation.

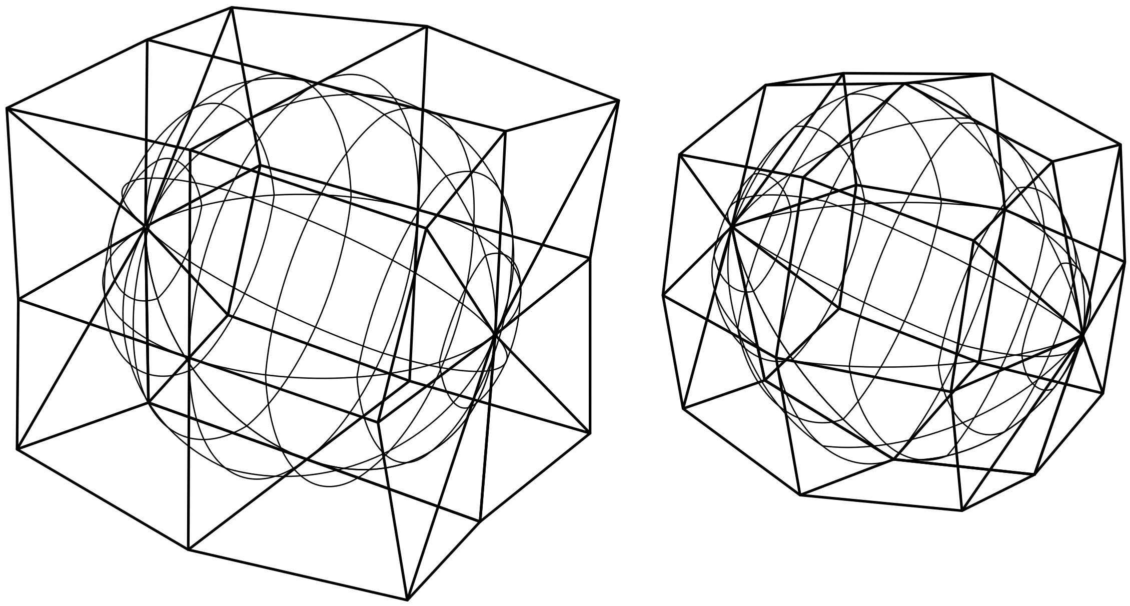

Figure 5: Control points perturbation can also improve offset surface accuracy.

[2] J. Bertrand. La theorie des courbes a double courbure. Journal de Mathematique Pures et Appliquees, 15 (1850), 332-350.

[3] B .Cobb Design of Sculptured Surfaces Using The B-spline representation. Ph.D. thesis, University of Utah, Computer Science Department, June 1984.

[4] G. Farin. Curves and Surfaces for Computer Aided Geometric Design Academic Press, Inc. Second Edition 1990.

[5] R. T. Farouki and C. A. Neff. Some Analytic and Algebraic Properties of Plane Offset Curves. Research Report 14364 (#64329) 1/25/89, IBM Research Division.

[6] R. T. Farouki and V. T. Rajan. Algorithms For Polynomials In Bernstein Form. Computer Aided Geometric Design 5, pp 1-26, 1988.

[7] G. Elber and E. Cohen. Error Bounded Variable Distance Offset Operator for Free form Curves and Surfaces. Technical report No. 91-001, Computer Science, University of Utah.

[8] J. Hoschek. Spline approximation of offset curves. Computer Aided Geometric Design 5 (1988) p. 33-40

[9] J. Hoschek and N .Wissel. Optimal approximate conversion of spline curves and spline approximation of offset curves. Computer Aided Design, vol. 20, no. 8, pp 475-483, october 1988.

[10] J .J. Chou. Numerical Control Milling machine Toolpath Generation for Regions Bounded by Free Form Curves and Surfaces. Ph.D. thesis, University of Utah, Computer Science Department, June 1989.

[11] K. Morken. Some Identities for Products and Degree Raising of Splines. To appear in the journal of Constructive Approximation.

[12] H. Persson. NC machining of arbitrarily shaped pockets. Computer Aided Design, vol. 10, no. 3, pp 170-174, may 1978.

[13] B. Pham. Offset Approximation of Uniform B-splines. Computer Aided Design, vol. 20, no. 8, pp 471-474, october 1988.

[14] S. Coquillart. Computing offset of Bspline curves. Computer Aided Design, vol. 19, no. 6, pp 305-309, july/august 1987.

[15] F. Salkowski. Zur Transformation von Raumkurven. Mathematische Annalen, 66 (1909), 517-557.

[16] A. Voss. Uber Kurvenpaare in Raume Sitzungsberichte, Akadamie der Wissenschaften zu Munchen. 39 (1909), 106.