CSE 557 Final Project: A C^2 Interpolating Parametric Curve With Local Control

Joel Hindorff

March 16, 2000

This document outlines the derivation of a novel C^2 interpolating parametric curve with local support in the control points. Nuts and bolts computations are omitted from this exposition, as are explicit computational results. This text is supplied only to amplify and clarify the projects achievements, as well as to motivate their proofs and illuminate the overall approach. Explicit computations are available in the computational manual.

1 A Simple Example

Consider a generic parametric segment curve

$$c(w) = \lambda(w)p(w) + \mu(w)q(w)$$

with $w \in [0,1]$ along this segment. In this general formulation, $\lambda(w)$ and $\mu(w)$ are regarded as unspecified polynomial weighting functions applied to unknown polynomial interpolants $p(w)$ and $q(w)$. We base the following derivation on this general notion of blending interpolants in parameter space. We will motivate the computation of $\lambda(w)$, $\mu(w)$, $p(w)$, and $q(w)$ by considering the simplest example.

Suppose that we have only two control points, $P1$ and $P2$, and want to find the parametric curve $c(w)$ that interpolates this data in a C^2 manner. We seek two polynomials that overlap between $P1$ and $P2$, and the simplest choices are found by setting $p(w) = P1$ and $q(w) = P2$. These $0th$ order polynomials clearly fulfill the data interpolation requirement, and lead to the representation

$$c(w) = \lambda(w)P1 + \mu(w)P2$$

Since any candidate weighting functions must also preserve the interpolation requirement, we must enforce the following conditions:

$$\lambda(0) = 1, \lambda(1) = 0, \mu(0) = 0, \mu(0) = 1.$$

These requirements suggest a classical linear interpolation, which in turn leads to

$$c(w) = (1 - w)P1 + wP2$$

Our current curve segment already represents local control point support and interpolation, which leads us to consider the question of continuity. Since a single segment comprised of simple sums and products of polynomials is C^∞ continuous, the real continuity test must be performed across polynomial segments. As a check on continuity, we compute

$$c'(w) = -P1 + P2$$

$$c'(w) = 0$$

for the current segment. Given any multiplicity of control points, we compute parametric segments in this manner by first mapping segment defining pairs onto the unit interval parameterized by w. This is achieved by normalizing the distance traveled along the chord length of a given interval by the total length. A piecewise parametric curve defined by these individual parametric segments has a derivative at any point based on the parametric derivative normalized by this chord length (i.e. the chain rule). The point of this digression is to assert that, a general aggregate curve will fail to be even $C1$ continuous because of the form of $C'(t) \propto c_i'(w)$. This is obvious since we've merely performed a linearly interpolation between data points. However, these observations made explicit now motivate a means of achieving C^2 continuity in the general case.

On initial inspection it seems that we need a more sophisticated polynomial interpolant than our $0th$ order constant interpolants. This intuition is misguided however since the polynomial interpolant is C^∞ continuous! The fault must therefore rest with the choice of weighting functions $\lambda(w)$ and $\mu(w)$. Reconsider the general parameterization of a segment:

$$c(w) = \lambda(w)p(w) + \mu(w)q(w)$$

and it's derivatives

$$c'(w) = \lambda(w)'p(w) + \lambda(w)p(w)' + \mu(w)'q(w) + \mu(w)q(w)'$$

$$c''(w) = \lambda(w)''p(w) + 2\lambda(w)'p(w)' + \lambda(w)p(w)'' + \mu(w)''q(w) + 2\mu(w)'q(w)' + \mu(w)q(t)''$$

Previously, we chose $\lambda(w)$ and $\mu(w)$ to satisfy the interpolation requirement which gained us C^0 continuity. Since this selection failed to deliver C^2 continuity across adjacent segments, we will now incorporate first and second derivative information between segments into our selection.

Clearly, the requirement is that $C(t)$, $C'(t)$, and $C''(t)$ for the piecewise polynomial agree across parametric segments. A simple way of achieving this is to require:

$$c(0) = p(0), c'(0) = p'(0), c''(0) = c''(0)$$

and

$$c(1) = q(1), c'(1) = q'(1), c''(1) = q''(1).$$

In other words, the properties of $c(w)$ are completely determined by the "left polynomial" at the left end point and the "right" polynomial at the right end point. This allows the curve to behave "arbitrarily" inside a curve segment, but force a pleasant C^2 taming of the curve at its endpoints. Since the "right" polynomial $p(w)$ of one segment is constructively the "left" polynomial $q(w)$ for the subsequent segment, we are doing a kind of boundary condition matching at the end points. Though any arbitrary behavior could be required on these boundaries, our requirement seems most natural. These conditions on $c(w)$ imply

$$\lambda(0) = 1, \lambda(1) = 0, \mu(0) = 0, \mu(0) = 1,$$
$$\lambda'(0) = 0, \lambda'(1) = 0, \mu'(0) = 0, \mu'(0) = 0,$$

and

$$\lambda''(0) = 0, \lambda''(1) = 0, \mu''(0) = 0, \mu''(0) = 0.$$

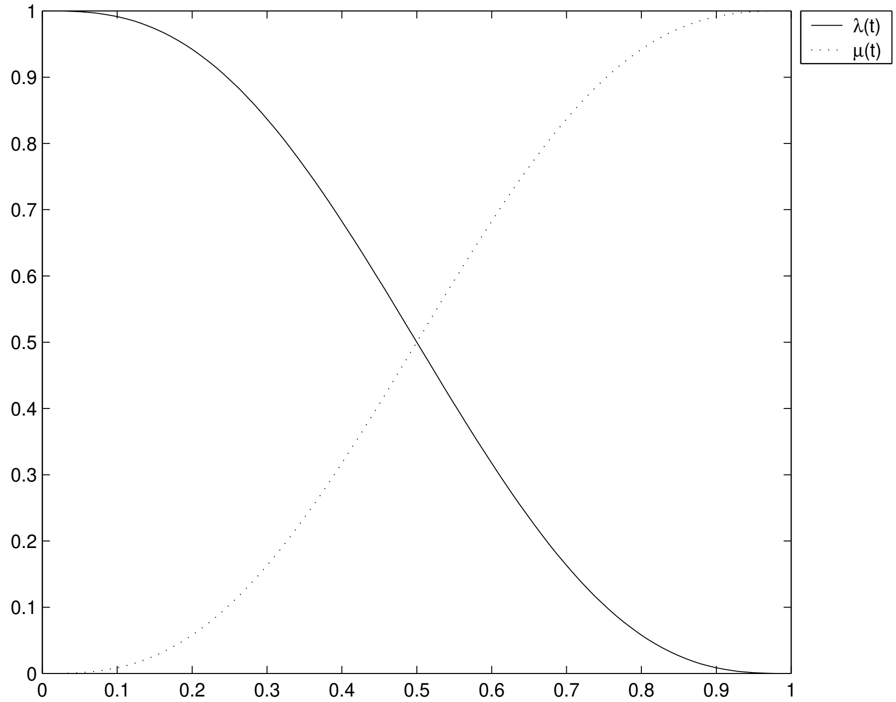
Assuming that $\lambda(w)$ and $\mu(w)$ are polynomial, these conditions determine $5th$ order polynomials uniquely determined by linear systems of 6 equations each. This computation is omitted (it involves a simple Vandermonde-esque matrix inversion), but the results is

$$\lambda(w) = -6w^5 + 15w^4 - 10w^3 + 1$$

and

$$\mu(w) = 6w^5 - 15w^4 + 10w^3 \equiv 1 - \lambda(w)$$

These curves are depicted below.



An immediate consequence of this choice of weighting functions is that they sum to unity. This is crucial for maintaining the integrity of the underlying polynomial interpolants $p(t)$ and $q(t)$.

We observe that under this simplest formulation we have a parametric curve that satisfies all of the required functional properties, despite using $0th$ order polynomial to interpolate the data! The C^2 continuity is secured via the blending functions. Interpolation is earned by the interpolants and maintained by the weighting functions. Local control is afforded since any control point influences only a left and right curve segment. This appears to be a significant achievement gained through trivial exertion. However, this curve has some very unattractive properties. Since the parametric curve is constructed such that it's value, derivative, and second derivate are equivalent to those of an interpolating polynomial at each control point, each control point in the above system corresponds to an inflection point on the curve. In essence, we're attempting to use properties that don't meaningfully exist. These spurious artifacts are compounded by the fact that the curve has the dynamics of a $5th$ order polynomial in between these inflection points. Obviously, these leads to a curve that essentially does nothing, but manages to do it very aggressively.

One obvious path to solving these problems is to select a polynomial that has non-degenerate higher order properties. We adopt quadratics because they are of relatively low order, cheap to compute, have nontrivial C^2 behavior, and can produce cubic-like results when two parabolas are blended across four control points. Though it may be possible to attain the C^2 continuity using a higher order spline interpolant, this would necessitate additional control points in the definition of a single curve segment. Thus, an additional novel feature of the following technique is that it attains these properties with the same number of control points as in the cubic spline case. We now propose a formal outline of the C^2 continuous blended quadratic spline.

2 Quadratic Based Interpolation

We regard a control point to be a coordinate pair (t, y) where t is some specified world-time value and y is the corresponding value of a given parameter at time t. We will construct a piecewise defined polynomial across $n+3$ such control points, (t_i, y_i) for $i = 1 \ldots n+3$. We observe that the $n+3$ control points together define $n + 2$ time intervals. Our approach within each interval is to construct a local polynomial based on the two control points defining that interval along with two corresponding neighbors that immediately bracket those control points. Since we require two interior and two exterior control points to define a polynomial inside a given interval, the first and last intervals will not be parameterized since there is insufficient information available. Plausible solutions to this problem are to ignore it, fit single quadratics to the end segments, or replicate the end control points to achieve the same effect. Our task is to construct the remaining n polynomial curve segments $c_i(t)$, $i = 1 \ldots n$, which together formulate the piecewise polynomial

$$C(t) = c_i(t) \text{ for } t_{i+1} \leq t \leq t_{i+2}$$

We now focus on the computation of the individual $c_i(t)$. We define each parametric curve segment by

$$c_i(w) = \lambda(w)p(w) + \mu(w)q(w)$$

where we have parameterized along the chord length onto $w \in [0,1]$ as described earlier. Note that $\lambda(w)$ and $\mu(w)$ are as before. Since $w(t)$ and $w(t)$ are assumed to be quadratics, we turn our attention to the computation of a general quadratic model. The general problem of resolving a parabola $p(t)$ with data (t_1, f_1), (t_2, f_2), (t_3, f_3) can be solved any number of ways. A relatively obscure but highly elegant approach to this problem is to map the t values onto an w parameterization of the unit interval such that $t_1 \to 0$, $t_2 \to \hat{w}$, and $t_3 \to 1$. Our new problem is to compute a quadratic $\hat{f}(w)$ satisfying $\hat{f}(0) = f_1$, $\hat{f}(\hat{w}) = f_2$, $\hat{f}(1) = f_3$. This new problem has a matrix based solution of the form

$$\hat{f}(w) = \begin{bmatrix} w^2 & w & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\hat{w}} & \frac{-1}{\hat{w}(1-\hat{w})} & \frac{1}{1-\hat{w}} \\ \frac{-(1+\hat{w})}{\hat{w}} & \frac{1}{\hat{w}(1-\hat{w})} & \frac{-\hat{w}}{1-\hat{w}} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}$$

Under this method, we can compute the full polynomial segment for any data interval. Given four control point (t_1, y_1), (t_2, y_2), (t_3, y_3), and (t_4, y_4), we first compute a 0 to 1 parameterization of the line segment by

$$w = \frac{t_3 - t}{t_3 - t_2}$$

Next, we compute the unit parameterized quadratics $p(w)$ and $q(w)$ by defining

$$\alpha = \frac{t_2 - t_1}{t_3 - t_1}$$

and

$$\beta = \frac{t_3 - t_2}{t_4 - t_2}$$

with yield

$$p(w) = \begin{bmatrix} w^2 & w & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\alpha} & \frac{-1}{\alpha(1-\alpha)} & \frac{1}{1-\alpha} \\ \frac{-(1+\alpha)}{\alpha} & \frac{1}{\alpha(1-\alpha)} & \frac{-\alpha}{1-\alpha} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

and

$$q(w) = \begin{bmatrix} w^2 & w & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\beta} & \frac{-1}{\beta(1-\beta)} & \frac{1}{1-\beta} \\ \frac{-(1+\beta)}{\beta} & \frac{1}{\beta(1-\beta)} & \frac{-\beta}{1-\beta} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Since we're only interested in the portions of these curves that lie within the current interval, the parameterization in the original parameter w is too aggressive. If instead we define $r = (1 - \alpha)w + \alpha$ and $s = \beta w$ and evaluate $p(r)$ and $q(s)$, we trace a path from t_2 to t_3 in both $p(r)$ and $q(s)$ as w traverses from 0 to 1. This localizes the portion of interest in each quadratic.

By this point our work is actually finished. Given any position $t \in [t2, t3]$, we can produce a sample w of the unit interval and then evaluate $c_i(w) = \lambda(w)p(r)+\mu(w)q(s)$. We observe that this polynomial segment is $7th$ order since it involves the sums of products of quadratics and quintics. A computational appendix gives the polynomial form as well as it's derivatives.

3 Understanding the Curve

We now justify the claims that the aggregate curve $C(t)$ based on these individual c_i is C^2 continuous, interpolates, and affords local control over the behavior of the curve. We address these issues in reverse order. A constructive argument supports the claim of local control. Any control point participates in three parabolas: the one for which it is one end point, the one for which it is the opposite end point, and the one for which it is the central control point. The parabola for which a control point is central will influence the curve segments defined directly adjacent to it. By a similar argument, the parabolas for which a control point is an end point must also influence two curve segments (determined by the center control point), one of which would be already counted and one of which would not. Since we have two such cases, a control point has influence over four curve segments, the two on either side of it. Outside of this domain, curve segments are defined by other parabolas not interpolating this control point. We have already demonstrated how our choice of weighting function preserves the data interpolation property in the "simple example". The argument is no different for these quadratics.

C^2 continuity is also quite trivial to show. The task amounts to demonstrating that

$$c_i(t_{i+2}) = c_{i+1}(t_{i+2})$$
$$c_i'(t_{i+2}) = c_{i+1}'(t_{i+2})$$
$$c_i''(t_{i+2}) = c_{i+1}''(t_{i+2})$$

We emphasize that t_{t+2} is the right end point of the segment over which $c_i(t)$ is defined and that t_{t+2} is the left end point of the segment over which $c_{i+1}(t)$ is defined. Since these curves are interpolating, C^0 continuity holds immediately. To show C^1 and C^2 continuity, we recall that the "right" quadratic for a set of control points is ostensibly identical to the "left" control point of the subsequent set of control points since they are both defined with the same three control points. Though these respective representations of this curve are distinct in that they are parameterized locally, this distinction is accounted for via the chain rule derivative computation. In essence these curves are squashed (or stretched) during re-parameterization, but the true behavior of the curve is restored by accounting for the derivative of the transformation. The computational manual has full details.