Floating Point Geometric Algorithms for Topologically Correct Scientific Visualization

E. L. F. Moore, T. J. Peters*

July 28, 2006

Abstract

The unresolved subtleties of floating point computations in geometric modeling become considerably more difficult in animations and scientific visualizations. Some emerging solutions based upon topological considerations will be presented. A novel geometric seeding algorithm for Newton's method was used in experiments to determine feasible support for these visualization applications.

1 Computing the pipe surface radius

Parametric curves have been shown to have a particular neighborhood whose boundary is non-self-intersecting [4]. It has also been shown that specified movements of the curve within this neighborhood preserve the topology of the curve [8, 7], as is desired in visualization. This neighborhood is defined by a single value, which is the radius of a pipe surface, where that radius depends on curvature and the minimum length over those line segments which are normal to the curve at both endpoints of the line segment [4]. The focus of this paper is efficient and accurate floating point techniques to compute that radius.

*Department of Computer Science & Engineering, University of Connecticut, Storrs, CT 06269-2155, tpeters@cse.uconn.edu.

Definition 1.1 *For a non-self-intersecting, parametric curve c, where*

$$c : [0, 1] \to \mathbb{R}^3,$$

and for distinct values $s, t \in [0, 1]$, the line segment $[c(s), c(t)]$ is doubly normal *if it is normal to c at both of the points $c(s)$ and $c(t)$.*

Definition 1.2 *The* global separation *is the minimum over the lengths of all doubly normal segments. (For compact curves, this minimum has been shown in be positive [5].)*

An example cubic B-splines curve is given in Figure 1, with

1. control points: (0.0 0.0 0.0) (-1.0 1.0 0.0) (4.5 5.5 2.0) (5.0 -1.0 8.5) (-1.5 2.5 -4.5) (4.5 6.0 8.5) (3.5 -3.5 0.0) (0.0 0.0 0.0) and
2. knot vector: {0 0 0 0 0.2 0.4 0.6 0.8 1 1 1 1}

For this curve, there exist many doubly normal segments, as shown in Figure 1. The problem is how to efficiently find all these doubly normal segments,

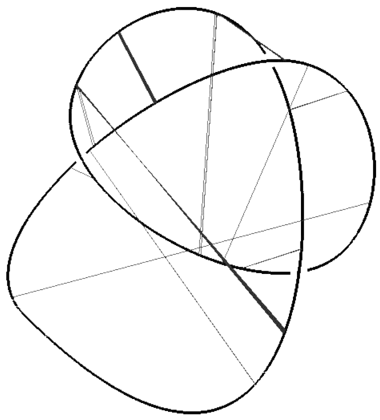

Figure 1: Many doubly normal segments exist on this curve.

and then find the pair which represents the global separation distance. A pair of distinct points at s and t on a parametric curve will be endpoints of a doubly normal segment if they satisfy the two equations [4]:

$$[c(s) - c(t)] \cdot c'(s) = 0 \tag{1}$$

$$[c(s) - c(t)] \cdot c'(t) = 0 \tag{2}$$

where, $s, t \in [0, 1]$

In principle, the system given by Equations 1 and 2 could be solved algebraically by writing them in their power basis form [9]. The power basis represents the curve in the following expression, where $u \in [0, 1]$ and each a_i is a point in $\mathbb{R}^3$.

$$c(u) = (x(u), y(u), z(u)) = \sum_{i=0}^{n} a_i u^i.$$

Despite the straigtforward conceptual appeal of such an algebraic strategy, it is well-known that these power basis forms lead to algorithmic difficulties [9]. Hence, alternative techniques, as will be presented, which rely upon Bézier or spline representations are preferred. The method presented here relies upon first converting a spline curve into its equivalent representation as a composite Bézier curve [9].

For example, the B-spline curve in Figure 1 with knot vector

U={0 0 0 0 .2 .4 .6 .8 1 1 1 1}

is decomposed into its five Bézier segments by inserting interior knots until

U={0 0 0 0 .2 .2 .2 .4 .4 .4 .6 .6 .6 .8 .8 .8 1 1 1 1}

and re-computing the new control polygon, which contains the control points for each Bézier segment with

U={0 0 0 0 1 1 1 1}.

This is shown graphically in Figure 2, where each of the five segments are shown in a different color and labeled 1 through 5. The control points for each of the five segments are as follows:

- curve 1 (yellow): (0 0 0) (-1 1 0) (1.75 3.25 1) (3.2083 3.2917 2.5833)
- curve 2 (cyan): (3.2083 3.2917 2.5833) (4.6667 3.3333 4.1667) (4.8333 1.1667 6.3333) (3.8333 0.6667 5.25)
- curve 3 (magenta): (3.8333 0.6667 5.25) (2.8333 0.1667 4.1667) (0.6667 1.3333 -0.1667) (0.5833 2.5 -0.1667)

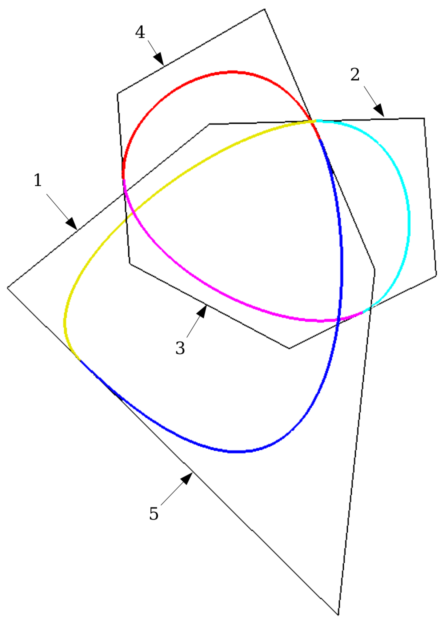

Figure 2: B-spline curve represented by its Bézier segments.

- curve 4 (red): (0.5833 2.5 -0.1667) (0.5 3.6667 -0.1667) (2.5 4.8333 4.1667) (3.25 3.0417 4.2083)
- curve 5 (blue): (3.25 3.0417 4.2083) (4 1.25 4.25) (3.5 -3.5 0) (0 0 0)

A primary motivation for this work is to perform ambient isotopic approximations at speeds that will support scientific visualization of molecular simulations. Towards that goal, Newton's method for two variables is a very efficient and general numerical algorithm [6] that was applied to Equations 1 and 2. The experiments reported on prototype code suggest that this approach could be sufficiently rapid to support scientific visualization. These experiments were performed on a 64-bit AMD processor with Red Hat Linux Fedora Core 2 and OpenGL with double buffering. As always, the integration with a specific graphics subsystem is highly dependent upon the underlying architecture, and incorporation of this code on any platform would require further development and experimentation.

The application of Newton's method for equations 1 and 2 follows. Compute

$$\begin{bmatrix} s_{n+1} \\ t_{n+1} \end{bmatrix} = \begin{bmatrix} s_n \\ t_n \end{bmatrix} - J^{-1}(s_n, t_n) \begin{bmatrix} f(s_n, t_n) \\ g(s_n, t_n) \end{bmatrix}, n = 0, 1, ... \tag{3}$$

until $|J^{-1}(s_n, t_n)[f(s_n, t_n)\ g(s_n, t_n)]^T|$ is less than some $\epsilon > 0$, where $J^{-1}(s_n, t_n)$ is the inverse Jacobian matrix. Hence,

$$J(s_n, t_n) = \begin{bmatrix} \frac{\partial f}{\partial s} & \frac{\partial f}{\partial t} \\ \frac{\partial g}{\partial s} & \frac{\partial g}{\partial t} \end{bmatrix} \tag{4}$$

$$J^{-1}(s_n, t_n) = \frac{1}{\frac{\partial f}{\partial s}\frac{\partial g}{\partial t} - \frac{\partial f}{\partial t}\frac{\partial g}{\partial s}} \begin{bmatrix} \frac{\partial g}{\partial t} & -\frac{\partial f}{\partial t} \\ -\frac{\partial g}{\partial s} & \frac{\partial f}{\partial s} \end{bmatrix} \tag{5}$$

and, from equations 1 and 2 above, (with f being given by the left-hand side of Equation 1 and g being given by the left-hand side of Equation 2),

$$\frac{\partial f}{\partial s} = [c(s) - c(t)] \cdot c''(s) + c'(s) \cdot c'(s) \tag{6}$$

$$\frac{\partial f}{\partial t} = -c'(t) \cdot c'(s) \tag{7}$$

$$\frac{\partial g}{\partial s} = c'(s) \cdot c'(t) \tag{8}$$

$$\frac{\partial g}{\partial t} = [c(s) - c(t)] \cdot c''(t) + c'(t) \cdot -c'(t) \tag{9}$$

As is typical, the ‘art’ required for the successful use of Newton’s method is highly dependent upon the determination of reasonable initial estimates. Any such estimates are clearly data dependent, but a viable approach is presented and verified on an illustrative example. The general idea is to generate a modest number of line segments, each connecting two distinct points on the curve. There is no *a priori* reason to expect that any such segment will be doubly normal and, indeed, many of them would be far from meeting that criterion. Fortunately, many of these failures can be excluded from further consideration by an easy culling technique based upon a lack of normality at one end point or the other. This culling is accomplished in two steps, with respective positive values of ϵ_1 and ϵ_2.

Let p_1 and p_2 be the endpoints of one of the generated segments and let $< p_1, p_2 >$ denote the vector of unit length, formed by taking the vector between p_1 and p_2 and dividing that vector by its norm. The first test is to determine if the unit vector $< p_1, p_2 >$ is normal to the curve c at p_1. Suppose $p_1 = c(s_1)$ for some $s_1 \in [0, 1]$ and then comparison is made as to whether

$$< p_1, p_2 > \cdot\ c'(s_1) < \epsilon_1, \tag{10}$$

where $c'(s_1)$ is the unit tangent vector at $c(s_1)$.

If the result of the preceding comparison is false, then this segment is not sufficiently close to being normal to the curve to warrant further consideration and it is excluded. Otherwise, it is retained for a similar culling comparison at the other endpoint. Specifically, this second test is to determine if the dot product of the vector $< p_2, p_1 >$ with the unit tangent vector to the curve c, where $p_2 = c(s_2)$ for some $s_2 \in [0, 1]$ is approximately zero. Namely, the comparison is made as to whether

$$< p_2, p_1 > \cdot\ c'(s_2) < \epsilon_2. \tag{11}$$

If the result of the preceding comparison is false, then this segment is rejected. Otherwise, it is sufficiently close to being doubly normal to serve as an initial estimate for Newton's method. In the ensuing algorithmic description, the use of Inequalities 10 and 11 will be called the 'doubly normal test'.

Values of $\epsilon_1 = 0.001$ and $\epsilon_2 = 0.01$ were found experimentally to give the best results for the curve depicted in Figure 2 when a large number of parametric values (i.e. $n = 2000$) were chosen. Values smaller than these would result in critical doubly normal segments not being found. Values greater than these would result in too many double normal segments being found. Similarly, values of $\epsilon_1 = \epsilon_2 = 0.4$ were found to give good results when a small number of parametric values (i.e. $n = 10$) were chosen for computing candidate points for input to Newton's method, as discussed below.

Let c be a composite C^2 Bézier curve composed of sub-curves

$$c_i, \text{ where } i = 1, \ldots, m \text{ for some positive integer } m.$$

It should be clear that the analysis for doubly normal segments should be conducted on all pairs of sub-curves, c_i, c_j, where $i, j \in \{1, \ldots, m\}$, *inclusive* of the case where $i = j$. When $i = j$, care must be taken to ensure that there are distinct parameter values. The principles should be sufficiently clear from this discussion to now summarize them in an algorithm developed to provide initial estimates for Newton's method. This approach was found to be quite efficient by experimentation. Furthermore, the accuracy was quite acceptable, as verified by an independent (but much slower) computation that is summarized later. Now, the initial estimate algorithm is presented. In practice, a value of $n = 10$ for the number of parametric values chosen, yielded accurate and efficient results.

```
Initial Estimate Algorithm for Newton's Method

Input: $n$, and sub-curves $c_i$, where $i = 1, \ldots, m$
   1. For $k = 1, \ldots, n-1$
      s = 1/n + (k - 1)/n.
   2. For $\ell = 1, \ldots, n-1$
      t = 1/n + ($\ell$ - 1)/n.
        3. For $i = 1, \ldots, m$
          4. For $j = 1, \ldots, m$
            5. If $i = j$ and $s \neq t$, cull versus doubly normal test.
            6. If $i \neq j$, cull versus doubly normal test.
Output: all segments not culled.
```

Figure 3: Initial Estimates for Newton's Method

It is notable that the parametric values chosen deliberately avoided the end-points of each sub-curve, as these points may not meet the differentiability assumptions required for Newton's method. However, it is trivial to explicitly test all the possible cases based upon these sub-curve endpoints. This is an easy, but required, step to ensure that a minimal length doubly normal segment is *not* precluded from consideration.

The composite Bézier curve shown in Figure 1 was used as an example input to perform these calculations. Each of the five cubic Bézier segments in Figure 2 are shown labeled 1 through 5 with their corresponding control polygons. The above Newton iteration was performed for all pairs of Bézier segments, c_i and c_j, $i = j = 1, ..., 5$ with $\epsilon = 0.0001$, and $\epsilon_1 = \epsilon_2 = 0.4$. (Note: ϵ is defined after Equation 3, while ϵ_1 and ϵ_2 are defined, respectively, in Inequalities 10 and 11.) The points on the B-spline at the endpoint of a doubly normal segment having minimal length occur at $s = 0.4774$ on Bézier segment 1 and $t = 0.7845$ on Bézier segment 3. The distance between these two points is 0.4427, which is the global separation for this curve. The candidate double normal points for Newton's method were found by computing points on each Bézier segment at parameter values,

$$0.1, 0.2, \ldots, 0.9,$$

where the endpoints at parameter values 0.0 and 1.0 were given special consideration as discussed previously. These candidate double normal points are shown graphically in Figure 4 with line segments connecting the pairs of candidate points for each pair of Bézier segments. Pairs of Bézier segments that do not have a connecting line segment, mean that no candidate points were found. Pairs of Bézier segments that have only one connecting line segment, mean that Newton's method did not converge for those particular points. Pairs of Bézier segments that have two pairs of connecting line segments mean that Newton's method did converge, and the resulting pair of minimum double normal points is one of the two line segments from each pair. Typically, convergence with $\epsilon = 0.0001$ occured after 3 or 4 iterations. Note that Figure 4 depicts the same curve as in Figure 1 and 2, however, the curve is rotated about the y-axis to get a better view of doubly normal points, with the global separation distance illustrated in the zoomed-in section of Figure refnewtons-method-color.

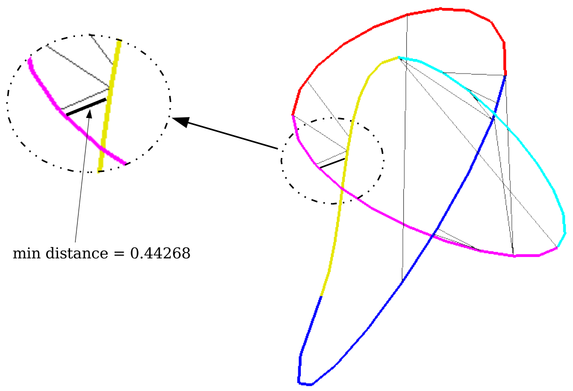

Figure 4: Newton's method.

A conceptually easier method for finding approximate solutions for s and t is to test for mutually perpendicular normals between a very large number of pairs of points. This is reflective of an exhaustive method to simply compute the dot product in equations 1 and 2 for all pairs of points. Although this approach is computationally inefficient, it serves as an independent verification of the accuracy of the values returned by Newton's method. To do so, the "Initial Estimate Algorithm for Newton's Method" would be run with a very large value of n, experimentally chosen to be 2,000. This results in consideration of

$$(2000)(1999)(5)+(2000)(2000)(4+3+2+1) = 1.999\times10^7+4\times10^7 \approx 60 million$$

tests for doubly normal segments. These nearly 60 million computations have performance that is not acceptable for the intended graphics applications. However, even with this slow performance, it serves as an 'off-line' check of the accuracy of the value computed via Newton's method. A comparison of the methods used to compute the global separation is shown in Table 1. Note that tests 1 through 4 used a large value of n to compute the global separation with the Initial Estimate Algorithm (IEA) for Newton's method. Test number 2 shows that $n = 2000$ is sufficient to compute the global separation of 0.4427 based on the result of test number 1 with $n = 10000$. Comparing test number 3 to test 2 shows the importance of carefully choosing the values of ϵ_1 and ϵ_2, since test number 3 does not compute the correct value for the global separation. Similarly, test 4 shows that $n = 1000$ is not sufficient to compute the correct global separation. Test number 5 shows the benefit of using Newton's method, since the computation time for computing the global separation was found to be negligible.

Test #	Method	Partition Size, n	eps1	eps2	Time(s)	Global Sep
1	IEA	10,000	0.001	0.01	85	0.44268
2	IEA	2,000	0.001	0.01	6	0.44268
3	IEA	2,000	0.001	0.001	6	0.81230
4	IEA	1,000	0.001	0.01	2	0.91921
5	Newton	10	0.4	0.4	Negligible	0.44268

Table 1: Comparison of Numerical Methods

As a verification of the Newton's code produced, the global separation for this experimental curve was implemented independently [2]. In this corroborating implementation, the culling technique to get starting values for Newton's method was implemented similarly, with the minor differences that

- values of ϵ_1 and ϵ_2 were chosen to both be equal to 0.1, and

- the starting values retained after the previous search were heap-sorted to use the shorter candidate doubly normal segments.

It is worthwhile to note that

- the final minimum distance obtained was verified to be 0.44268, and
- this method found this minimum to occur for two different pairs of points in close proximity to one another.

The agreement regarding 0.44268 can likely be attributed to the robustness of Newton's method, where that is more dominant than minor differences in coding or starting values. The other minor differences remain as small issues to consider relative to optimizing performance.

Regarding the combinatorial complexity incurred by breaking the spline curve into m Bézier segments, it is clear that this results in $O(m^2)$ computations of Newton's method. Usually, m will be small, so the quadratic complexity will not be a significant concern. However, much of this work is directed towards supporting scientific visualization for high performance computing (HPC) architectures. Then, it is obvious that the $O(m^2)$ calculations could be conducted in parallel. These HPC platforms will often be massively parallel, where the primary platform under consideration can have as many as 10,000 nodes [1, 3]. Hence, the combinatorial complexity incurred is not expected to be prohibitive.

Acknowledgements: The authors were partially supported by NSF grants DMS-9985802, DMS-0138098, CCR-022654, CCR-0429477 and by an IBM Faculty Award. All statements here are the responsibility of the authors, *not* of the National Science Foundation *nor* of IBM. Both authors thank the Dagstuhl Seminar organizers and the Dagstuhl staff for providing the intellectually stimulating environment for refinement of these ideas, which were initially based upon the dissertation of E. L. F. Moore.

The helpful comments on an anonymous referee, are acknowledged with appreciation. In particular, an alternative proposed to exploit the parallelism possible using normal planes and culling is of interest but beyond the scope of this paper.

References

[1] G. S. Almasi, C. Cascaval, J. G. Castanos, W. D. M. Denneau, M. Eleftheriou, M. Giampapa, D. L. H. Ho, J. E. Moreira, D. Newns, M. Snir, and H. S. Warren. Computational topology for reconstruction of surfaces with boundary: integrating experiments and theory. In M. Spagnuolo, A. Pasko, and A. Belyaev, editors, *International Conference of Shape Modeling and Applications*, pages 288–297, Los Alamitos, CA, June 13 - 17, 2005 2005. IEEE, IEEE Computer Society.

[2] J. Bisceglio. Personal communication. justin.bisceglio@gmail.com, October 2005. UConn department of CS&E alumni, Computer graphics engineer at Blue Sky Studios.

[3] R. S. Germain, B. Fitch, A. Rayshubskiy, M. Eleftheriou, M. C. Pitman, F. Suits, M. Giampapa, and T. J. C. Ward. Biochips and bioinformatics: Blue matter on blue gene/l: massively parallel computation for biomolecular simulation. In *Proceedings of the 3rd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis CODES+ISSS*, September 2005.

[4] T. Maekawa and N. M. Patrikalakis. *Shape Interrogation for Computer Aided Design and Manufacturing.* Springer, New York, 2002.

[5] T. Maekawa, N. M. Patrikalakis, T. Sakkalis, and G. Yu. Analysis and applications of pipe surfaces. *Computer Aided Geometric Design*, 15:437–458, 1998.

[6] J. H. Mathews. *Numerical Methods for Computer Science, Engineering and Mathematics.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1987.

[7] E. L. F. Moore. *Computational Topology of Spline Curves for Geometric and Molecular Approximations.* PhD thesis, The University of Connecticut, 2006.

[8] E. L. F. Moore, T. J. Peters, and J. A. Roulier. Preserving computatational topology by subdivision of quadratic and cubic bézier curves. *dagstuhl*, to appear.

[9] L. Piegl and W. Tiller. *The NURBS Book.* Springer, 2nd edition, 1997.