Bezier curves

Michael S. Floater

August 25, 2011

These notes provide an introduction to Bezier curves.

1 Bernstein polynomials

Recall that a real polynomial of a real variable $x \in \mathbb{R}$, with degree $\leq n$, is a function of the form

$$p(x) = a_0 + a_1 x + \cdots + a_n x^n = \sum_{i=0}^{n} a_i x^i, \qquad a_i \in \mathbb{R}.$$

We will denote by π_n the linear (vector) space of all such polynomials. The actual degree of p is the largest i for which a_i is non-zero. The functions $1, x, \ldots, x^n$ form a *basis* for π_n, known as the *monomial basis*, and the *dimension* of the space π_n is therefore $n+1$.

Bernstein polynomials are an alternative basis for π_n, and are used to construct Bezier curves. The i-th Bernstein polynomial of degree n is

$$B_i^n(x) = \binom{n}{i} x^i (1-x)^{n-i}, \tag{1}$$

where $0 \leq i \leq n$ and

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}.$$

The first few examples are

$$B_0^0(x) = 1,$$
$$B_0^1(x) = 1 - x, \quad B_1^1(x) = x,$$

$$B_0^2(x) = (1-x)^2, \quad B_1^2(x) = 2x(1-x), \quad B_2^2(x) = x^2,$$

$$B_0^3(x) = (1-x)^3, \quad B_1^3(x) = 3x(1-x)^2, \quad B_2^3(x) = 3x^2(1-x), \quad B_3^3(x) = x^3.$$

These polynomials are defined for all $x \in \mathbb{R}$, but are usually restricted to $x \in [0,1]$ in practice. They have various important properties. They are *linearly independent*, for if

$$\sum_{i=0}^{n} c_i x^i (1-x)^{n-i} = 0, \qquad x \in (0,1),$$

then, by dividing by $(1-x)^n$ and letting $y = x/(1-x)$, we see that

$$\sum_{i=0}^{n} c_i y^i = 0, \qquad y > 0,$$

which implies that $c_0 = c_1 = \cdots = c_n = 0$. Since there are $n+1$ Bernstein polynomials of degree n, they do indeed form a basis for π_n. They are *symmetric* in the sense that

$$B_i^n(x) = B_{n-i}^n(1-x).$$

They are positive for x in the open interval $(0,1)$ and at the endpoints,

$$B_i^n(0) = \begin{cases} 1 & i = 0; \\ 0 & i = 1, \ldots, n, \end{cases} \quad \text{and} \quad B_i^n(1) = \begin{cases} 0 & i = 0, \ldots, n-1; \\ 1 & i = n. \end{cases} \tag{2}$$

They form a *partition of unity*: by the Binomial theorem,

$$\sum_{i=0}^{n} B_i^n(x) = (x + (1-x))^n = 1^n = 1.$$

They satisfy the *recursion formula*,

$$B_i^n(x) = x B_{i-1}^{n-1}(x) + (1-x) B_i^{n-1}(x), \tag{3}$$

which follows from the definition (1) and the binomial identity

$$\binom{n}{i} = \binom{n-1}{i-1} + \binom{n-1}{i}.$$

In (3) and elsewhere we define

$$B^n_{-1} = B^n_{n+1} = 0.$$

The computation of the Bernstein polynomials up to degree n can be arranged in the following triangular scheme, with each column being computed from the previous column, starting from the left:

$$\begin{array}{ccccccc} 1 & = & B_0^0 & B_0^1 & B_0^2 & \cdots & B_0^n \\ & & & B_1^1 & B_1^2 & \cdots & B_1^n \\ & & & & B_2^2 & \cdots & B_2^n \\ & & & & & \ddots & \vdots \\ & & & & & & B_n^n \end{array}$$

2 Bezier curves

Since the $n+1$ Bernstein polynomials of degree n form a basis for π_n, every polynomial p in π_n can be represented in *Bernstein form*, i.e, as

$$p(x) = \sum_{i=0}^{n} c_i B_i^n(x),$$

for some coefficients $c_i \in \mathbb{R}$. When modelling geometry in some Euclidean space $\mathbb{R}^d$ we often model a curve, or part of a curve, as a *parametric* polynomial,

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{a}_i t^i, \qquad \mathbf{a}_i \in \mathbb{R}^d. \tag{4}$$

In practice the Euclidean space will often be $\mathbb{R}^2$ or $\mathbb{R}^3$. Such a curve also has a Bernstein representation,

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{c}_i B_i^n(t), \qquad \mathbf{c}_i \in \mathbb{R}^d. \tag{5}$$

A polynomial curve expressed in this form is known as a *Bezier curve* and the points $\mathbf{c}_i$ are known as the *control points* of $\mathbf{p}$. The curve is usually restricted to the parameter domain (parameter interval) $[0,1]$, in which case $\mathbf{p}$ is a parametric curve $\mathbf{p} : [0,1] \to \mathbb{R}^d$. The polygon formed by connecting the sequence of control points $\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_n$ is known as the *control polygon* of $\mathbf{p}$. To a large extent the shape of a Bezier curve relects the shape of its control polygon, which is why it is a popular choice for designing geometry in an interactive graphical environment. As the user moves the control points interactively, the shape of the Bezier curve tends to change in an intuitive and predictable way.

Various properties of Bezier curves follow from properties of the Bernstein polynomials, for example symmetry:

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{c}_{n-i} B_i^n(1-t).$$

From (2), we obtain the *endpoint property* of Bezier curves,

$$\mathbf{p}(0) = \mathbf{c}_0, \qquad \mathbf{p}(1) = \mathbf{c}_n.$$

Since the Bernstein polynomials sum to one, every point $\mathbf{p}(t)$ is an *affine combination* of the control points $\mathbf{c}_0, \ldots, \mathbf{c}_n$. From this it follows that Bezier curves are *affinely invariant*, i.e., if Φ is an affine map in $\mathbb{R}^d$ then the mapped curve $\Phi(\mathbf{p})$ has control points $\Phi(\mathbf{c}_i)$. Since the Bernstein polynomials are non-negative in $[0, 1]$, every point $\mathbf{p}(t)$ is a *convex combination* of the control points $\mathbf{c}_0, \ldots, \mathbf{c}_n$, and therefore, the Bezier curve $\mathbf{p}$ (restricted to the parameter domain $t \in [0, 1]$) lies in the convex hull of its control points. By treating each of the d coordinates of $\mathbf{p}$ separately, a similar reasoning shows that $\mathbf{p}$ also lies in the *bounding box*

$$[a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_d, b_d],$$

where, if the point $\mathbf{c}_i$ has coordinates $c_{i1}, \ldots, c_{id}$,

$$a_k = \min_{0 \le i \le n} c_{ik} \quad \text{and} \quad b_k = \max_{0 \le i \le n} c_{ik}, \qquad k = 1, \ldots, d.$$

Bounding boxes are used in various algorithms, and are easier to compute than convex hulls.

3 The de Casteljau algorithm

One way of computing a point $\mathbf{p}(t)$ of the Bezier curve $\mathbf{p}$ is first to evaluate the Bernstein polynomials B_i^n at the parameter value t, and then to use the formula (5).

Another, more direct method, is de Casteljau's algorithm. We first set $\mathbf{c}_i^0 = \mathbf{c}_i$, and then for each $r = 1, \ldots, n$, let

$$\mathbf{c}_i^r = (1-t)\mathbf{c}_i^{r-1} + t\mathbf{c}_{i+1}^{r-1}, \qquad i = 0, 1, \ldots, n-r. \tag{6}$$

The last point computed in this algorithm is the point on the curve:

$$\mathbf{p}(t) = \mathbf{c}_0^n.$$

We can show this by showing more generally that for any $r = 1, \ldots, n$,

$$\mathbf{p}(t) = \sum_{i=0}^{n-r} \mathbf{c}_i^r B_i^{n-r}(t). \tag{7}$$

This equation clearly holds for $r = 0$, and for $r \geq 1$ we use induction on r. Applying the recursion formula (3) to (7) gives

$$\begin{aligned} \mathbf{p}(t) &= \sum_{i=0}^{n-r} \mathbf{c}_i^r \left(tB_{i-1}^{n-r-1}(t) + (1-t)B_i^{n-r-1}(t) \right) \\ &= \sum_{i=0}^{n-r-1} \left(t\mathbf{c}_{i+1}^r + (1-t)\mathbf{c}_i^r \right) B_i^{n-r-1}(t), \end{aligned}$$

which is (7) with r replaced by $r+1$. Like the recursive algorithm for computing Bernstein polynomials, the de Casteljau algorithm can be viewed as a triangular scheme, here arranged row-wise, with each row being computed from the row above:

$$\begin{array}{ccccccccc} \mathbf{c}_0^0 & & \mathbf{c}_1^0 & & \mathbf{c}_2^0 & & \cdots & & \mathbf{c}_n^0 \\ & \mathbf{c}_0^1 & & \mathbf{c}_1^1 & & \cdots & & \mathbf{c}_{n-1}^1 & \\ & & \ddots & & & & \iddots & & \\ & & & \mathbf{c}_0^{n-1} & & \mathbf{c}_1^{n-1} & & & \\ & & & & \mathbf{c}_0^n & & & & \end{array}$$

For example with $n = 3$, $t = 2/3$, $d = 1$ and

$$[\mathbf{c}_0 \quad \mathbf{c}_1 \quad \mathbf{c}_2 \quad \mathbf{c}_3] = [4 \quad 0 \quad 4 \quad 18],$$

the weights in (6) are $t = 2/3$ and $1 - t = 1/3$, and the algorithm has three steps,

$$[4 \quad 0 \quad 4 \quad 18] \to [4/3 \quad 8/3 \quad 40/3] \to [20/9 \quad 88/9] \to [196/27].$$

4 Derivatives

Differentiation of the expression for the Bernstein polynomial in (1) with respect to x gives

$$\frac{d}{dx}B_i^n(x) = n\left(B_{i-1}^{n-1}(x) - B_i^{n-1}(x)\right), \tag{8}$$

and so if we differentiate the Bezier curve in (5) with respect to its parameter t we find

$$\mathbf{p}'(t) = n\left(\sum_{i=0}^{n-1}\mathbf{c}_{i+1}B_i^{n-1}(t) - \sum_{i=0}^{n-1}\mathbf{c}_iB_i^{n-1}(t)\right). \tag{9}$$

Thus one way of expressing the derivative is as

$$\mathbf{p}'(t) = n\sum_{i=0}^{n-1}\Delta\mathbf{c}_iB_i^{n-1}(t), \tag{10}$$

where Δ denotes the forward difference operator,

$$\Delta\mathbf{c}_i = \mathbf{c}_{i+1} - \mathbf{c}_i.$$

This means that the derivative of $\mathbf{p}$ is itself a Bezier curve with control points (which we now view as vectors) $n\Delta\mathbf{c}_i$. For each $t \in [0,1]$, the tangent vector $\mathbf{p}'(t)$ lies in the convex cone of the vectors $\Delta\mathbf{c}_i$, and by the endpoint property of Bezier curves,

$$\mathbf{p}'(0) = n\Delta\mathbf{c}_0 \quad \text{and} \quad \mathbf{p}'(1) = n\Delta\mathbf{c}_{n-1}.$$

An alternative way of expressing the derivative is in terms of the intermediate points (6) of the de Casteljau algorithm. The intermediate point $\mathbf{c}_i^r$ depends on t and so we write it as $\mathbf{c}_i^r(t)$. This point is itself the result of applying the de Casteljau algorithm at t in r steps to the points $\mathbf{c}_i, \ldots, \mathbf{c}_{i+r}$, and therefore

$$\mathbf{c}_i^r(t) = \sum_{j=0}^{r}\mathbf{c}_{i+j}B_j^r(t).$$

Setting $r = n-1$, it follows from (9) that

$$\mathbf{p}'(t) = n\left(\mathbf{c}_1^{n-1}(t) - \mathbf{c}_0^{n-1}(t)\right) = n\Delta\mathbf{c}_0^{n-1}(t). \tag{11}$$

5 Higher derivatives

Applying the derivative formula (10) repeatedly leads to

$$\mathbf{p}^{(r)}(t) = \frac{d^r}{dt^r}\mathbf{p}(t) = \frac{n!}{(n-r)!}\sum_{i=0}^{n-r} \Delta^r \mathbf{c}_i B_i^{n-r}(t),$$

for any $r = 1, \ldots, n$, where Δ^r is the r-th forward difference operator

$$\Delta^r \mathbf{c}_i = \Delta^{r-1}\mathbf{c}_{i+1} - \Delta^{r-1}\mathbf{c}_i.$$

At the endpoints of the curve, the r-th derivative depends only on the first or last $r+1$ control points:

$$\mathbf{p}^{(r)}(0) = \frac{n!}{(n-r)!}\Delta^r \mathbf{c}_0 \quad \text{and} \quad \mathbf{p}^{(r)}(1) = \frac{n!}{(n-r)!}\Delta^r \mathbf{c}_{n-r}.$$

Alternatively, in terms of the intermediate de Casteljau points, differentiating (11) repeatedly gives

$$\mathbf{p}^{(r)}(t) = \frac{n!}{(n-r)!}\Delta^r \mathbf{c}_0^{n-r}(t).$$

6 Integration

Integrating the dervative formula (8) with respect to x in $[0,1]$ gives

$$B_i^n(1) - B_i^n(0) = n\left(\int_0^1 B_{i-1}^{n-1}(x)\,dx - \int_0^1 B_i^{n-1}(x)\,dx\right),$$

and since the left hand side is zero for any $i = 1, \ldots, n-1$, we deduce that

$$\int_0^1 B_{i-1}^{n-1}(x)\,dx = \int_0^1 B_i^{n-1}(x)\,dx.$$

Thus the integral on $[0,1]$ of each Bernstein polynomial of the same degree is constant. Since the Bernstein polynomials of degree n sum to one and there are $n+1$ of them,

$$\int_0^1 B_i^n(x)\,dx = \frac{1}{n+1}.$$

It follows that the integral over t in $[0,1]$ of the Bezier curve $\mathbf{p}$ in (5) is

$$\int_0^1 \mathbf{p}(t)\,dt = \frac{\mathbf{c}_0 + \mathbf{c}_1 + \cdots + \mathbf{c}_n}{n+1},$$

which is the barycentre of the control points $\mathbf{c}_0, \ldots, \mathbf{c}_n$.

7 Conversion to Bezier form

Sometimes we need to convert a polynomial from monomial form to Bezier form and vice versa.

Suppose we start with the monomial form (4) and want to convert it to the Bezier form (5). We use the fact that

$$1 = (1 - t + t)^{n-i} = \sum_{j=0}^{n-i} \binom{n-i}{j} t^j (1-t)^{n-i-j},$$

to show that

$$\begin{aligned}
\mathbf{p}(t) &= \sum_{i=0}^{n} \mathbf{a}_i \sum_{j=0}^{n-i} \binom{n-i}{j} t^{i+j} (1-t)^{n-i-j} \\
&= \sum_{i=0}^{n} \mathbf{a}_i \sum_{j=i}^{n} \binom{n-i}{j-i} t^j (1-t)^{n-j}, \\
&= \sum_{j=0}^{n} \sum_{i=0}^{j} \mathbf{a}_i \binom{n-i}{j-i} t^j (1-t)^{n-j},
\end{aligned}$$

and therefore

$$\mathbf{c}_j = \frac{1}{\binom{n}{j}} \sum_{i=0}^{j} \binom{n-i}{j-i} \mathbf{a}_i.$$

Conversely, suppose we want to convert the Bezier form (5) to the monomial form (4). To do this, observe that

$$(1-t)^{n-i} = \sum_{j=0}^{n-i} \binom{n-i}{j} (-1)^j t^j,$$

and so

$$\begin{aligned}\mathbf{p}(t) &= \sum_{i=0}^{n} \mathbf{c}_i \binom{n}{i} \sum_{j=0}^{n-i} \binom{n-i}{j} (-1)^j t^{i+j} \\ &= \sum_{i=0}^{n} \mathbf{c}_i \binom{n}{i} \sum_{j=i}^{n} \binom{n-i}{j-i} (-1)^{j-i} t^j, \\ &= \sum_{j=0}^{n} \sum_{i=0}^{j} \mathbf{c}_i \binom{n}{i} \binom{n-i}{j-i} (-1)^{j-i} t^j,\end{aligned}$$

and it follows that

$$\mathbf{a}_j = \sum_{i=0}^{j} \binom{n}{i} \binom{n-i}{j-i} (-1)^{j-i} \mathbf{c}_i.$$

This can alternatively we written as

$$\mathbf{a}_j = \sum_{i=0}^{j} \binom{n}{j} \binom{j}{i} (-1)^{j-i} \mathbf{c}_i = \binom{n}{j} \Delta^j \mathbf{c}_0.$$