Bezier Curves and B-splines, Blossoming

Lecture #2: 16 September 2002
Lecturer: Prof. Denis Zorin
Scribe: Kranthi K Gade

In this lecture, we discuss polynomial parametric curves, which is the most commonly used type of curves in geometric modeling. We consider two ways of representing such curves which are useful for computational purposes *viz.*, *Bezier* and *B-spline* curves. We also discuss some properties of *Bezier* and *B-spline* curves.

Representation of Curves

For computational purposes, we need to represent curves in some form. The most commonly used representation is the parametric representation. As we have seen in the previous lecture, parametric method is to represent the curve in the form of a function of one parameter

$$y(t) : \mathbb{R} \to \mathbb{R}^n, n = 2, 3, \ldots$$

It would be good if the function $y(t)$ is simple because using arbitrary functions which can be obtained say by composition of commonly used elementary functions is computationally expensive. So we look for class of functions which is as simple as possible and yet diverse enough to represent a wide variety of curves. Polynomial functions to large extent satisfy this requirement. A general polynomial function is represented in the following way:

$$y(t) = \sum_{i=0}^{n} a_i t^i$$

where n is the degree of the polynomial.

The variety of curves that you can obtain using polynomials depends on the maximum allowed degree. The higher the degree, the greater variety of shapes one can represent. For example, to define a curve with n wiggles, we need a polynomial of degree $n + 1$. But higher degrees result in some problems. The computational complexity of maintaining the curve increases with increasing degree. The higher the degree of a curve, the less controllable it is, in a sense that small changes in coefficients are likely to result in large changes in the shape of the curve. Furthermore, the curves of high degree are more likely to develop "bumps" in unpredictable when the shape of the curve is changed.

Let us try to figure out what is the minimal degree we can get away with. For simplicity, let us consider the functional case only. Degree 1 polynomials can only be used to represent straight lines; degree 2 curves are parabolas in non-degenerate cases. The problem with parabola is that it does not have a point of inflection, where the curve turns from being concave to convex or vice-versa, and hence a curve with point of inflection cannot be represented using a degree 2 polynomial; we can stitch several degree 2 polynomials together, but at inflection points the

curvature is likely to be discontinuous unless it is zero. Cubics on the other hand may have points of inflection with curvature changing continuously.

To represent arbitrarily complex curves one usually uses piecewise polynomials, stitching together many polynomial pieces.

The next question we need to address is how to specify polynomial curves. The most straightforward approach is to define a curve using the polynomial coefficients. For example, a two-dimensional curve can be represented using polynomials in the following way:

$$y(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

where a_i's are two-dimensional points. One can see that $y(t)$ is the weighted average of a_0, a_1, a_2, a_3 but it is not intuitively clear though how $y(t)$ is traversed as t changes, and how a_i influence the shape of $y(t)$.

Bezier Curves

One can make the coefficients describing a polynomial more intuitive by changing the basis functions. Instead of using the standard monomial basis $[1, t, t^2, t^3]$, we use the *Bernstein* basis: $[(1-t)^3, 3t(1-t)^2, 3t^2(1-t), t^3]$. One way to derive this set of basis function is to look at the expansion of 1^3:

$$1 = (1 - t + t)^3 = (1-t)^3 + 3t(1-t)^2 + 3t^2(1-t) + t^3 \tag{1}$$

Why this basis? We shall see that Bezier curves have a number of nice properties. Bernstein basis is defined for any n. For $n = 1$, the *Bernstein* basis is $[(1-t), t]$ and for $n = 2$ it is $[(1-t)^2, 2t(1-t), t^2]$. The Bernstein basis for $n = 3$ are shown in Figure 1.

Any polynomial can be written as a combination of the four polynomials of the Bernstein basis. If p_0, p_1, p_2, p_3 are four points in space, then the cubic polynomial curve

$$p^3(t) = \sum_{i=0}^{3} p_i B_{3,i}(t) \tag{2}$$

where $B_{3,i}$ is the ith *Bernstein* polynomial is called a
cubic Bezier curve with control points $p_0, \ldots p_3$.

Properties of cubic Bezier curves

1. **Interpolation.** One can easily see that $p^3(0) = p_0$ and $p^3(1) = p_3$, i.e. the Bezier curve $p^3(t)$ interpolates the points p_0,and p_3

2. **Affine Invariance.** This property can be easily verified by considering an affine map $\Phi(\mathbf{x}) = A\mathbf{x} + \mathbf{v}$ where A is a 3×3 matrix and $\mathbf{v}$ is in $\mathbb{R}^3$. Now

$$\sum_{i=0}^{3} \Phi(p_i) B_{3,i}(t) = \sum_{i=0}^{3} (Ap_i + \mathbf{v}) B_{3,i})(t) = A \sum_{i=0}^{3} p_i B_{3,i}(t) + \mathbf{v} = \Phi(\sum_{i=0}^{3} p_i B_{3,i}(t))$$

   The last transformation uses the fact that Bernstein basis functions sum up to one (1).

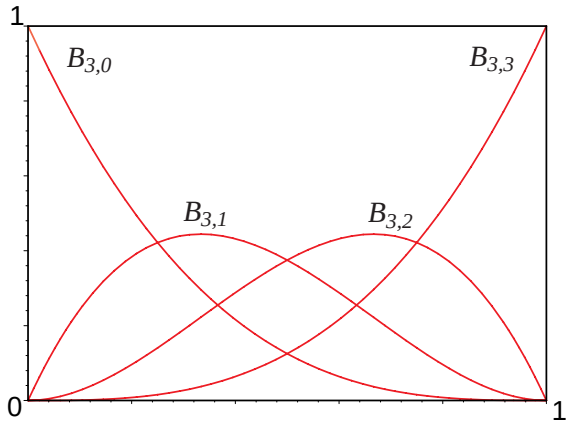

Figure 1: Bernstein basis functions for $n = 3$

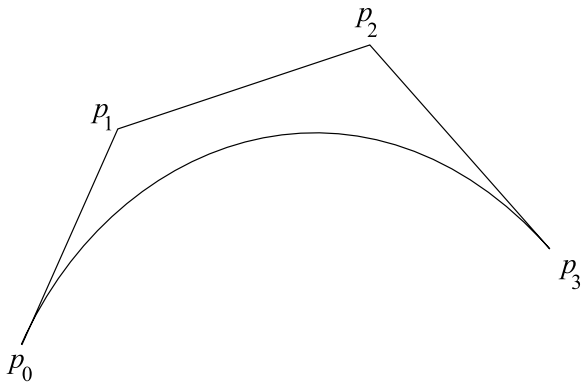

Figure 2: Bezier Interpolation

3. Tangent to the curve at point p_0 is the vector $p_0\bar{p}_1$ and tangent to the curve at p_3 is $p_2\bar{p}_3$. This can be easily verified by differentiating (2) and substituting appropriate parameter values: $t = 0$ for p_0 and $t = 1$ for p_3. This property can be used to test whether two Bezier curves are joined smoothly: use this property to find the tangents at the common point and then see whether they are parallel.

4. **Convex Hull.** It is clear that $B_{3,i}(t) \geq 0$ for $t \in [0, 1]$. This and (1) means that all points on the Bezier curve lie inside the convex hull of points $p_0 \ldots p_3$.

Polar forms and blossoming

Polar form of a polynomial of $P(t) : \mathbb{R} \to \mathbb{R}$ of degree n is a multiaffine symmetric function $P(t_1, t_2, \ldots, t_n) : \mathbb{R}^n \to \mathbb{R}$ such that

$$P(\underbrace{t, t, \ldots, t}_{n}) = P(t) \tag{3}$$

We focus on the case $n = 3$. The two properties of polar forms are defined as follows:

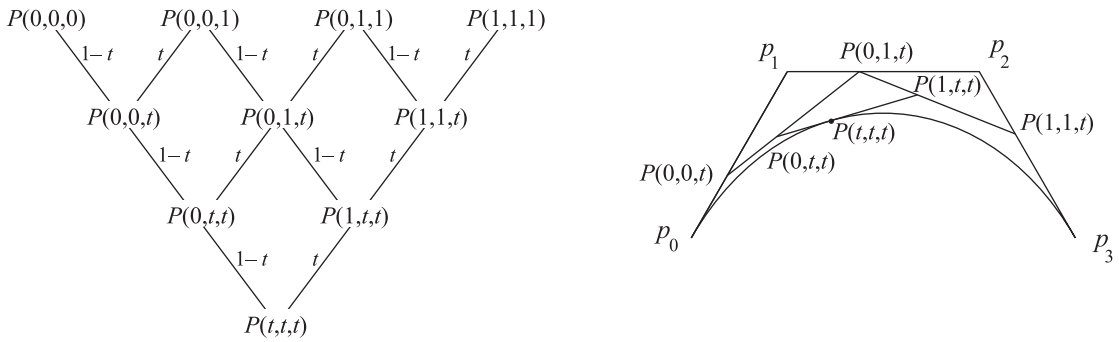

Figure 3: Blossoming for Bezier curves.

- A function $f(t_1, \ldots, t_n)$ is **symmetric** if the value of the function is the same for any permutation of the arguments $t_1, \ldots t_n$.
- A function $f(t_1, \ldots, t_n)$ is **multiaffine** if for any arguments $t_1, \ldots t_j^0, t_j^1, \ldots t_n$ and any a

$$f(t_1, \ldots, at_j^0 + (1-a)t_j^1, \ldots t_n) = af(t_1, \ldots t_j^0, t_{j+1}, \ldots t_n) + (1-a)f(t_1, \ldots, t_j^1, \ldots, t_n)$$

For any polynomial, there exists a unique symmetric and multiaffine polar form. We do not give a formal proof for it but an intuition behind it for the case $n = 3$ will make it clear. Because the polar form is multiaffine it is linear in each argument; therefore it is a polynomial in t_1, t_2, t_3, of total power 3 as it follows from (3). Therefore the polar form can be expressed as $b_0 + b_{11}t_2 + b_{12}t_3 + b_{13}t_3 + b_{21}t_1t_2 + b_{22}t_2t_3 + b_{23}t_1t_3 + +b_3t_1t_2t_3$. Writing equations for symmetry and taking (3) into account yields a

$$P(t_1, t_2, t_3) = a_0 + \frac{a_1}{3}(t_1 + t_2 + t_3) + \frac{a_2}{3}(t_1t_2 + t_2t_3 + t_1t_3) + a_3t_1t_2t_3$$

Suppose $P(t_1, t_2, t_3)$ is the polar form of a cubic polynomial $P(t)$, then the value of $P(t)$ for any arbitrary t can be found if the values of $P(0, 0, 0), P(0, 0, 1)P(0, 1, 1)$ and $P(1, 1, 1)$ are given. This is done by repeated linear interpolation and is shown in Figure 3.

In this way we can compute the value of the polynomial at any point t using a sequence of linear interpolations. Polar form representation is useful because it provides a uniform and simple approach to computing values of a polynomial using a variety of representations from Bezier curves to NURBS (non-uniform rational B-splines).

B-splines

B-splines is a different approach to representing piecewise polynomial curves, which overcomes some of the drawbacks of Bezier curves.

Suppose we wish to design a long curve with many undulations. One approach would be to use a high-degree Bezier curve. As we have already discussed, this is not a very good approach; also note that each of $n + 1$ control points of a Bezier curve of degree n influences the whole curve, which means that it is difficult to introduce a small feature without changing the curve everywhere. Another approach using Bezier curves is to construct the curve from many parts

(*piecewise Bezier*); in this case we need to match at least the values and tangents at the endpoints of each segment. Values are easily matched by constraining the last control point of segment j and the first control point of segment $j+1$ to be the same: $p_3^j = p_0^{j+!}$. Tangents can be made the same if the points $p_2^j, p_3^j = p_0^{j+1}, p_1^{j+1}$ of the segments j and $j+1$ are constrained to be on the same line. This is the way curves are constructed in many drawing programs, for example, in Adobe Illustrator. Resulting curves need not be C^2 however, and it is not that easy to make them "fair" which typically means that we want the curvature not to change abruptly.

To summarize, three commonly desirable properties of curves are:

1. **C^2-continuity:** Should be C^2 continuous at all points.
2. **Interpolation:** Should interpolate the input control points.
3. **Local control:** The modification of a particular control point should modify the curve only locally.

As we have seen, piecewise Bezier curves interpolate the control points shared by segments, and have local control, but need not be C^2-continuous. It turns out that these three properties are incompatible for cubic curves: e.g. if we attempt to write a system of equations that ensures that sequential Bezier segments are joined with C^2 continuity, we will see that "interior" control points p_1^j p_2^j for segment j have to depend on the endpoints $p_3^i = p_0^{i+1}$ for all i, i.e. there is no local control.

If we give up the interpolation property there is a type of piecewise cubic polynomial curves which satisfies the remaining requirements called the *B-spline curves*. If we require interpolation but give up local control we get curves called *natural splines*, which we will not discuss in detail.

A B-spline curve defined everywhere on $\mathbb{R}$ can be written in the following form:

$$p(t) = \sum_{i==\infty}^{\infty} p_i B_i(t)$$

where p_i are control points and $B_i(t)$ are the basis functions associated with control points p_i. Each basis function can be thought of as the variable weight which determines how the control point p_i influences the curve at parametric value t. For *uniform splines*, the basis functions satisfy

$$B_i(t) = B(t-i) \tag{4}$$

for a fixed function B.

We choose the basis function $B(t)$ in such a way that the resulting curves are C^2-continuous, the influence of the control points is local (i.e. $B(t) \neq 0$ on the smallest possible interval) and the curves are piecewise cubic polynomials on each each interval $[i, i+1]$. We would like the influence of a control point to be maximal at regions of the curve close to it and for this influence to decrease as we move away along the curve and disappear completely at some distance. Also desirable is affine invariance property that we have seen for Bezier curves. we describe how to construct basis functions satisfying all these requirements.

There are many different ways to define B-splines; we will consider two equivalent definitions: using convolution and using blossoming.

B-spline basis functions

We start with the simplest functions which already meet some of the requirements above: piecewise constant coordinate functions. Any piecewise constant function can be written as

$$p(t) = \sum_{i=-\infty}^{\infty} p_i Box(t-i)$$

where $Box(t)$ is the box function defined as

$$Box(t) = \begin{cases} 1, & \text{if } 0 \le t < 1 \\ 0 & \text{otherwise} \end{cases}$$

We define the convolution of two functions $f(t)$ and $g(t)$ as

$$(f \star g)(t) = \int_{-\infty}^{\infty} f(s)g(t-s)ds$$

The remarkable property of convolution is that each time we convolve a function with a box its smoothness increases. We will see that convolution can be seen as "moving average" operation.

A B-spline basis function of degree n can be obtained by convolving a B-spline basis function of degree $n-1$ with the box $Box(t)$. For example, the basis function of degree 1 is defined as the convolution of $Box(t)$ with itself. We need to compute

$$\int_{-\infty}^{\infty} Box(s)Box(t-s)ds$$

Graphically, this convolution can be evaluated by sliding one box function along the coordinate axis from $-\infty$ to ∞ and keeping the second box fixed (see Figure 4). The value of the convolution for a given position of the second box is simply the area under the product of the boxes, which is just the length of the interval where both boxes are non-zero.

At first the two graphs do not have common area. Once the moving box reaches 0, there is a growing overlap between the areas of the graphs. The value of the convolution increases until $t = 1$. Then the overlap starts decreasing, and the value of the convolution decreases down to zero at $t = 2$. The function $B_1(t) = (Box \star Box)(t)$ is as shown in Figure 4.

Now take the convolution of $Box(t)$ with $B_1(t)$ and we obtain $B_2(t)$, the degree 2 basis function. We can continue further and obtain degree 3 B-spline, $B_3(t)$ by convolving $B_2(t)$ with $Box(t)$ (Figure 5).

Properties of B-splines

- $B_n(t)$ is a piecewise polynomial of degree n (each convolution increases the degree by 1).
- $B_n(t)$ has a support of length $n+1$. We have seen that $Box(t)$ has a support 1 and each convolution increases the support by 1.

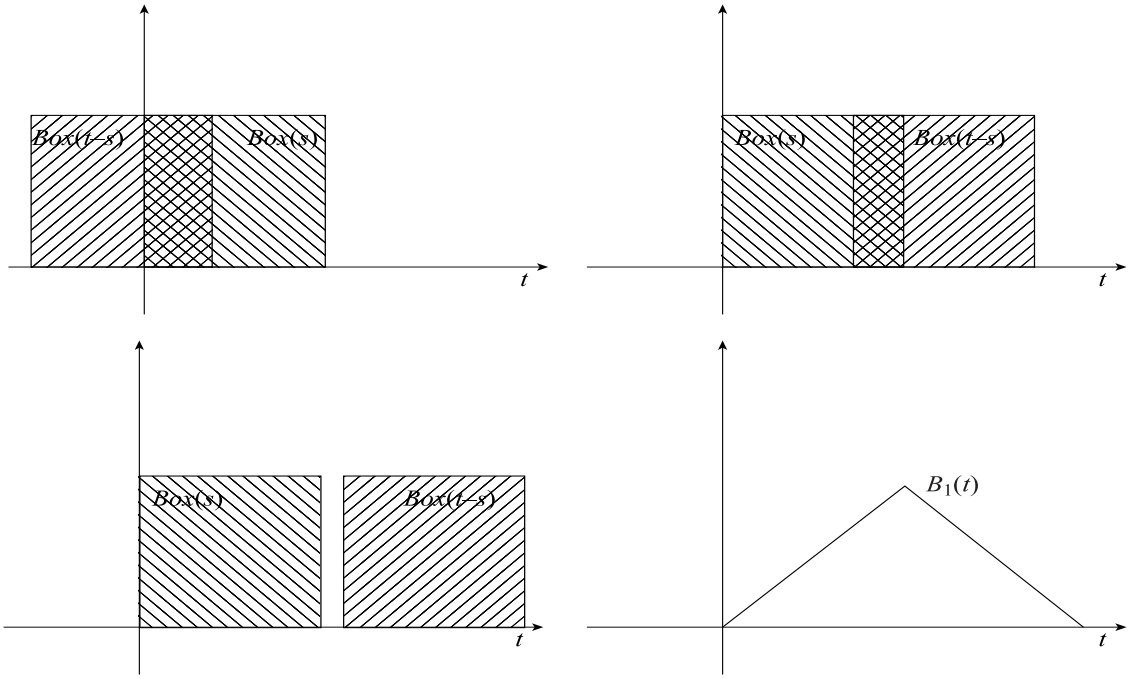

Figure 4: The definition of Degree 1 B-spline basis function through convolution of Box function with itself.

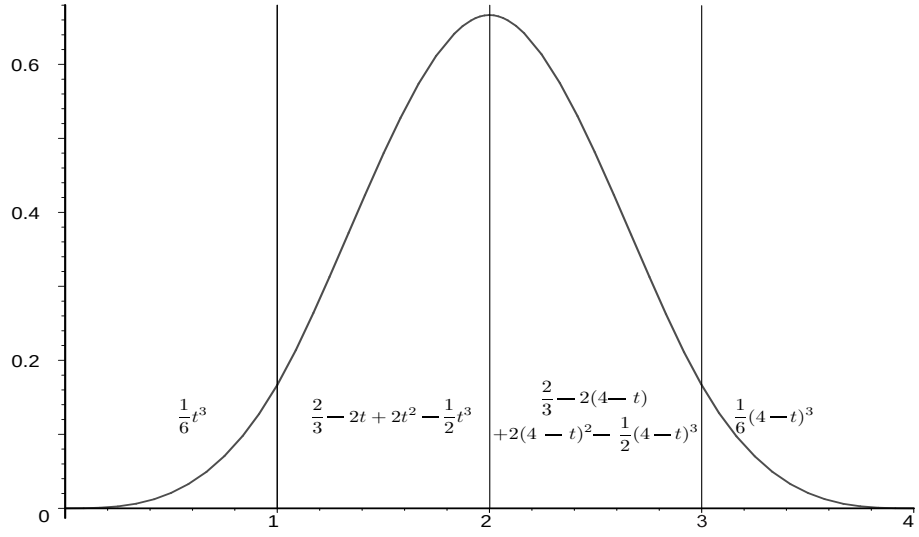

Figure 5: Degree 3 B-spline basis function

- $B^n(t)$ is C^{n-1}-continuous. $Box(t)$ is C^0 continuous and each convolution increases smoothness by 1.

- The set of functions $B_n(t-i)$, $i = -\infty \ldots \infty$ is affine invariant. This comes from the observation that

$$\sum_{i=-\infty}^{\infty} B_n(t-i) = 1$$

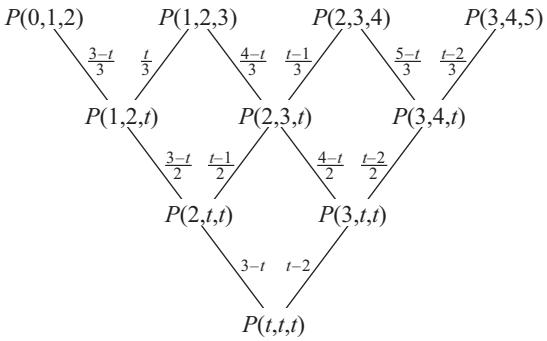

Figure 6: Blossoming in B-splines

which can be proved by in induction observing that this property holds for the box function and is preserved by the convolution.

Blossoming in B-splines

For a cubic B-spline, if $t \in [2, 3]$, then we can obtain the value of $P(t) = P(t, t, t)$ from the values of $P(0, 1, 2), P(1, 2, 3), P(2, 3, 4), P(3, 4, 5)$ as shown in Figure 6.

For $t \in [i, i + 1]$ and cubic B-splines, we need four control points $P(i - 2, i - 1, i), P(i - 1, i, i + 1), P(i, i + 1, i + 2), P(i + 1, i + 2, i + 3)$.