On-Line Geometric Modeling Notes

BERNSTEIN POLYNOMIALS

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

Polynomials are incredibly useful mathematical tools as they are simply defined, can be calculated quickly on computer systems and represent a tremendous variety of functions. They can be differentiated and integrated easily, and can be pieced together to form spline curves that can approximate any function to any accuracy desired. Most students are introducted to polynomials at a very early stage in their studies of mathematics, and would probably recall them in the form below:

$$p(t) \;=\; a_n t^n + a_{n-1} t^{n-1} + \cdots + a_1 t + a_0$$

which represents a polynomial as a linear combination of certain elementary polynomials $\left\{(1, t, t^2, ..., t^n\right\}$.

In general, any polynomial function that has degree less than or equal to n, can be written in this way, and the reasons are simply

- The set of polynomials of degree less than or equal to n forms a vector space: polynomials can be added together, can be multiplied by a scalar, and all the vector space properties hold.
- The set of functions $\left\{1, t, t^2, ..., t^n\right\}$ form a basis for this vector space – that is, any polynomial of degree less than or equal to n can be uniquely written as a linear combinations of these functions.

This basis, commonly called the *power basis*, is only one of an infinite number of bases for the space of polynomials.

In these notes we discuss another of the commonly used bases for the space of polynomials, the *Bernstein basis*, and discuss its many useful properties.

Bernstein Polynomials

The Bernstein polynomials of degree n are defined by

$$B_{i,n}(t) \;=\; \binom{n}{i} t^i (1-t)^{n-i}$$

for $i = 0, 1, ..., n$, where

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

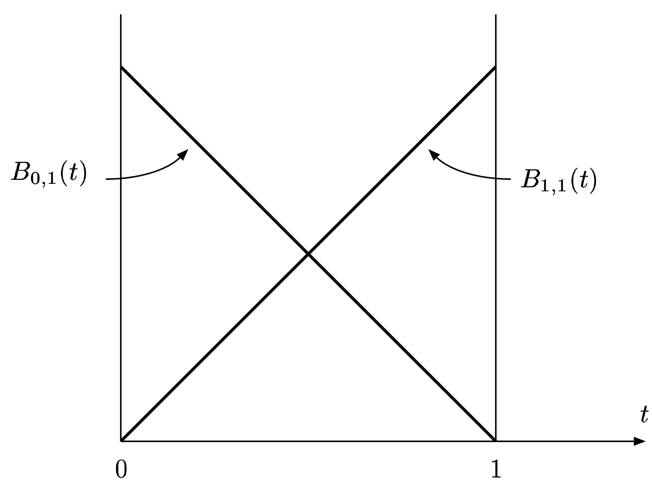
There are $n+1$ nth-degree Bernstein polynomials. For mathematical convenience, we usually set $B_{i,n} = 0$, if $i < 0$ or $i > n$.

These polynomials are quite easy to write down: the coefficients $\binom{n}{i}$ can be obtained from Pascal's triangle; the exponents on the t term increase by one as i increases; and the exponents on the $(1-t)$ term decrease by one as i increases. In the simple cases, we obtain

- The Bernstein polynomials of degree 1 are

$$B_{0,1}(t) = 1 - t$$
$$B_{1,1}(t) = t$$

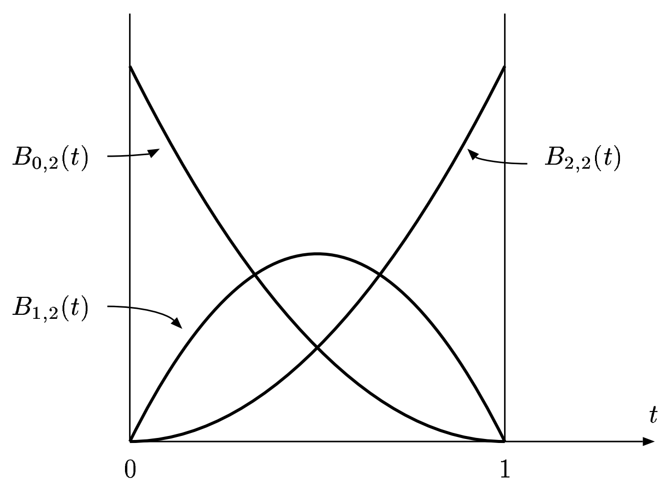
  and can be plotted for $0 \le t \le 1$ as



- The Bernstein polynomials of degree 2 are

$$B_{0,2}(t) = (1-t)^2$$
$$B_{1,2}(t) = 2t(1-t)$$
$$B_{2,2}(t) = t^2$$

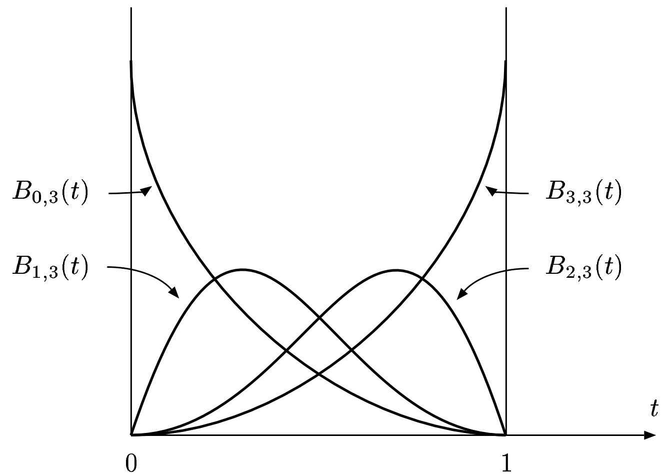
and can be plotted for $0 \leq t \leq 1$ as



- The Bernstein polynomials of degree 3 are

$$B_{0,3}(t) = (1-t)^3$$
$$B_{1,3}(t) = 3t(1-t)^2$$
$$B_{2,3}(t) = 3t^2(1-t)$$
$$B_{3,3}(t) = t^3$$

and can be plotted for $0 \leq t \leq 1$ as


$B_{0,3}(t)$
$B_{1,3}(t)$
$B_{3,3}(t)$
$B_{2,3}(t)$
t
0
1

A Recursive Definition of the Bernstein Polynomials

The Bernstein polynomials of degree n can be defined by blending together two Bernstein polynomials of degree $n-1$. That is, the kth nth-degree Bernstein polynomial can be written as

$$B_{k,n}(t) \;=\; (1-t)B_{k,n-1}(t) + tB_{k-1,n-1}(t)$$

To show this, we need only use the definition of the Bernstein polynomials and some simple algebra:

$$\begin{aligned}
(1-t)B_{k,n-1}(t) + tB_{k-1,n-1}(t) &= (1-t)\binom{n-1}{k}t^k(1-t)^{n-1-k} + t\binom{n-1}{k-1}t^{k-1}(1-t)^{n-1-(k-1)} \\
&= \binom{n-1}{k}t^k(1-t)^{n-k} + \binom{n-1}{k-1}t^k(1-t)^{n-k} \\
&= \left[\binom{n-1}{k} + \binom{n-1}{k-1}\right] t^k(1-t)^{n-k} \\
&= \binom{n}{k}t^k(1-t)^{n-k} \\
&= B_{k,n}(t)
\end{aligned}$$

The Bernstein Polynomials are All Non-Negative

A function $f(t)$ is non-negative over an interval $[a,b]$ if $f(t) \geq 0$ for $t \in [a,b]$. In the case of the Bernstein polynomials of degree n, each is non-negative over the interval $[0,1]$. To show this we use the recursive definition property above and mathematical induction.

It is easily seen that the functions $B_{0,1}(t) = 1-t$ and $B_{1,1}(t) = t$ are both non-negative for $0 \leq t \leq 1$. If we assume that all Bernstein polynomials of degree less than k are non-negative, then by using the recursive definition of the Bernstein polynomial, we can write

$$B_{i,k}(t) \;=\; (1-t)B_{i,k-1}(t) + tB_{i-1,k-1}(t)$$

and argue that $B_{i,k}(t)$ is also non-negative for $0 \leq t \leq 1$, since all components on the right-hand side of the equation are non-negative components for $0 \leq t \leq 1$. By induction, all Bernstein polynomials are non-negative for $0 \leq t \leq 1$.

In this process, we have also shown that each of the Bernstein polynomials is *positive* when $0 < t < 1$.

The Bernstein Polynomials form a Partition of Unity

A set of functions $f_i(t)$ is said to partition unity if they sum to one for all values of t. The $k+1$ Bernstein polynomials of degree k form a partition of unity in that they all sum to one.

To show that this is true, it is easiest to first show a slightly different fact: for each k, the sum of the $k+1$ Bernstein polynomials of degree k is equal to the sum of the k Bernstein polynomials of degree $k-1$. That is,

$$\sum_{i=0}^{k} B_{i,k}(t) \;=\; \sum_{i=0}^{k-1} B_{i,k-1}(t)$$

This calculation is straightforward, using the recursive definition and cleverly rearranging the sums:

$$\begin{aligned}
\sum_{i=0}^{k} B_{i,k}(t) &= \sum_{i=0}^{k} \left[(1-t)B_{i,k-1}(t) + tB_{i-1,k-1}(t)\right] \\
&= (1-t)\left[\sum_{i=0}^{k-1} B_{i,k-1}(t) + B_{k,k-1}(t)\right] + t\left[\sum_{i=1}^{k} B_{i-1,k-1}(t) + B_{-1,k-1}(t)\right] \\
&= (1-t)\sum_{i=0}^{k-1} B_{i,k-1}(t) + t\sum_{i=1}^{k} B_{i-1,k-1}(t) \\
&= (1-t)\sum_{i=0}^{k-1} B_{i,k-1}(t) + t\sum_{i=0}^{k-1} B_{i,k-1}(t) \\
&= \sum_{i=0}^{k-1} B_{i,k-1}(t)
\end{aligned}$$

(where we have utilized $B_{k,k-1}(t) = B_{-1,k-1}(t) = 0$).

Once we have established this equality, it is simple to write

$$\sum_{i=0}^{n} B_{i,n}(t) \;=\; \sum_{i=0}^{n-1} B_{i,n-1}(t) \;=\; \sum_{i=0}^{n-2} B_{i,n-2}(t) \;=\; \cdots \;=\; \sum_{i=0}^{1} B_{i,1}(t) \;=\; (1-t)+t \;=\; 1$$

The partition of unity is a very important property when utilizing Bernstein polynomials in geometric modeling and computer graphics. In particular, for any set of points $\mathbf{P}_0$, $\mathbf{P}_1$, ..., $\mathbf{P}_n$, in three-dimensional space, and for any t, the expression

$$\mathbf{P}(t) = \mathbf{P}_0 B_{0,n}(t) + \mathbf{P}_1 B_{1,n}(t) + \cdots + \mathbf{P}_n B_{n,n}(t)$$

is an affine combination of the set of points $\mathbf{P}_0, \mathbf{P}_1, ..., \mathbf{P}_n$ and if $0 \leq t \leq 1$, it is a convex combination of the points.

Degree Raising

Any of the lower-degree Bernstein polynomials (degree $< n$) can be expressed as a linear combination of Bernstein polynomials of degree n. In particular, any Bernstein polynomial of degree $n-1$ can be written as a linear combination of Bernstein polynomials of degree n. We first note that

$$\begin{aligned} tB_{i,n}(t) &= \binom{n}{i} t^{i+1}(1-t)^{n-i} \\ &= \binom{n}{i} t^{i+1}(1-t)^{(n+1)-(i+1)} \\ &= \frac{\binom{n}{i}}{\binom{n+1}{i+i}} B_{i+1,n+1}(t) \\ &= \frac{i+1}{n+1} B_{i+1,n+1}(t) \end{aligned}$$

and

$$\begin{aligned} (1-t)B_{i,n}(t) &= \binom{n}{i} t^{i}(1-t)^{n+1-i} \\ &= \frac{\binom{n}{i}}{\binom{n+1}{i}} B_{i,n+1}(t) \\ &= \frac{n-i+1}{n+1} B_{i,n+1}(t) \end{aligned}$$

and finally

$$\begin{aligned} \frac{1}{\binom{n}{i}} B_{i,n}(t) + \frac{1}{\binom{n}{i+1}} B_{i+1,n}(t) &= t^i(1-t)^{n-i} + t^{i+1}(1-t)^{n-(i+1)} \\ &= t^i(1-t)^{n-i-1}((1-t)+t) \\ &= t^i(1-t)^{n-i-1} \\ &= \frac{1}{\binom{n-1}{i}} B_{i,n-1}(t) \end{aligned}$$

Using this final equation, we can write an arbitrary Bernstein polynomial in terms of Bernstein polynomials of higher degree. That is,

$$\begin{aligned}
B_{i,n-1}(t) &= \binom{n-1}{i}\left[\frac{1}{\binom{n}{i}}B_{i,n}(t) + \frac{1}{\binom{n}{i+1}}B_{i+1,n}(t)\right] \\
&= \left(\frac{n-i}{n}\right)B_{i,n}(t) + \left(\frac{i+1}{n}\right)B_{i+1,n}(t)
\end{aligned}$$

which expresses a Bernstein polynomial of degree $n-1$ in terms of a linear combination of Bernstein polynomials of degree n. We can easily extend this to show that any Bernstein polynomial of degree k (less than n) can be written as a linear combination of Bernstein polynomials of degree n – e.g., a Bernstein polynomial of degree $n-2$ can be expressed as a linear combination of two Bernstein polynomials of degree $n-1$, each of which can be expressed as a linear combination of two Bernstein polynomials of degree n, etc.

Converting from the Bernstein Basis to the Power Basis

Since the power basis $\{1, t, t^2, ..., t^n\}$ forms a basis for the space of polynomials of degree less than or equal to n, any Bernstein polynomial of degree n can be written in terms of the power basis. This can be directly calculated using the definition of the Bernstein polynomials and the binomial theorem, as follows:

$$\begin{aligned}
B_{k,n}(t) &= \binom{n}{k}t^k(1-t)^{n-k} \\
&= \binom{n}{k}t^k\sum_{i=0}^{n-k}(-1)^i\binom{n-k}{i}t^i \\
&= \sum_{i=0}^{n-k}(-1)^i\binom{n}{k}\binom{n-k}{i}t^{i+k} \\
&= \sum_{i=k}^{n}(-1)^{i-k}\binom{n}{k}\binom{n-k}{i-k}t^i \\
&= \sum_{i=k}^{n}(-1)^{i-k}\binom{n}{i}\binom{i}{k}t^i
\end{aligned}$$

where we have used the binomial theorem to expand $(1-t)^{n-k}$.

To show that each power basis element can be written as a linear combination of Bernstein Polynomials, we use the degree elevation formulas and induction to calculate:

$$\begin{aligned}
t^k &= t(t^{k-1}) \\
&= t \sum_{i=k-1}^{n} \frac{\binom{i}{k-1}}{\binom{n}{k-1}} B_{i,n-1}(t) \\
&= \sum_{i=k}^{n} \frac{\binom{i-1}{k-1}}{\binom{n-1}{k-1}} t B_{i-1,n-1}(t) \\
&= \sum_{i=k-1}^{n-1} \frac{\binom{i}{k-1}}{\binom{n}{k-1}} \frac{i}{n} B_{i,n}(t) \\
&= \sum_{i=k-1}^{n-1} \frac{\binom{i}{k}}{\binom{n}{k}} B_{i,n}(t),
\end{aligned}$$

where the induction hypothesis was used in the second step.

Derivatives

Derivatives of the nth degree Bernstein polynomials are polynomials of degree $n-1$. Using the definition of the Bernstein polynomial we can show that this derivative can be written as a linear combination of Bernstein polynomials. In particular

$$\frac{d}{dt} B_{k,n}(t) \;=\; n(B_{k-1,n-1}(t) - B_{k,n-1}(t))$$

for $0 \leq k \leq n$. This can be shown by direct differentiation

$$\begin{aligned}
\frac{d}{dt} B_{k,n}(t) &= \frac{d}{dt} \binom{n}{k} t^k (1-t)^{n-k} \\
&= \frac{kn!}{k!(n-k)!} t^{k-1}(1-t)^{n-k} + \frac{(n-k)n!}{k!(n-k)!} t^k (1-t)^{n-k-1} \\
&= \frac{n(n-1)!}{(k-1)!(n-k)!} t^{k-1}(1-t)^{n-k} + \frac{n(n-1)!}{k!(n-k-1)!} t^k (1-t)^{n-k-1} \\
&= n \left(\frac{(n-1)!}{(k-1)!(n-k)!} t^{k-1}(1-t)^{n-k} + \frac{(n-1)!}{k!(n-k-1)!} t^k (1-t)^{n-k-1} \right) \\
&= n \left(B_{k-1,n-1}(t) - B_{k,n-1}(t) \right)
\end{aligned}$$

That is, the derivative of a Bernstein polynomial can be expressed as the degree of the polynomial, multiplied by the difference of two Bernstein polynomials of degree $n-1$.

The Bernstein Polynomials as a Basis

Why do the Bernstein polynomials of order n form a basis for the space of polynomials of degree less than or equal to n?

1. They span the space of polynomials – any polynomial of degree less than or equal to n can be written as a linear combination of the Bernstein polynomials.

   This is easily seen if one realizes that The power basis spans the space of polynomials and any member of the power basis can be written as a linear combination of Bernstein polynomials.

2. They are linearly independent – that is, if there exist constants $c_0, c_1, ..., c_n$ so that the identity

   $$0 \;=\; c_0 B_{0,n}(t) + c_1 B_{1,n}(t) + \cdots + c_n B_{n,n}(t)$$

   holds for all t, then all the c_i's must be zero.

   If this were true, then we could write

   $$\begin{aligned} 0 &= c_0 B_{0,n}(t) + c_1 B_{1,n}(t) + \cdots + c_n B_{n,n}(t) \\ &= c_0 \sum_{i=0}^{n} (-1)^i \binom{n}{i}\binom{i}{0} t^i + c_1 \sum_{i=1}^{n} (-1)^{i-1} \binom{n}{i}\binom{i}{1} t^i + \cdots + c_n \sum_{i=n}^{n} (-1)^{i-n} \binom{n}{i}\binom{i}{n} t^i \\ &= c_0 + \left[\sum_{i=0}^{1} c_i \binom{n}{1}\binom{1}{1}\right] t^1 + \cdots + \left[\sum_{i=0}^{n} c_i \binom{n}{n}\binom{n}{n}\right] t^n \end{aligned}$$

   Since the power basis is a linearly independent set, we must have that

   $$\begin{aligned} c_0 &= 0 \\ \sum_{i=0}^{1} c_i \binom{n}{1}\binom{1}{1} &= 0 \\ &\vdots \\ \sum_{i=0}^{n} c_i \binom{n}{n}\binom{n}{n} &= 0 \end{aligned}$$

   which implies that $c_0 = c_1 = \cdots = c_n = 0$ (c_0 is clearly zero, substituting this in the second equation gives $c_1 = 0$, substituting these two into the third equation gives ...)

A Matrix Representation for Bernstein Polynomials

In many applications, a matrix formulation for the Bernstein polynomials is useful. These are straightforward to develop if one only looks at a linear combination in terms of dot products.

Given a polynomial written as a linear combination of the Bernstein basis functions

$$B(t) \;=\; c_0 B_{0,n}(t) + c_1 B_{1,n}(t) + \cdots + c_n B_{n,n}(t)$$

It is easy to write this as a dot product of two vectors

$$B(t) \;=\; \begin{bmatrix} B_{0,n}(t) & B_{1,n}(t) & \cdots & B_{n,n}(t) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix}$$

We can convert this to

$$B(t) \;=\; \begin{bmatrix} 1 & t & t^2 & \cdots & t^n \end{bmatrix} \begin{bmatrix} b_{0,0} & 0 & 0 & \cdots & 0 \\ b_{1,0} & b_{1,1} & 0 & \cdots & 0 \\ b_{2,0} & b_{2,1} & b_{2,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{n,0} & b_{n,1} & b_{n,2} & \cdots & b_{n,n} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

where the $b_{i,j}$ are the coefficients of the power basis that are used to determine the respective Bernstein polynomials. We note that the matrix in this case is lower triangular.

In the quadratic case ($n = 2$), the matrix representation is

$$B(t) \;=\; \begin{bmatrix} 1 & t & t^2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \end{bmatrix}$$

and in the cubic case ($n = 3$), the matrix representation is

$$B(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

On-Line Geometric Modeling Notes

A DIVIDE AND CONQUER METHOD FOR CURVE DRAWING

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

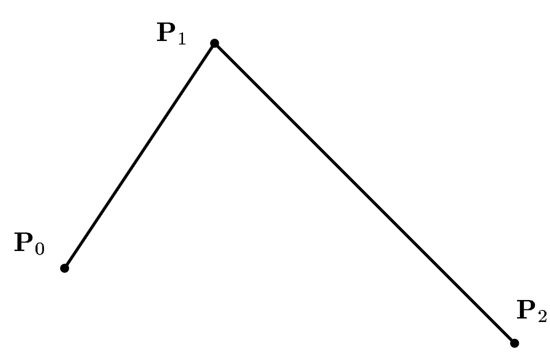
Overview

In the late 1960s, two European engineers independently developed a mathematical curve formulation which was extremely useful for modeling and design and also easily adaptable to use on a computer system. The primary feature of this method was that the controlling parameters of the curve were simply points in three-dimensional space, and each of these points had an influence on the curve. This curve, commonly called the Bézier curve, is the representation that is most frequently used in computer graphics and geometric modeling.

We present in these notes a form of the Bézier curve which can be developed through a simple divide-and-conquer, or subdivision method. This will not give us a rigorous definition of the Bézier curve, but will serve as motivation as it follows the general construction procedure for the curve.

The Subdivision Procedure

This curve is defined by using three control points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$. Whereas these points can be arbitrarily placed in three-dimensional space, we will illustrate the algorithm with the points below:

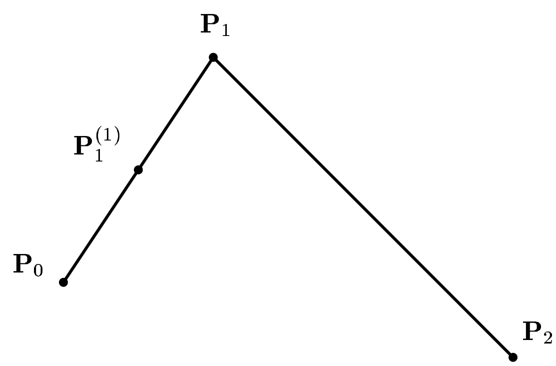


We will develop a method that uses these points to construct a curve. The curve will pass through the points $\mathbf{P}_0$ and $\mathbf{P}_2$ and will lie within the triangle $\triangle\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$. $\mathbf{P}_1$ will be a control point that serves as a "handle" or a "influence" on the curve. Our general procedure will split the curve into two segments, each of which is again specified by three control points. With this procedure, we can recursively generate many small segments of the curve, which can be eventually approximated by straight lines when the curve is to be drawn. The procedure is quite simple, the most complicated mathematics being the calculation of midpoints of the lines connecting control points.

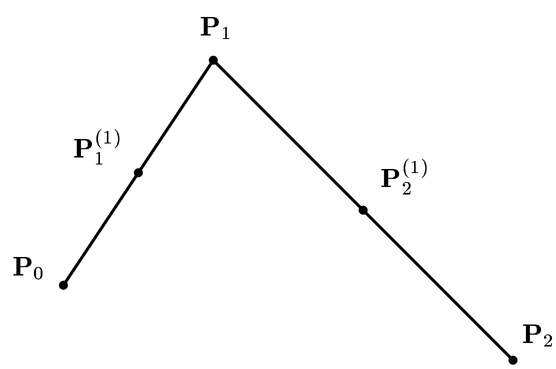
The Basic Subdivision Procedure

The procedure to subdivide the curve into two segments can be described as follows:

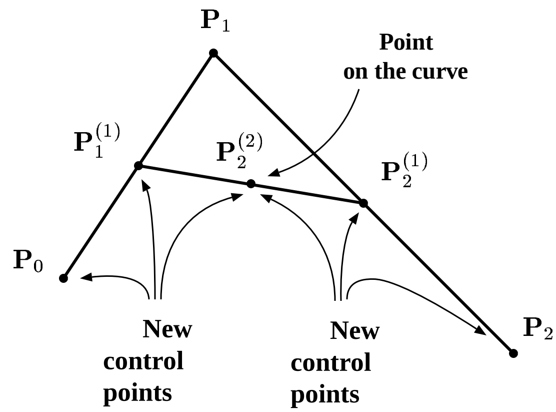
- First, let $\mathbf{P}_1^{(1)}$ be the midpoint of the segment $\overline{\mathbf{P}_0\mathbf{P}_1}$,



- then, let $\mathbf{P}_2^{(1)}$ be the midpoint of segment $\overline{\mathbf{P}_1\mathbf{P}_2}$,



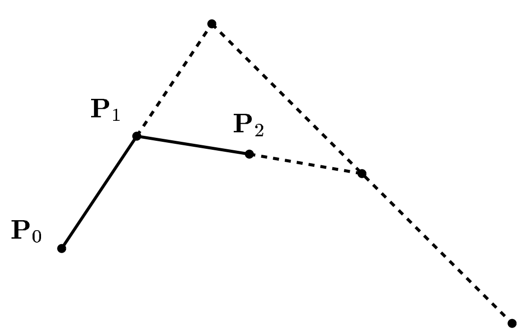
- and finally let $\mathbf{P}_2^{(2)}$ be the midpoint of the segment $\overline{\mathbf{P}_1^{(1)}\mathbf{P}_2^{(1)}}$.



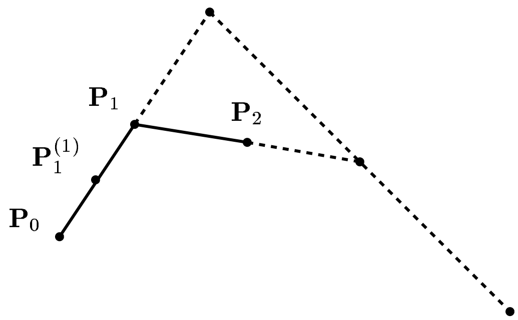
We define $\mathbf{P}_2^{(2)}$ to be a point on the curve. Note that we have created two new sets control points ($\left\{\mathbf{P}_0, \mathbf{P}_1^{(1)}, \mathbf{P}_2^{(2)}\right\}$ and $\left\{\mathbf{P}_2^{(2)}, \mathbf{P}_2^{(1)}, \mathbf{P}_2\right\}$) which can be use to define the first and second portions of the subdivided curve, respectively. We now have define an additional point on the curve and two new sets of three control points that can be used to continue subdividing the curve.

Continuing the Subdivision

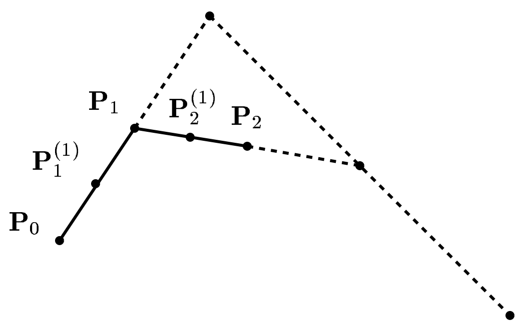
Performing the procedure again, we use the control points $\left\{\mathbf{P}_0, \mathbf{P}_1^{(1)}, \mathbf{P}_2^{(2)}\right\}$, relabeling them for convenience as $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$ respectively, and apply our procedure



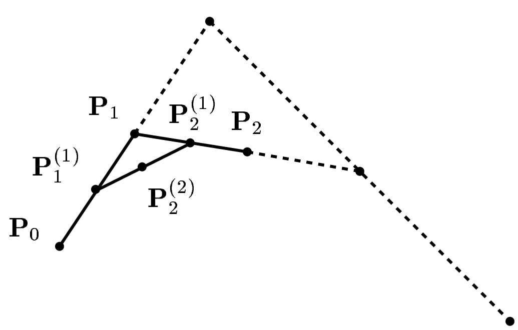
- First let $\mathbf{P}_1^{(1)}$ be the midpoint of the segment $\overline{\mathbf{P}_0\mathbf{P}_1}$.



- then let $\mathbf{P}_2^{(1)}$ be the midpoint of segment $\overline{\mathbf{P}_1\mathbf{P}_2}$.

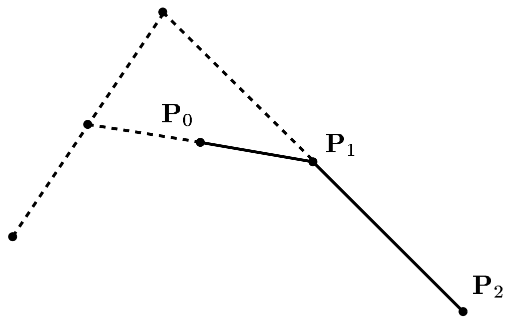


- and finally let $\mathbf{P}_2^{(2)}$ as the midpoint of the segment $\overline{\mathbf{P}_1^{(1)}\mathbf{P}_2^{(1)}}$.

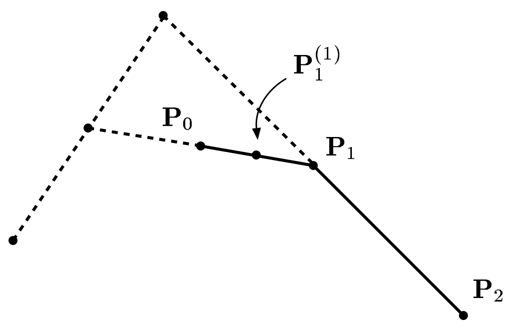


We now define $\mathbf{P}_2^{(2)}$ to be a point on the curve. This process produces another point on the curve, and creates two new sets of control points as was the case before.

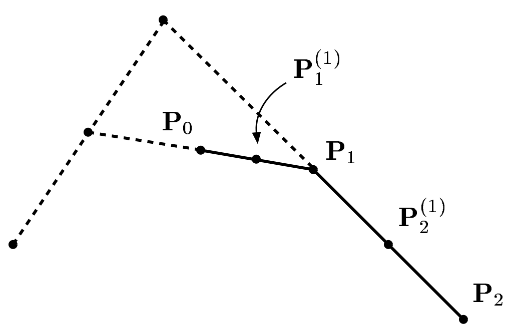
If we consider the control points $\mathbf{P}_2^{(2)}$, $\mathbf{P}_2^{(1)}$, and $\mathbf{P}_2$, generated in the first subdivision, and relabel them as $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$ respectively, we can again apply the subdivision procedure



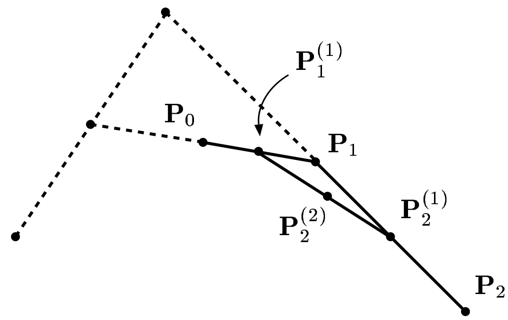
- $\mathbf{P}_1^{(1)}$ as the midpoint of the segment $\overline{\mathbf{P}_0\mathbf{P}_1}$.



- $\mathbf{P}_2^{(1)}$ as the midpoint of segment $\overline{\mathbf{P}_1\mathbf{P}_2}$.



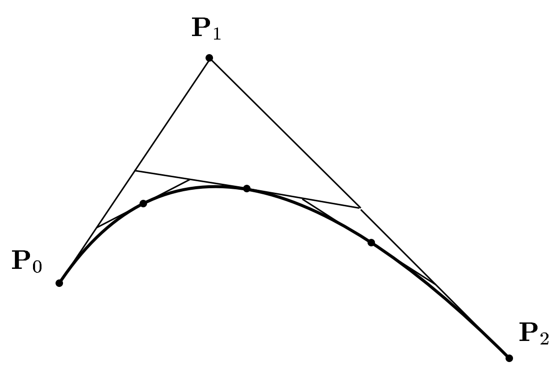
- $\mathbf{P}_2^{(2)}$ as the midpoint of the segment $\overline{\mathbf{P}_1^{(1)}\mathbf{P}_2^{(1)}}$.



We now have $\mathbf{P}_2^{(2)}$ on the curve.

The Subdivision Algorithm

Three points have now been generated on the curve and four subcurves have been generated. The final curve, together with the points generated, is shown as follows:



You should now see how to proceed. At each step the process creates both a point on the curve and two new sets of control points. This effectively subdivides the curve into two new curve segments, each of which can be handled separately.

Summary

This is a somewhat unique method to define a curve, and probably not previously seen by many students. It is a *geometric* method, as it uses only the midpoint formula as it's fundamental tool. It uses the basic computer science paradigm of (sub)divide and conquer to calculate points on the curve. The curve can be “drawn” using computer graphics by calculating a somewhat-dense set of points, and connecting them with straight lines.

The curve drawn by this method is a quadratic Bézier curve.

QUADRATIC BÉZIER CURVES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

The Bézier curve representation is one that is utilized most frequently in computer graphics and geometric modeling. The curve is defined geometrically, which means that the parameters have geometric meaning – they are just points in three-dimensional space. It was developed by two competing European engineers in the late 1960s to attempt to draw automotive components.

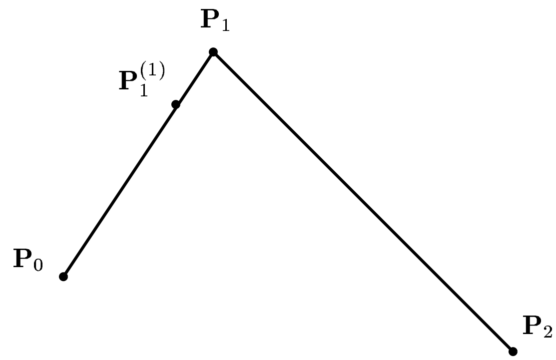
In these notes, we develop the quadratic Bézier curve. This curve can be developed through a divide-and-conquer approach whose basic operation is the generation of midpoints on the curve. However, this time we develop the curve by calculating points other than midpoints – resulting in a useful parameterization for the curve.

Development of the Quadratic Bézier Curve

Given three control points $\mathbf{P}_0$, $\mathbf{P}_1$ and $\mathbf{P}_2$ we develop a divide procedure that is based upon a parameter t, which is a number between 0 and 1 (the illustrations utilize the value $t = .75$). This proceeds as follows:

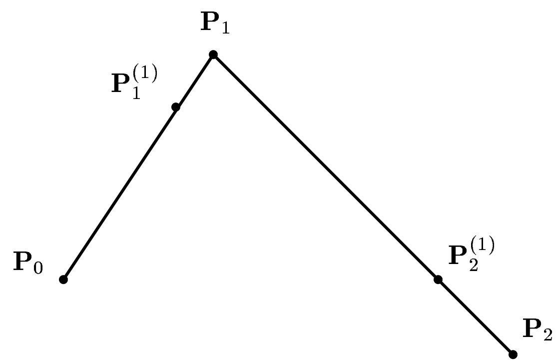
- First let $\mathbf{P}_1^{(1)}$ be the point on the segment $\overline{\mathbf{P}_0\mathbf{P}_1}$ defined by

$$\mathbf{P}_1^{(1)} = (1-t)\mathbf{P}_0 + t\mathbf{P}_1 \; (= \; \mathbf{P}_0 + t(\mathbf{P}_1 - \mathbf{P}_0))$$



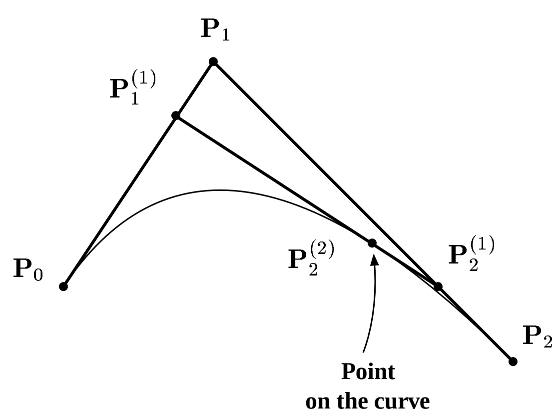
- then let $\mathbf{P}_2^{(1)}$ be the point on the segment $\overline{\mathbf{P}_1\mathbf{P}_2}$ defined by

$$\mathbf{P}_2^{(1)} = (1-t)\mathbf{P}_1 + t\mathbf{P}_2$$



- and finally let $\mathbf{P}_2^{(2)}$ be the point on the segment $\overline{\mathbf{P}_1^{(1)}\mathbf{P}_2^{(1)}}$ defined by

$$\mathbf{P}_2^{(2)} = (1-t)\mathbf{P}_1^{(1)} + t\mathbf{P}_2^{(1)}$$



- Define $\mathbf{P}(t) = \mathbf{P}_2^{(2)}$.

This is a similar procedure to the divide-and-conquer method in that geometric means are used to define points on the curve. Each time a new point is calculated, the control points are subdivided into two sets, each of which may be use to generate new subcurves. The method is identical to the divide-and-conquer method in the case $t = \frac{1}{2}$.

Developing the Equation of the Curve

There is a different way of looking at this procedure – because there is a parameter involved. Each one of the points $\mathbf{P}_1^{(1)}$, $\mathbf{P}_2^{(1)}$, and $\mathbf{P}_2^{(2)}$ is really a function of the parameter t – and $\mathbf{P}_2^{(2)}$ can be equated with $\mathbf{P}(t)$ since it is a point on the curve that corresponds to the parameter value t. In this way, $\mathbf{P}(t)$ becomes a functional representation of the Bézier curve.

Writing down the algebra, we see that

$$\begin{aligned}\mathbf{P}(t) &= \mathbf{P}_2^{(2)}(t)\\ &= (1-t)\mathbf{P}_1^{(1)}(t) + t\mathbf{P}_2^{(1)}(t)\end{aligned}$$

where

$$\begin{aligned}\mathbf{P}_1^{(1)}(t) &= (1-t)\mathbf{P}_0 + t\mathbf{P}_1, \text{ and}\\ \mathbf{P}_2^{(1)}(t) &= (1-t)\mathbf{P}_1 + t\mathbf{P}_2\end{aligned}$$

(Note that we have now denoted $\mathbf{P}_1^{(1)}$ and $\mathbf{P}_2^{(1)}$ as functions of t.) Substituting these two equations back into the original, we have

$$\begin{aligned}\mathbf{P}(t) &= (1-t)\mathbf{P}_1^{(1)}(t) + t\mathbf{P}_2^{(1)}(t)\\ &= (1-t)\left[(1-t)\mathbf{P}_0 + t\mathbf{P}_1\right] + t\left[(1-t)\mathbf{P}_1 + t\mathbf{P}_2\right]\\ &= (1-t)^2\mathbf{P}_0 + (1-t)t\mathbf{P}_1 + t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2\\ &= (1-t)^2\mathbf{P}_0 + 2t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2\end{aligned}$$

This is quadratic polynomial (as it is a linear combination of quadratic polynomials), and therefore it is a parabolic segment. Thus, the quadratic Bézier curve is simply a parabolic curve.

Properties of the Quadratic Curve

The quadratic Bézier curve has the following properties, which can be easily verified.

1. $\mathbf{P}(0) = \mathbf{P}_0$ and $\mathbf{P}(1) = \mathbf{P}_2$, so the curve passes through the control points $\mathbf{P}_0$ and $\mathbf{P}_2$.

2. The curve $\mathbf{P}(t)$ is continuous and has continuous derivatives of all orders. (This is automatic for a polynomial.)

3. We can differentiate $\mathbf{P}(t)$ with respect to t and obtain

$$\begin{aligned}\frac{d}{dt}\mathbf{P}(t) &= -2(1-t)\mathbf{P}_0 + [-2t + 2(1-t)]\,\mathbf{P}_1 + 2t\mathbf{P}_2 \\ &= 2\left[(1-t)\left(\mathbf{P}_1 - \mathbf{P}_0\right) + t\left(\mathbf{P}_2 - \mathbf{P}_1\right)\right]\end{aligned}$$

   Thus $\frac{d}{dt}\mathbf{P}(0) = 2(\mathbf{P}_1 - \mathbf{P}_0)$, is the tangent vector at $t = 0$ and $\frac{d}{dt}\mathbf{P}(1) = 2(\mathbf{P}_2 - \mathbf{P}_1)$ is the tangent vector at $t = 1$. This implies that the slope of the curve at $t = 0$ is the same as that of the vector $2(\mathbf{P}_1 - \mathbf{P}_0)$ and the slope of the curve at $t = 1$ is the same as that of the vector $2(\mathbf{P}_2 - \mathbf{P}_1)$.

4. The functions $(1-t)^2$, $2t(1-t)$, and t^2 that are used to "blend" the control points $\mathbf{P}_0$, $\mathbf{P}_1$ and $\mathbf{P}_2$ together are the degree-2 Bernstein Polynomials They are all non-negative functions and sum to one. Clearly

$$(1-t)^2 + 2t(1-t) + t^2 \;=\; 1 - 2t + t^2 + 2t - 2t^2 + t^2 \;=\; 1$$

5. The curve is contained within the triangle $\triangle\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$.

   Since the blending functions are non-negative and add to one, $P(t)$ is an affine combination of the points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$. Thus $\mathbf{P}(t)$ must lie in the convex hull of the control points for all $0 \leq t \leq 1$. The convex hull of a triangle is the triangle itself.

6. If the points $\mathbf{P}_0$, $\mathbf{P}_1$ and $\mathbf{P}_2$ are colinear, then the curve is a straight line.

   If the points are colinear, then the convex hull is a straight line, and the curve must lie within the convex hull.

7. The process of calculating one $\mathbf{P}(t)$ subdivides the control points into two sets $\left\{\mathbf{P}_0, \mathbf{P}_1^{(1)}(t), \mathbf{P}_2^{(2)}(t)\right\}$, and $\left\{\mathbf{P}_2^{(2)}(t), \mathbf{P}_2^{(1)}(t), \mathbf{P}_2\right\}$, each of which can be used to define another curve, as in our subdivision process above.

8. All the points, generated from the divide-and-conquer method, lie on this curve.

   Clearly $\mathbf{P}(\frac{1}{2})$ is the first point calculated by the divide and conquer method.

   Lets show that $\mathbf{P}(\frac{1}{4})$ is exactly the point obtained by performing the divide-and-conquer method, on the control points $\mathbf{Q}_0 = \mathbf{P}_0$, $\mathbf{Q}_2 = \mathbf{P}_1^{(1)}(\frac{1}{2})$ and $\mathbf{Q}_2 = \mathbf{P}_2^{(2)}(\frac{1}{2})$ which were generated in the first step of the divide-and-conquer method. If we call this point $\mathbf{Q}$, then by the divide-and-conquer method

$$\begin{aligned}\mathbf{Q} &= \frac{1}{2}\mathbf{Q}_1^{(1)} + \frac{1}{2}\mathbf{Q}_2^{(1)} \\ &= \frac{1}{2}\left[\frac{1}{2}\mathbf{Q}_0 + \frac{1}{2}\mathbf{Q}_1\right] + \frac{1}{2}\left[\frac{1}{2}\mathbf{Q}_1 + \frac{1}{2}\mathbf{Q}_2\right] \\ &= \frac{1}{4}\mathbf{Q}_0 + \frac{1}{2}\mathbf{Q}_1 + \frac{1}{4}\mathbf{Q}_2\end{aligned}$$

   and by substituting for the $\mathbf{Q}$s, and similifying

$$\begin{aligned}\mathbf{Q} &= \frac{1}{4}\mathbf{P}_0 + \frac{1}{2}\mathbf{P}_1^{(1)}(t) + \frac{1}{4}\mathbf{P}_2^{(2)}(t) \\ &= \frac{1}{4}\mathbf{P}_0 + \frac{1}{2}\mathbf{P}_1^{(1)}(t) + \frac{1}{4}\left[\frac{1}{2}\mathbf{P}_1^{(1)}(t) + \frac{1}{2}\mathbf{P}_2^{(1)}(t)\right] \\ &= \frac{1}{4}\mathbf{P}_0 + \frac{3}{8}\mathbf{P}_1^{(1)}(t) + \frac{1}{8}\mathbf{P}_2^{(1)}(t) \\ &= \frac{1}{4}\mathbf{P}_0 + \frac{5}{8}\left[\frac{1}{2}\mathbf{P}_0 + +\frac{1}{2}\mathbf{P}_1\right] + \frac{1}{8}\left[\frac{1}{2}\mathbf{P}_1 + +\frac{1}{2}\mathbf{P}_2\right] \\ &= \frac{9}{16}\mathbf{P}_0 + \frac{3}{8}\mathbf{P}_1 + \frac{1}{16}\mathbf{P}_2\end{aligned}$$

   If we calculate $\mathbf{P}(t)$ with $t = \frac{1}{4}$, we have

$$\begin{aligned}\mathbf{P}(\frac{1}{4}) &= (1-t)^2\mathbf{P}_0 + 2t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2 \\ &= \frac{9}{16}\mathbf{P}_0 + \frac{3}{8}\mathbf{P}_1 + \frac{1}{16}\mathbf{P}_2\end{aligned}$$

   So $\mathbf{P}(\frac{1}{4}$ is exactly the point constructed in from the divide-and-conquer algorithm. Similar calculations exist for all other points generated in the divide and conquer method – each point generated by the method will correspond to one with a corresponding parameter of $\frac{k}{2^n}$ for some k and n.

Summarizing the Development of the Curve

We now have two methods by which we can generate points on the curve. The first of which is geometrically based – points are found on the curve by selecting successive points on line segments. The other is an analytic formula, which expresses the curve in functional notation.

- **The Geometrical Construction Method** – Given points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$, we can construct a curve $\mathbf{P}(t)$ by the following construction

$$\mathbf{P}(t) \;=\; \mathbf{P}_2^{(2)}(t)$$

  where

$$\mathbf{P}_i^{(j)}(t) = \begin{cases} (1-t)\mathbf{P}_{i-1}^{(j-1)}(t) + t\mathbf{P}_i^{(j-1)}(t) & \text{if } j > 0 \\ \mathbf{P}_i & \text{if } \; j = 0 \end{cases}$$

  for $t \in [0, 1]$.

- **The Analytical Formula** – Given points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$, we can construct a curve $\mathbf{P}(t)$ by the following

$$\mathbf{P}(t) \;=\; (1-t)^2\mathbf{P}_0 + 2t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2$$

  for $t \in [0, 1]$.

Summary

The quadratic curve serves as a good example for discussing the development of the Bézier curve, but really only generates parabolas. This eliminates the curve for many applications where smooth curves with inflection points are necessary. The problem can be addressed by performing exactly the same steps as above, but utilizing the procedure on four control points – resulting in the cubic Bézier curve.

On-Line Geometric Modeling Notes

CUBIC BÉZIER CURVES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

The Bézier curve representation is one that is utilized most frequently in computer graphics and geometric modeling. The curve is defined geometrically, which means that the parameters have geometric meaning – they are just points in three-dimensional space. It was developed by two competing European engineers in the late 1960s to attempt to draw automotive components.

In these notes, we develop the cubic Bézier curve. This curve can be developed through a divide-and-conquer approach similar to the quadratic curve However, in these notes, we will develop a parameterized version of the curve which proceeds almost identically to the development for the quadratic Bézier curve

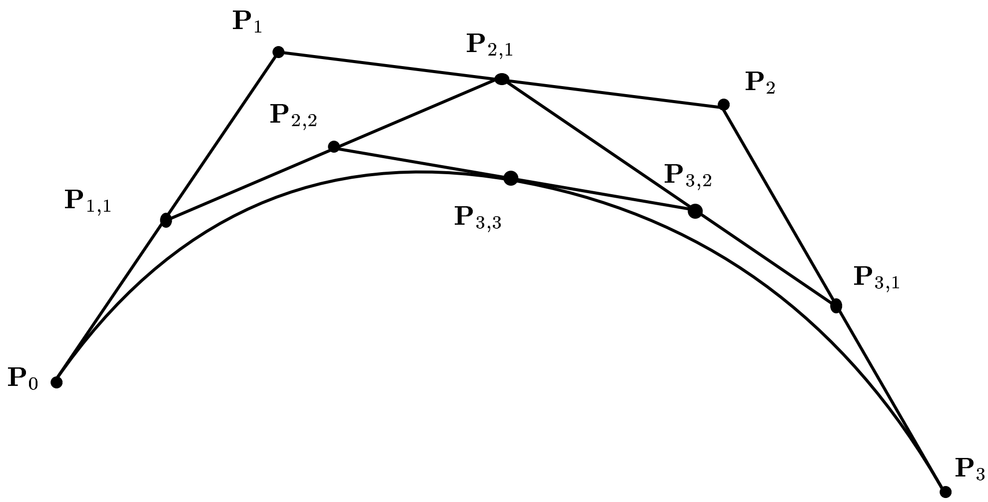
---

Defining The Cubic Bézier Curve

Given four control points, $\mathbf{P}_0,\ \mathbf{P}_1,\ \mathbf{P}_2,\ \mathbf{P}_3$, one can generate a curve $\mathbf{P}(t)$, as we did in the case of the quadratic Bézier curve, by

- let $\mathbf{P}_1^{(1)}(t) \ = \ t\mathbf{P}_1 + (1-t)\mathbf{P}_0$
- let $\mathbf{P}_2^{(1)}(t) \ = \ t\mathbf{P}_2 + (1-t)\mathbf{P}_1$
- let $\mathbf{P}_3^{(1)}(t) \ = \ t\mathbf{P}_3 + (1-t)\mathbf{P}_2$
- let $\mathbf{P}_2^{(2)}(t) \ = \ t\mathbf{P}_2^{(1)}(t) + (1-t)\mathbf{P}_1^{(1)}(t)$
- let $\mathbf{P}_3^{(2)}(t) \ = \ t\mathbf{P}_3^{(1)}(t) + (1-t)\mathbf{P}_2^{(1)}(t)$
- let $\mathbf{P}_3^{(3)}(t) \ = \ t\mathbf{P}_3^{(2)}(t) + (1-t)\mathbf{P}_2^{(2)}(t)$
- $\mathbf{P}_3^{(3)}(t)$ is defined to be $\mathbf{P}(t)$

This construction is shown in the figure below



notice that we did the same process as in the quadratic Bézier curve, but did one additional level. The procedure, as in the quadratic case, produces a point on the curve and subdivides the curve by producing 2 new sets of 4 control points.

Simplifying the above construction, we have

$$
\begin{aligned}
\mathbf{P}(t) &= \mathbf{P}_3^{(3)}(t) \\
&= t\mathbf{P}_3^{(2)}(t) + (1-t)\mathbf{P}_2^{(2)}(t) \\
&= t\left[t\mathbf{P}_3^{(1)}(t) + (1-t)\mathbf{P}_2^{(1)}(t)\right] \\
&\quad + \ (1-t)\left[t\mathbf{P}_2^{(1)}(t) + (1-t)\mathbf{P}_1^{(1)}(t)\right] \\
&= t^2\mathbf{P}_3^{(1)}(t) + 2t(1-t)\mathbf{P}_2^{(1)}(t) + (1-t)^2\mathbf{P}_1^{(1)}(t) \\
&= t^2\left[t\mathbf{P}_3 + (1-t)\mathbf{P}_2\right] + 2t(1-t)\left[t\mathbf{P}_2 + (1-t)\mathbf{P}_1\right] \\
&\quad + \ (1-t)^2\left[t\mathbf{P}_1 + (1-t)\mathbf{P}_0\right] \\
&= t^3\mathbf{P}_3 + 3t^2(1-t)\mathbf{P}_2 + 3t(1-t)^2\mathbf{P}_1 + (1-t)^3\mathbf{P}_0
\end{aligned}
$$

which is the analytic form of the curve.

Summarizing the Development of the Curve

As in the quadratic case, we have developed two methods for generating points on the curve.

- **The Geometric Method** – Given the control points $\mathbf{P}_0,\ \mathbf{P}_1,\ \mathbf{P}_2,\ \mathbf{P}_3$, and a value $t \in [0,1]$, we can generate the point $\mathbf{P}(t)$ on the Bézier curve by

$$\mathbf{P}(t) \ = \ \mathbf{P}_3^{(3)}(t)$$

where

$$\mathbf{P}_i^{(j)}(t) \ = \ \begin{cases} (1-t)\mathbf{P}_{i-1}^{(j-1)}(t) \ + \ t\mathbf{P}_i^{(j-1)}(t) & \text{if } j \ > \ 0 \\ \mathbf{P}_i & \text{otherwise} \end{cases}$$

- **The Analytic Method** – Given the control points $\mathbf{P}_0,\ \mathbf{P}_1,\ \mathbf{P}_2,\ \mathbf{P}_3$, we define the Bézier curve to be

$$\mathbf{P}(t) \ = \ \sum_{i=0}^{3} \mathbf{P}_i B_{i,3}(t)$$

where

$$\begin{aligned} B_{0,3}(t) &= (1-t)^3 \\ B_{1,3}(t) &= 3t(1-t)^2 \\ B_{2,3}(t) &= 3t^2(1-t) \\ B_{3,3}(t) &= t^3 \end{aligned}$$

the Bernstein polynomials of degree three.

Properties of the Cubic Bézier Curve

The cubic Bézier curve has properties similar to that of the quadratic curve. These can be verified directly from the equations above.

- $\mathbf{P}_0$ and $\mathbf{P}_3$ are on the curve.

- The curve is continuous, infinitely differentiable, and the second derivatives are continuous (automatic for a polynomial curve).
- The tangent line to the curve at the point $\mathbf{P}_0$ is the line $\overline{\mathbf{P}_0\mathbf{P}_1}$. The tangent to the curve at the point $\mathbf{P}_3$ is the line $\overline{\mathbf{P}_2\mathbf{P}_3}$.
- The curve lies within the convex hull of its control points. This is because each successive $\mathbf{P}_i^{(j)}$ is a convex combination of the points $\mathbf{P}_i^{(j-1)}$ and $\mathbf{P}_{i-1}^{(j-1)}$.
- Both $\mathbf{P}_1$ and $\mathbf{P}_2$ are on the curve only if the curve is linear.

Summary

The procedure for developing the cubic Bézier curve is nearly identical to that for the quadratic curve – the primary difference is that we have four control points and must proceed one additional level in the recursion to get a point on the curve. This procedure is extendable so that Bézier curves can be developed for any number of control points.

A MATRIX FORMULATION OF THE CUBIC BÉZIER CURVE

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

A cubic Bézier curve has a useful representation in a matrix form. This is a non-standard representation but extremely valuable if we can multiply matrices quickly. The matrix which we develop, when examined closely, is uniquely defined by the cubic Bernstein polynomials. We can use this form to develop "subdivision matrices" that allow us to use matrix multiplication to generate different Bézier control polygons for the cubic curve.

Developing the Matrix Equation

A cubic Bézier Curve can be written in a matrix form by expanding the analytic definition of the curve into its Bernstein polynomial coefficients, and then writing these coefficients in a matrix form using the polynomial power basis. That is,

$$\begin{aligned}
\mathbf{P}(t) &= \sum_{i=0}^{3} \mathbf{P}_i B_i(t) \\
&= (1-t)^3\mathbf{P}_0 + \, 3t(1-t)^2\mathbf{P}_1 + \, 3t^2(1-t)\mathbf{P}_2 + \, t^3\mathbf{P}_3 \\
&= \left[\begin{array}{cccc} (1-t)^3 & 3t(1-t)^2 & 3t^2(1-t) & t^3 \end{array} \right] \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array} \right] \\
&= \left[\begin{array}{cccc} 1 & t & t^2 & t^3 \end{array} \right] \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{array} \right] \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array} \right]
\end{aligned}$$

and so a cubic Bézier curve is can be written in a matrix form of

$$\left[\begin{array}{cccc} 1 & t & t^2 & t^3 \end{array} \right] M \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array} \right]$$

where

$$M \; = \; \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{array} \right]$$

The matrix M defines the blending functions for the curve $\mathbf{P}(t)$ – i.e. the cubic Bernstein polynomials. In reality there are three equations here, one for each of the x, y and z components of $\mathbf{P}(t)$.

Utilizing equipment that is designed for fast 4×4 matrix calculations, this formulation can be used to quickly calculate points on the curve.

Subdivision Using the Matrix Form

Suppose we wish to generate the control polygon for the portion of the curve $\mathbf{P}(t)$ where t ranges between 0 and $\frac{1}{2}$ – subdivide the curve at the point $t = \frac{1}{2}$. This can be done by defining a new curve $\mathbf{Q}(t)$ which is equal to $\mathbf{P}(\frac{t}{2})$. Clearly this new curve is a cubic polynomial, and traces out the desired portion of $\mathbf{P}$ as t ranges between 0 and 1. We can calculate the Bézier control polygon for $\mathbf{Q}$ by using the matrix form of the curve $\mathbf{P}$.

$$
\begin{aligned}
\mathbf{Q}(t) &= \mathbf{P}(\frac{t}{2}) \\
&= \begin{bmatrix} 1 & (\frac{t}{2}) & (\frac{t}{2})^2 & (\frac{t}{2})^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} M S_{[0,\frac{1}{2}]} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}
\end{aligned}
$$

where the matrix $S_{[0,\frac{1}{2}]}$ is defined as

$$\begin{aligned}
S_{[0,\frac{1}{2}]} &= M^{-1}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} M \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & 0 & 0 \\ 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{6} & 0 & 0 \\ 1 & \frac{1}{3} & \frac{1}{12} & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{bmatrix}
\end{aligned}$$

So $\mathbf{Q}(t)$ is a Bézier curve, with a control polygon given by

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{P}_0 \\ \frac{1}{2}\mathbf{P}_1 + \frac{1}{2}\mathbf{P}_0 \\ \frac{1}{4}\mathbf{P}_2 + \frac{1}{2}\mathbf{P}_1 + \frac{1}{4}\mathbf{P}_0 \\ \frac{1}{8}\mathbf{P}_3 + \frac{3}{8}\mathbf{P}_2 + \frac{3}{8}\mathbf{P}_1 + \frac{1}{8}\mathbf{P}_0 \end{bmatrix}$$

In the same way, we can obtain the Bézier control polygon for the second half of the curve – the portion where t ranges between $\frac{1}{2}$ and 1. If we call this new curve $\mathbf{Q}(t)$, then

$$
\begin{aligned}
\mathbf{Q}(t) &= \mathbf{P}(\frac{1}{2}+\frac{t}{2}) \\
&= \left[\begin{array}{cccc} 1 & (\frac{1}{2}+\frac{t}{2}) & (\frac{1}{2}+\frac{t}{2})^2 & (\frac{1}{2}+\frac{t}{2})^3 \end{array}\right] \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{array}\right] \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array}\right] \\
&= \left[\begin{array}{cccc} 1 & t & t^2 & t^3 \end{array}\right] \left[\begin{array}{cccc} 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} & \frac{3}{8} \\ 0 & 0 & \frac{1}{4} & \frac{3}{8} \\ 0 & 0 & 0 & \frac{1}{8} \end{array}\right] \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{array}\right] \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array}\right] \\
&= \left[\begin{array}{cccc} 1 & t & t^2 & t^3 \end{array}\right] M S_{[\frac{1}{2},1]} \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array}\right]
\end{aligned}
$$

where

$$
S_{[\frac{1}{2},1]} = \left[\begin{array}{cccc} \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{array}\right]
$$

obtaining a matrix that can be applied to the original Bézier control points to produce Bézier control points for the second half of the curve.

Generating a Sequence of Bézier Control Polygons.

Using matrix calculations similar to those above, we can generate an iterative scheme to generate a sequence of points on the curve. To do this, we need one additional S matrix. If we consider the portion of the cubic curve $\mathbf{P}(t)$ where t ranges between 1 and 2, We generate the Bézier control points of $\mathbf{Q}(t)$ by reparameterization of the original curve – namely by replacing t by $t+1$ – to obtain

$$\begin{aligned}
\mathbf{Q}(t) &= \mathbf{P}(t+1) \\
&= \begin{bmatrix} 1 & (t+1) & (t+1)^2 & (t+1)^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} M S_{[1,2]} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}
\end{aligned}$$

where, after some calculation, $S_{[1,2]}$ is given by

$$S_{[1,2]} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 2 \\ 0 & 1 & -4 & 4 \\ -1 & 6 & -12 & 8 \end{bmatrix}$$

Now, using a combination of $S_{[0,\frac{1}{2}]}$, $S_{[\frac{1}{2},1]}$ and $S_{[1,2]}$, we can produce Bézier control polygons along the curve similar to methods developed with divided differences. To see what I mean here, first notice that

$$S_{[1,2]} S_{[0,\frac{1}{2}]} = S_{[\frac{1}{2},1]}$$

This states that by applying $S_{[0,\frac{1}{2}]}$ to obtain a Bézier control polygon for the first half of the curve, we can then apply $S_{[1,2]}$ to this control polygon to obtain the Bézier control polygon for the second half of the curve.

Extending this, if we apply

$$S^i_{[1,2]} S^k_{[0,\frac{1}{2}]}$$

(that is, apply $S_{[0,\frac{1}{2}]}$ k times and then $S_{[1,2]}$ i times), we obtain the Bézier control polygon for the portion of the curve where t ranges between $\frac{i}{2^k}$ and $\frac{i+1}{2^k}$. By repeatedly applying $S_{[1,2]}$, we move our control polygons along the curve.

Summary

We have developed a matrix form for the cubic Bézier curve. Using reparameterization, we then developed matrices which enabled us to produce Bézier control polygons for sections of the curve, and to move from one Bézier control polygon to an adjacent for on the curve. These operations are extremely useful when utilizing hardware with geometry engines that multiply 4×4 matrices rapidly.

REPARAMETERIZING BÉZIER CURVES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

The Bézier curve is the representation that is most utilized in computer graphics and geometric modeling. This curve is usually defined by a set of control points $\{\mathbf{P}_0, \mathbf{P}_0, ..., \mathbf{P}_n\}$ where

$$\mathbf{P}(t) = \sum_{i=0}^{n} \mathbf{P}_i B_{i,n}(t)$$

for $0 \leq t \leq 1$.

Running the parameter t from 0 to 1 gives a simple analytic and geometric definition of the curve. However, when we wish to examine general B-spline curves, which are piecewise Bézier curves, we will need to vary this parameter over an arbitrary interval. This is actually quite simple, and is discussed in the sections below.

Defining the Reparameterized Curve

Given a Bézier curve $\mathbf{P}(t)$, we can develop a new parameterization of the curve where t ranges between the values a and b by

$$\mathbf{P}_{[a,b]}(t) = \mathbf{P}(\frac{t-a}{b-a})$$

We note that $\mathbf{P}_{[a,b]}$ and $\mathbf{P}(t)$ are exactly the same curve, but traversed through different ranges of t. This change impacts only a few of the Bézier curve properties, namely

- $\mathbf{P}_{[0,1]}(t) = \mathbf{P}(t)$.

- Using the chain rule, the derivative of the curve $\mathbf{P}_{[a,b]}(t)$ at a value t is equal to

$$\frac{1}{b-a}\mathbf{P}'(\frac{t-a}{b-a})$$

- Subdividing the curve $\mathbf{P}_{[a,b]}(t)$ at the point $c \in [a,b]$, is equivalent to subdividing the curve $\mathbf{P}(t)$ at the point $\frac{c-a}{b-a}$.

Summary

The Bézier curve is normally developed by using a parameter that ranges between 0 and 1. By a simple modification, we can reparameterize the curve so that t can range between any two values a and b. The resulting curve algorithms for $\mathbf{P}_{[a,b]}(t)$ can all be related to the algorithms for the originally defined $\mathbf{P}(t)$.

CONTROL POLYGONS FOR CUBIC CURVES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

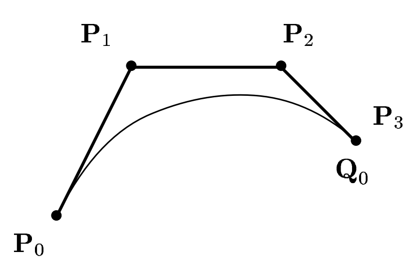
B-Spline curves are piecewise Bézier curves. To develop B-splines, and to do so in a continuous smooth way, we must discover the conditions on which two Bézier curves can be pieced together. To examine this process, we will first consider a single cubic curve and show how to construct the many Bézier control polygons that represent the curve. These control polygons, and their geometric constraints, are paramount in the definition of the B-spline curve.

A Matrix Equation for a Cubic Curve

A cubic polynomial curve $\mathbf{P}(t)$ can be written as a Bézier curve. If we let $\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$ be the control points of the curve, then it can be written as

$$\mathbf{P}(t) = \sum_{i=0}^{3} \mathbf{P}_i B_{i,3}(t)$$

where the $B_{i,3}(t)$ are the cubic Bernstein polynomials. In this representation, $\mathbf{P}_0 = \mathbf{P}(0)$ and $\mathbf{P}_3 = \mathbf{P}(1)$.



The representation of the curve can be written in a matrix form by

$$\begin{aligned}
\mathbf{P}(t) &= \sum_{i=0}^{3} \mathbf{P}_i B_i(t) \\
&= (1-t)^3 \mathbf{P}_0 + \, 3t(1-t)^2 \mathbf{P}_1 + \, 3t^2(1-t)\mathbf{P}_2 + \, t^3 \mathbf{P}_3 \\
&= \left[\begin{array}{cccc} (1-t)^3 & 3t(1-t)^2 & 3t^2(1-t) & t^3 \end{array} \right] \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array} \right] \\
&= \left[\begin{array}{cccc} 1 & t & t^2 & t^3 \end{array} \right] M \left[\begin{array}{c} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{array} \right]
\end{aligned}$$

where

$$M = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{array} \right]$$

The matrix M defines the blending functions for the curve $\mathbf{P}(t)$ – i.e. the cubic Bernstein polynomials. (In reality there are three equations here, one for each of the x, y and z components of $\mathbf{P}(t)$.)

Reparameterization using the Matrix Form

The control polygon $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3\}$ defines the unique cubic curve $\mathbf{P}(t)$, and is most frequently used to represent the curve between $t = 0$ and $t = 1$, where $\mathbf{P}_0 = \mathbf{P}(0)$ and $\mathbf{P}_3 = \mathbf{P}(1)$. However, given an interval $[a, b]$, there exists a unique control polygon $\{\mathbf{Q}_0, \mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3\}$ defining a Bézier curve $\mathbf{Q}(t)$, such that $\mathbf{Q}(0) = \mathbf{Q}_0 = \mathbf{P}(a)$ and $\mathbf{Q}(1) = \mathbf{Q}_3 = \mathbf{P}(b)$. These control polygons, called Bézier polygons can be generated by reparameterization and by manipulating the matrix representation above.

Suppose that we wish to find the Bézier polygon for the portion of the curve $\mathbf{P}(t)$ where $t \in [a, b]$. If we define this new curve as $\mathbf{Q}(t)$, then we can define $\mathbf{Q}(t) = \mathbf{P}((b-a)t + a)$. It is straightforward to check that both $\mathbf{Q}(t)$ and $\mathbf{P}(t)$ are cubic curves, *and represent the same curve*. We can calculate the control points for $\mathbf{Q}(t)$ by using our matrix form, that is

$$\begin{aligned}
\mathbf{Q}(t) &= \mathbf{P}((b-a)t+a) \\
&= \begin{bmatrix} 1 & (b-a)t+a & ((b-a)t+a)^2 & ((b-a)t+a)^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} & & \\ & C & \\ & & \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}
\end{aligned}$$

where the matrix $[C]$ has columns whose entries are the coefficients of 1, t, t^2 and t^3 respectively in the polynomials 1, $(b-a)t+a$, $((b-a)t+a)^2$, and $((b-a)t+a)^3$, respectively. This can be rewritten as

$$\begin{aligned}
\mathbf{Q}(t) &= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} CM\mathbf{P} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} M\left(S_{[a,b]}\mathbf{P}\right)
\end{aligned}$$

where $S_{[a,b]}$ is equal to

$$S_{[a,b]} = M^{-1}CM$$

The new control points for the portion of the curve where t ranges from a to b can now be written as $\left(S_{[a,b]}\mathbf{P}\right)$.

A Specific Example

An example of this which will be useful to us in learning how to piece together two Bézier curves is to find the control polygon for the curve $\mathbf{P}(t)$ when its parameter ranges from 1 to 2. In this case, we have

$$\begin{aligned}
\mathbf{Q}(t) &= \mathbf{P}(t+1) \\
&= \begin{bmatrix} 1 & (t+1) & (t+1)^2 & (t+1)^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} M \left(S_{[1,2]} \mathbf{P} \right)
\end{aligned}$$

where

$$\begin{aligned}
S_{[1,2]} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & 0 & 0 \\ 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \frac{4}{3} & \frac{5}{3} & 2 \\ 1 & \frac{5}{3} & \frac{8}{3} & 4 \\ 1 & 2 & 4 & 8 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 2 \\ 0 & 1 & -4 & 4 \\ -1 & 6 & -12 & 8 \end{bmatrix}
\end{aligned}$$

So the control polygon for that portion of $\mathbf{P}(t)$ curve where t ranges from 1 to 2 is given by

$$\begin{aligned}
\begin{bmatrix} \mathbf{Q}_0 \\ \mathbf{Q}_1 \\ \mathbf{Q}_2 \\ \mathbf{Q}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 2 \\ 0 & 1 & -4 & 4 \\ -1 & 6 & -12 & 8 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{P}_3 \\ -\mathbf{P}_2 + 2\mathbf{P}_3 \\ \mathbf{P}_1 - 4\mathbf{P}_2 + 4\mathbf{P}_3 \\ -\mathbf{P}_0 + 6\mathbf{P}_1 - 12\mathbf{P}_2 + 8\mathbf{P}_3 \end{bmatrix} \qquad (1)
\end{aligned}$$

Working with some algebra, and defining new temporary points $\mathbf{R}_1, \mathbf{R}_2$ and $\mathbf{R}_3$, we see that

$$\mathbf{Q}_0 = \mathbf{P}_3$$

$$\mathbf{Q}_1 = \mathbf{P}_3 + (\mathbf{P}_3 - \mathbf{P}_2) \qquad (2)$$

$$\mathbf{R}_1 = \mathbf{P}_1 + (\mathbf{P}_1 - \mathbf{P}_0) \qquad (3)$$

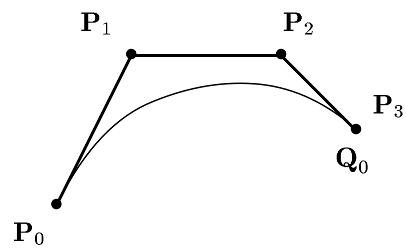
$$\mathbf{R}_2 = \mathbf{P}_2 + (\mathbf{P}_2 - \mathbf{P}_1) \qquad (4)$$

$$\begin{aligned}
\mathbf{R}_3 &= \mathbf{R}_2 + (\mathbf{R}_2 - \mathbf{R}_1) \\
&= \mathbf{P}_2 + (\mathbf{P}_2 - \mathbf{P}_1) + (\mathbf{P}_2 + (\mathbf{P}_2 - \mathbf{P}_1) - \mathbf{P}_1 + (\mathbf{P}_1 - \mathbf{P}_0)) \\
&= \mathbf{P}_0 - 4\mathbf{P}_1 + 4\mathbf{P}_2 \qquad (5)
\end{aligned}$$

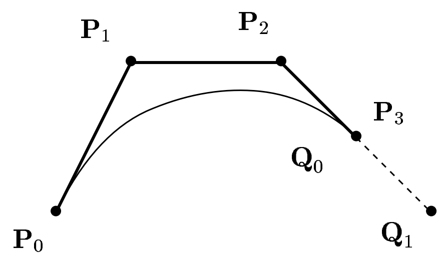
$$\begin{aligned}
\mathbf{Q}_2 &= \mathbf{Q}_1 + (\mathbf{Q}_1 - \mathbf{R}_2) \\
&= \mathbf{P}_3 + (\mathbf{P}_3 - \mathbf{P}_2) + (\mathbf{P}_3 + (\mathbf{P}_3 - \mathbf{P}_2) - \mathbf{P}_2 + (\mathbf{P}_2 - \mathbf{P}_1)) \\
&= \mathbf{P}_1 - 4\mathbf{P}_2 + 4\mathbf{P}_3 \qquad (6)
\end{aligned}$$

$$\begin{aligned}
\mathbf{Q}_3 &= \mathbf{Q}_2 + (\mathbf{Q}_2 - \mathbf{R}_3) \\
&= \mathbf{P}_1 - 4\mathbf{P}_2 + 4\mathbf{P}_3 + (\mathbf{P}_1 - 4\mathbf{P}_2 + 4\mathbf{P}_3 - \mathbf{P}_0 - 4\mathbf{P}_1 + 4\mathbf{P}_2) \\
&= \mathbf{P}_0 + 6\mathbf{P}_1 - 12\mathbf{P}_2 + 8\mathbf{P}_3 \qquad (7)
\end{aligned}$$

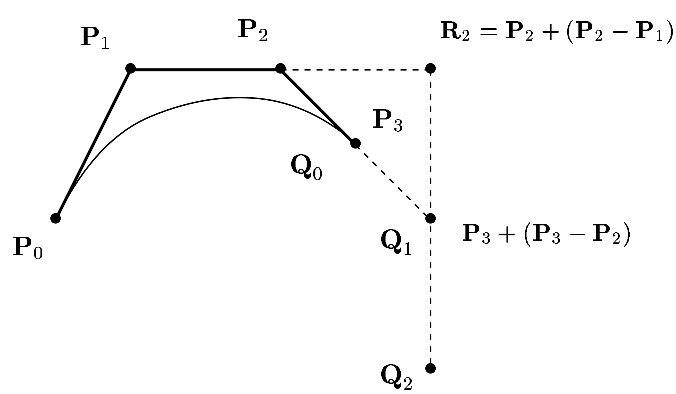
Using these equations, these new control points can be analyzed geometrically and as a result each can be calculated by a simple geometric process using only the initial control polygon $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3\}$. If we consider the following figure, where we have displayed the control point $\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2,$ and $\mathbf{P}_3$, it is easy to locate the point $\mathbf{Q}_0 = \mathbf{P}_3$.



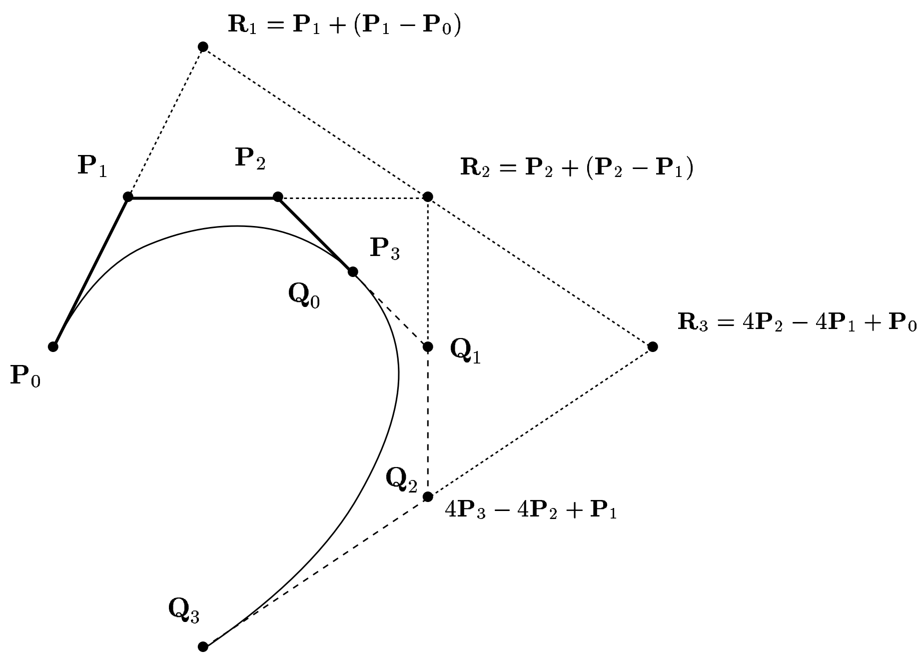
By equation (2), $\mathbf{Q}_1$ lies on an extension of the line $\overline{\mathbf{P}_2\mathbf{P}_3}$ where the distance between $\mathbf{P}_2$ and $\mathbf{P}_3$, and between $\mathbf{Q}_0$ and $\mathbf{Q}_1$ are equal.



By equation (4), $\mathbf{R}_2$ lies on an extension of the line $\overline{\mathbf{P}_1\mathbf{P}_2}$, where the lengths defined by $\overline{\mathbf{P}_1\mathbf{P}_2}$ and $\overline{\mathbf{P}_2\mathbf{R}_2}$ are equal – and as a result of this fact and equation (6), $\mathbf{Q}_2$ lies on an extension of the line $\overline{\mathbf{R}_2\mathbf{Q}_1}$, where the lengths defined by $\overline{\mathbf{R}_2\mathbf{Q}_1}$ and $\overline{\mathbf{Q}_1\mathbf{Q}_2}$ are equal. This enables us to construct $\mathbf{Q}_2$.



Similarly, using equations (3), (4), (5), and (7), we can construct $\mathbf{Q}_3$ as in the the following illustration



The result of this exercise is that we can construct the control points of the curve $\mathbf{Q}(t)$ directly from the original control points for $\mathbf{P}(t)$. These two functions represent the *same curve*.

An interesting exercise for the reader is to calculate the portion of the curve $\mathbf{P}(t)$ as t ranges from 0 to 2. In this case, the new curve $\mathbf{Q}(t)$ can be defined as $\mathbf{Q}(t) = \mathbf{P}(2t)$, and by substituting this into the matrix form, the resulting Bézier polygon should be $\{\mathbf{P}_0, \mathbf{R}_1, \mathbf{R}_3, \mathbf{Q}_3\}$. **Try it out**.

A Expanded Example

The example above illustrated the fact that there are many Bézier polygons that can represent a cubic curve. However the geometric construction process generated by this example did not quite illustrate the fine details of the algorithm. To see the necessary characteristics of the algorithm, we will use the following example: Find the control polygon for the portion of the curve $\mathbf{P}(t)$ when t ranges between 1 and b, for an arbitrary value of b. In this case, we define the curve $\mathbf{Q}(t) = \mathbf{P}(at+1)$, where $a = b-1$ and use our matrix representation to calculate

$$
\begin{aligned}
\mathbf{Q}(t) &= \mathbf{P}(at+1) \\
&= \begin{bmatrix} 1 & (at+1) & (at+1)^2 & (at+1)^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & a & 2a & 3a \\ 0 & 0 & a^2 & 3a^2 \\ 0 & 0 & 0 & a^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} M \left(S_{[1,b]} \mathbf{P} \right)
\end{aligned}
$$

where

$$
\begin{aligned}
S_{[1,b]} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & a & 2a & 3a \\ 0 & 0 & a^2 & 3a^2 \\ 0 & 0 & 0 & a^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & 0 & 0 \\ 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & a & 2a & 3a \\ 0 & 0 & a^2 & 3a^2 \\ 0 & 0 & 0 & a^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -a & (a+1) \\ 0 & a^2 & -2a(a+1) & (a+1)^2 \\ -a^3 & 3a^2(a+1) & -3a(a+1)^2 & (a+1)^3 \end{bmatrix}
\end{aligned}
$$

So the control polygon for that portion of $\mathbf{P}(t)$ curve where t ranges from 1 to b is given by

$$\begin{aligned}
\begin{bmatrix} \mathbf{Q}_0 \\ \mathbf{Q}_1 \\ \mathbf{Q}_2 \\ \mathbf{Q}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -a & (a+1) \\ 0 & a^2 & -2a(a+1) & (a+1)^2 \\ -a^3 & 3a^2(a+1) & -3a(a+1)^2 & (a+1)^3 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{P}_3 \\ -a\mathbf{P}_2 + (a+1)\mathbf{P}_3 \\ a^2\mathbf{P}_1 - 2a(a+1)\mathbf{P}_2 + (a+1)^2\mathbf{P}_3 \\ -a^3\mathbf{P}_0 + 3a^2(a+1)\mathbf{P}_1 - 3a(a+1)^2\mathbf{P}_2 + (a+1)^3\mathbf{P}_3 \end{bmatrix}
\end{aligned}$$

These new control points can again be analyzed geometrically and as a result each can be calculated by a simple geometric process using only the initial control polygon $\{\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3\}$. To accomplish this, we first write

$$\mathbf{Q}_0 = \mathbf{P}_3$$

$$\mathbf{Q}_1 = \mathbf{P}_3 + a(\mathbf{P}_3 - \mathbf{P}_2)$$

$$\mathbf{R}_1 = \mathbf{P}_1 + a(\mathbf{P}_1 - \mathbf{P}_0)$$

$$\mathbf{R}_2 = \mathbf{P}_2 + a(\mathbf{P}_2 - \mathbf{P}_1)$$

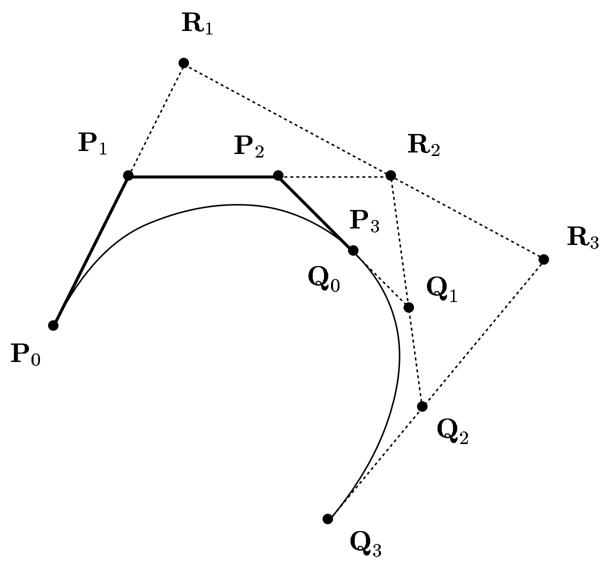
$$\begin{aligned}
\mathbf{R}_3 &= \mathbf{R}_2 + a(\mathbf{R}_2 - \mathbf{R}_1) \\
&= \mathbf{P}_2 + a(\mathbf{P}_2 - \mathbf{P}_1) + a(\mathbf{P}_2 + a(\mathbf{P}_2 - \mathbf{P}_1) - \mathbf{P}_1 + a(\mathbf{P}_1 - \mathbf{P}_0)) \\
&= a^2\mathbf{P}_0 - 2a(a+1)\mathbf{P}_1 + (a+1)^2\mathbf{P}_2
\end{aligned}$$

$$\begin{aligned}
\mathbf{Q}_2 &= \mathbf{Q}_1 + a(\mathbf{Q}_1 - \mathbf{R}_2) \\
&= \mathbf{P}_3 + a(\mathbf{P}_3 - \mathbf{P}_2) + a(\mathbf{P}_3 + a(\mathbf{P}_3 - \mathbf{P}_2) - \mathbf{P}_2 + a(\mathbf{P}_2 - \mathbf{P}_1)) \\
&= a^2\mathbf{P}_1 - 2a(a+1)\mathbf{P}_2 + (a+1)^2\mathbf{P}_3
\end{aligned}$$

$$\begin{aligned}
\mathbf{Q}_3 &= \mathbf{Q}_2 + a(\mathbf{Q}_2 - \mathbf{R}_3) \\
&= -a^3\mathbf{P}_0 + 3a^2(a+1)\mathbf{P}_1 - 3a(a+1)^2\mathbf{P}_2 + (a+1)^3\mathbf{P}_3
\end{aligned}$$

where the last calculation can be done with some algebra.

The important factor here is the a term. Each of these points is on an extension of a line of the original control polygon, or the extension of a constructed line. The factor a determines how much to extend. The following illustration shows the construction for our previous Bézier curve with $a = \frac{3}{4}$, giving the portion of the $\mathbf{P}(t)$ where t ranges from 1 to $\frac{7}{4}$.



Summary

We have shown here that for a cubic curve, there are many control polygons that can define the curve. Using our matrix representation, we have shown how to determine the control polygon that covers an arbitary interval $[c, d]$ of the original curve. Our examples will be very useful when we discuss how to piece two or more Bézier curves together.

On-Line Geometric Modeling Notes

BÉZIER CURVES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

The Bézier curve representation is one that is utilized most frequently in computer graphics and geometric modeling. The curve is defined geometrically, which means that the parameters have geometric meaning – they are just points in three-dimensional space. It was developed by two competing European engineers in the late 1960s to attempt to draw automotive components.

In these notes, we develop the mathematical description for the Bézier curve of arbitrary degree by generalizing the development for the quadratic and cubic Bézier curves, creating a parameterized version of the curve.

Specification of the Curve

Given the set of control points, $\{\mathbf{P}_0, \mathbf{P}_1, ..., \mathbf{P}_n\}$, we can define a Bézier curve of degree n by either of the following definitions:

The Analytic Definition

$$\mathbf{P}(t) \;=\; \sum_{i=0}^{n} \mathbf{P}_i B_{i,n}(t)$$

where

$$B_{i,n}(t) \;=\; \binom{n}{i} t^i (1-t)^{n-i}$$

are the Bernstein polynomials of degree n, and t ranges between zero and one – $0 \leq t \leq 1$.

Geometric Definition

$$\mathbf{P}(t) \;=\; \mathbf{P}_n^{(n)}(t)$$

where

$$\mathbf{P}_i^{(j)}(t) \;=\; \begin{cases} (1-t)\mathbf{P}_{i-1}^{(j-1)}(t) \;+\; t\mathbf{P}_i^{(j-1)}(t) & \text{if } j > 0, \\ \mathbf{P}_i & \text{otherwise} \end{cases}$$

where t ranges between zero and one – $0 \le t \le 1$.

Properties of the Bézier Curve

The Bézier curve has properties similar to that of the quadratic and cubic curve. These can be verified directly from the equations above.

- $\mathbf{P}_0$ and $\mathbf{P}_n$ are on the curve.
- The curve is continuous and has continuous derivatives of all orders.
- The tangent line to the curve at the point $\mathbf{P}_0$ is the line $\overline{\mathbf{P}_0\mathbf{P}_1}$. The tangent to the curve at the point $\mathbf{P}_n$ is the line $\overline{\mathbf{P}_{n-1}\mathbf{P}_n}$.
- The curve lies within the convex hull of its control points. This is because each successive $\mathbf{P}_i^{(j)}$ is a convex combination of the points $\mathbf{P}_i^{(j-1)}$ and $\mathbf{P}_{i-1}^{(j-1)}$.
- $\mathbf{P}_1$, $\mathbf{P}_2$, ..., $\mathbf{P}_{n-1}$ are all on the curve only if the curve is linear.

Summary

Given a sequence of $n+1$ control points, one can specify a Bézier curve of degree n defined by these points. Two definitions of the curve can be given: an analytic definition specifying the blending of the control points with Bernstein polynomials, and a geometric definition specifying a recursive generation procedure that calculates successive points on line segments developed from the control point sequence.

BÉZIER PATCHES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

The representation of a surface that is most frequently used in computer graphics, was independently discovered by Pierre Bézier (pronounced Bez-ye), who was an engineer for Renault and Paul de Casteljau, who was an engineer for Citroën, both working for automobile companies in France. These engineers initially developed a curve representation scheme that is geometrical in construction, and based upon polynomial functions. They extended it to a surface patch methodology that has become the defacto standard for surface generation in computer graphics.

If you are a novice to this field it is suggested that you review the notes on Bézier Curves first, as the equations are easier to understand. Also the fundamental mathematical work on Bernstein Polynomials will be useful.

- Definition and properties of the Bézier patch
- Viewing the Bézier patch as a continuous set of Bézier curves.
- Subdividing the Bézier patch.
- A matrix formulation of the cubic Bézier patch , including subdivision methods by matrix multiplication .

On-Line Geometric Modeling Notes

THE BÉZIER PATCH

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

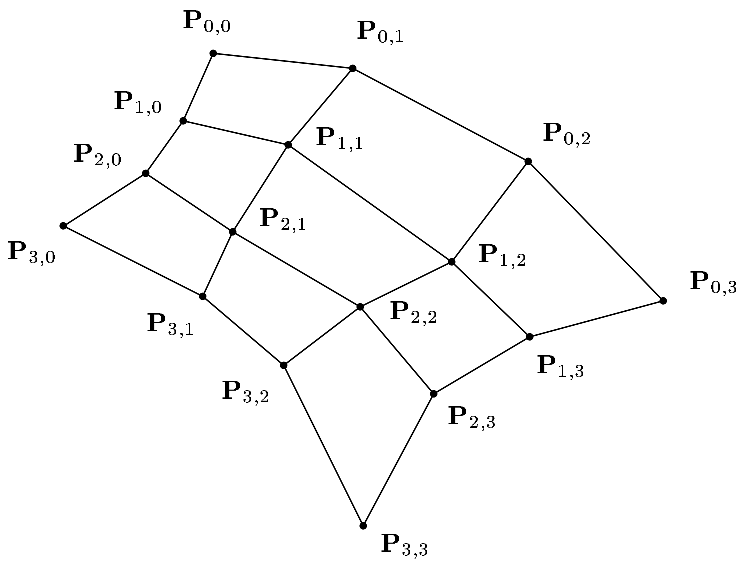
The Bézier patch is the surface extension of the Bézier curve. Whereas a curve is a function of one variable and takes a sequence of control points, the patch is a function of two variables with an array of control points. Most of the methods for the patch are direct extensions of those for the curves.

The Bézier patch is the most commonly used surface representation in computer graphics. An understanding of the patch is fundamental to an understanding of this field.

If you are a novice to this field it is suggested that you review the notes on Bézier Curves first, as the equations are easier to understand. Also the fundamental mathematical work on Bernstein Polynomials will be useful.

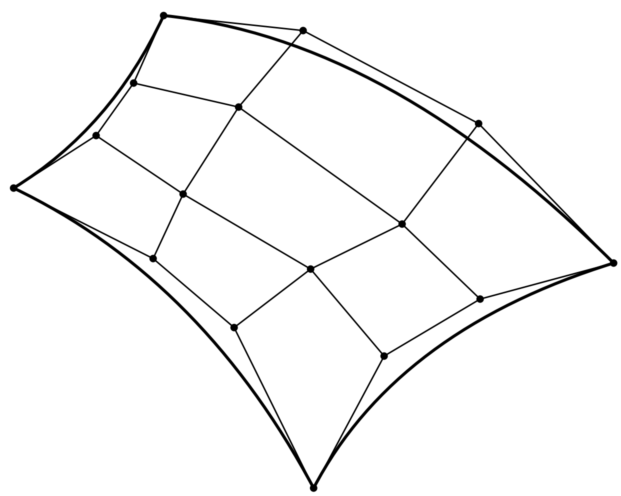
Definition of the Bézier Patch

The extension of Bézier curves to surfaces is called the Bézier patch. The patch is constructed from an $n \times m$ array of control points $\{\mathbf{P}_{i,j} \, : \, 0 \leq i \leq n, \, 0 \leq j \leq m\}$.



and the resulting surface, which is now parameterized by two variables, is given by the equation

$$\mathbf{P}(u,v) \;=\; \sum_{j=0}^{m}\sum_{i=0}^{n}\mathbf{P}_{i,j}B_{i,n}(u)B_{j,m}(v)$$



It is easily seen that this is in the same general form as the Bézier Curve – with the summations running over all the control points, and the *bi-variate* Bernstein Polynomials serving as the functions that blend the control points together. The bi-variate Bernstein polynomials

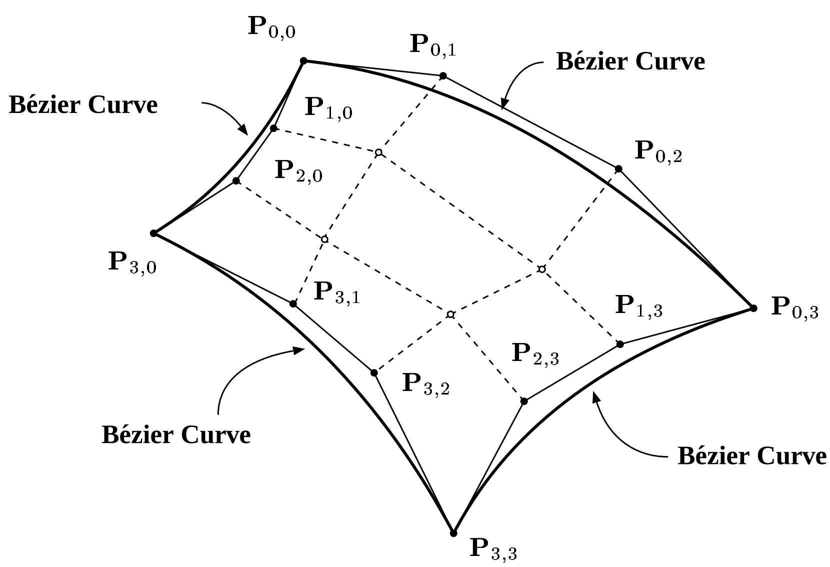
$$B_{i,n}(u)B_{j,m}(v)$$

are just products of two of the uni-variate Bernstein Polynomials

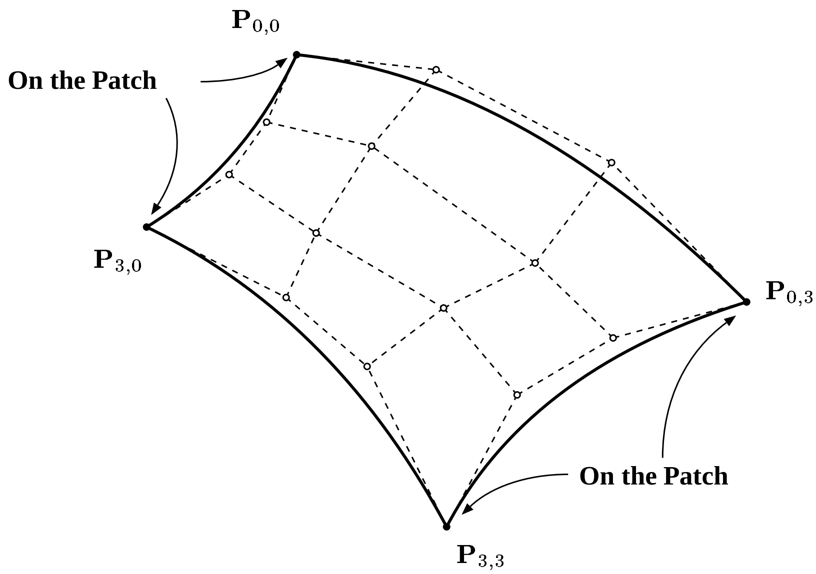
If we set v equal to zero in the equation above, we obtain

$$\begin{aligned}\mathbf{P}(u,0) &= \sum_{j=0}^{m}\sum_{i=0}^{n}\mathbf{P}_{i,j}B_{i,n}(u)B_{j,m}(0) \\ &= \sum_{i=0}^{n}\mathbf{P}_{i,0}B_{i,n}(u)\end{aligned}$$

since $B_{0,m}(0) = 1$ and $B_{j,m}(0) = 0$ for $j = 1, 2, ..., m$. This is just a Bézier curve – and with similar calculations for $\mathbf{P}(u,1)$, $\mathbf{P}(0,v)$ and $\mathbf{P}(1,v)$, we can show that all the edge curves of the patch are Bézier curves.



This then implies that the corner points are actually on the patch.



Properties of the Bézier Patch

The Bézier patch has properties similar to that of the Bézier curve.. These can be verified directly from the defining equations.

- The four points $\mathbf{P}_{0,0}$, $\mathbf{P}_{0,m}$, $\mathbf{P}_{n,0}$ and $\mathbf{P}_{n,m}$ are on the patch. The other control points are all on the patch only if the patch is planar.
- The patch is continuous and partial derivatives of all orders exist and are continuous.
- The patch lies within the convex hull of its control points.

Summary

The Bézier patch is a direct extension of Bézier curves to surfaces. The definition of the patch follows directly the definition of the curve, with the primary differences being the use of an array of control points and the bivariate Bernstein Polynomials.

As it turns out, the Bézier patch can be viewed as a continuous set of Bézier curves. This greatly simplifies computation on the patch because in many instances that calculations on the patch can be reduced to calculations on Bézier curves.

A MATRIX REPRESENTATION OF A CUBIC BÉZIER PATCH

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

A cubic Bézier patch has a useful representation when written in a matrix form. This form allows us to specify many operations with Bézier patches as matrix operations which can be performed quickly on computer systems optimized for geometry operations with matrices.

This is an unusual representation for many students as it is not frequently shown in basic courses. If you have not seen this before it is suggested that you begin with the section on matrix representations for Bézier curves in which the equations are simpler and easier to understand.

Developing the Matrix Formulation

A cubic Bézier curve can be written in a convenient matrix form. A bicubic Bézier patch can be written in a matrix form by using methods similar to that for a Bézier Curve. Utilizing the representation of a Bézier patch as a continuous set of Bézier curves we have

$$
\begin{aligned}
\mathbf{P}(u,v) &= \sum_{j=0}^{3}\sum_{i=0}^{3}\mathbf{P}_{i,j}B_{i,3}(u)B_{j,3}(v) \\
&= \sum_{j=0}^{3}\left[\sum_{i=0}^{3}\mathbf{P}_{i,j}B_{i,3}(u)\right]B_{j,3}(v) \\
&= \sum_{j=0}^{3}\begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{P}_{0,j} \\ \mathbf{P}_{1,j} \\ \mathbf{P}_{2,j} \\ \mathbf{P}_{3,j} \end{bmatrix} B_{j,3}(v) \\
&= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}\begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix}\begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}
\end{aligned}
$$

and so the cubic Bézier patch is frequently written

$$
\mathbf{P}(t) \;=\; \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} M \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}
$$

where

$$
M \;=\; \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}
$$

Patch Subdivision Using the Matrix Form

Suppose we wish to subdivide the patch at the point $u = \frac{1}{2}$. We reparameterize the matrix equation above (by substituting $\frac{u}{2}$ for u) to cover only the first half of the patch, and simplify to obtain.

$$\mathbf{P}(\frac{u}{2},v) = \begin{bmatrix} 1 & \left(\frac{u}{2}\right) & \left(\frac{u}{2}\right)^2 & \left(\frac{u}{2}\right)^3 \end{bmatrix} M \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} M \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} M S_L \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

where the matrix S_L is defined as

$$
\begin{aligned}
S_L &= M^{-1}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} M \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & 0 & 0 \\ 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{6} & 0 & 0 \\ 1 & \frac{1}{3} & \frac{1}{12} & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{bmatrix}
\end{aligned}
$$

and is identical to the left subdivision matrix for the curve case. So in particular, the subpatch $\mathbf{P}(\frac{u}{2})$ is again a Bézier patch and the quantity

$$
\begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{bmatrix} \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix}
$$

forms the control points of this patch.

Calculation of the Second Half of the Patch

In the same way, we can obtain the subdivision matrix for the second half of the patch. First we reparameterize the original curve, and then simplify to obtain

$$\mathbf{P}(\frac{1}{2}+\frac{u}{2},v) = \begin{bmatrix} 1 & \left(\frac{1}{2}+\frac{u}{2}\right) & \left(\frac{1}{2}+\frac{u}{2}\right)^2 & \left(\frac{1}{2}+\frac{u}{2}\right)^3 \end{bmatrix} M \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} \\ 0 & \frac{1}{2} & \frac{1}{2} & \frac{3}{8} \\ 0 & 0 & \frac{1}{4} & \frac{3}{8} \\ 0 & 0 & 0 & \frac{1}{8} \end{bmatrix} M \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} M S_R \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix} M^T \begin{bmatrix} 1 \\ v \\ v^2 \\ v^3 \end{bmatrix}$$

where

$$S_R = \begin{bmatrix} \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

which is identical to the right subdivision matrix in the curve case and we obtain a matrix that can be applied to a set of control points to produce the control points for the second half of the patch.

General Subdivision with either Parameter

We can develop a procedure to generate the control points for the first and second portions of the patch when subdivision is done with respect to v. These are

$$PS_L \; and \; PS_R$$

where

$$P = \begin{bmatrix} \mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3} \\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3} \\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3} \\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3} \end{bmatrix}$$

The development is exactly parallel to that with respect to u.

Combining these two methods, we can see that the arrays

$$S_L P S_L$$
$$S_R P S_L$$
$$S_L P S_R$$
$$S_R P S_R$$

segment the patch into quarters, the first array being the quarter that corresponds to $0 \leq u \leq \frac{1}{2}, 0 \leq v \leq \frac{1}{2}$, the second to the one that corresponds to $0 \leq u \leq \frac{1}{2}, \frac{1}{2} \leq v \leq 1$, etc.

Summary

We have developed a matrix form for the Bézier patch which parallels the development for the Bézier curve. This representation allows us to develop matrices that, which multiplied by the control point array, calculate the control points of a subdivided portion of the patch.

These matrix equations exist for patches of all orders – we have done order 4 (or degree 3) patches here. However, the matrices are $n \times n$ for a patch of order n, and are not as easily written down.

BÉZIER CURVES ON BÉZIER PATCHES

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

The Bézier patch is the surface extension of the Bézier curve. The definition of the patch follows directly the definition of the curve, with the primary differences being the use of an array of control points and the bivariate Bernstein Polynomials. The edge curves of the patch are Bézier curves and the "corner" control points are always on the curve.

In these notes we show that a patch can be treated as a continuous set of Bézier curves. That is, for any fixed parameter u_0 or v_0 we can define a Bézier curve that lies directly on the surface of the patch. This is a very valuable tool for calculations on the patch.

Calculating Bézier Curves on Bézier Patches

In the development of the Bézier patch, we have shown that the boundary curves of the patch are Bézier curves – that is, $\mathbf{P}(0, v)$ and $\mathbf{P}(1, v)$ are Bézier curves lying on the boundary of the patch.

If we examine the definition of a Bézier patch closely, and group factors appropriately,

$$\mathbf{P}(u,v) \;=\; \sum_{j=0}^{m} \left[\sum_{i=0}^{n} \mathbf{P}_{i,j} B_{i,n}(u) \right] B_{j,m}(v)$$

we notice that portion of the equation inside the brackets is the representation of a Bézier curve. If we fix $u = u_0$, the internal sum can be calculated (for $j = 0, ..., m$). This implies that $\mathbf{P}(u_0, v)$ is a Bézier curve *on the surface*.

If we define $\mathbf{Q}_j(u)$ to be the value

$$\mathbf{Q}_j(u) \;=\; \sum_{i=0}^{n} \mathbf{P}_{i,j} B_{i,n}(u)$$

(i.e. the value inside the brackets above) then we can see that

$$\mathbf{P}(u,v) \;=\; \sum_{j=0}^{m} \mathbf{Q}_j(u) B_{j,m}(v)$$

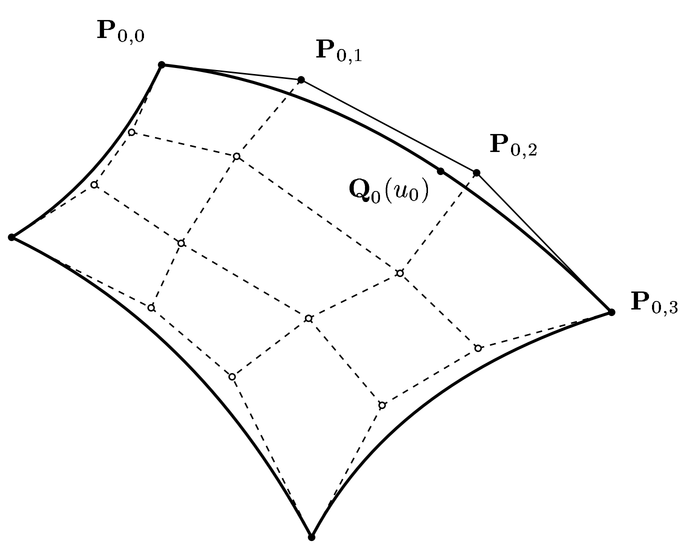
That is, the quantities $\mathbf{Q}_j(u)$ form the control points of another Bézier curve, and together for all u and v, they form the surface.

Therefore, given $u = u_0$, we can calculate the quantities $\mathbf{Q}_0(u_0)$, $\mathbf{Q}_1(u_0)$, ..., $\mathbf{Q}_m(u_0)$, giving m control points to utilize for the curve

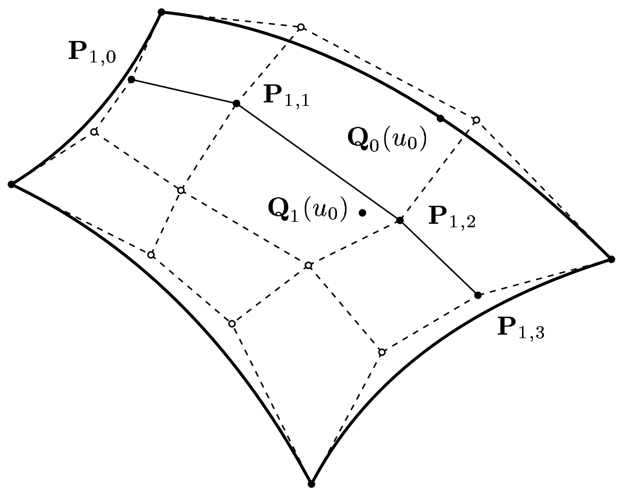
$$\mathbf{Q}(v) \;=\; \sum_{j=0}^{m} \mathbf{Q}_j(u_0) B_{j,m}(v)$$

This curve lies on the patch – since it is really $\mathbf{P}(u_0, v)$, and calculating $\mathbf{Q}(v_0)$ gives us the point on the patch at (u_0, v_0). Since $\mathbf{Q}(v)$ is a Bézier curve, this calculating is straightforward. The following illustration shows the relationship between the $\mathbf{Q}$s and the $\mathbf{P}$s in the 4×4 case.

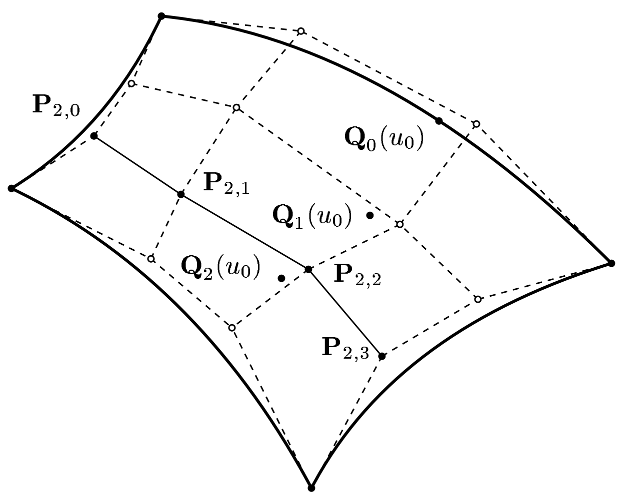
First the point $\mathbf{Q}_0(u_0)$ is calculated as a point on the Bézier curve defined by the control points $\mathbf{P}_{0,0}$, $\mathbf{P}_{0,1}$, $\mathbf{P}_{0,2}$ and $\mathbf{P}_{0,3}$.



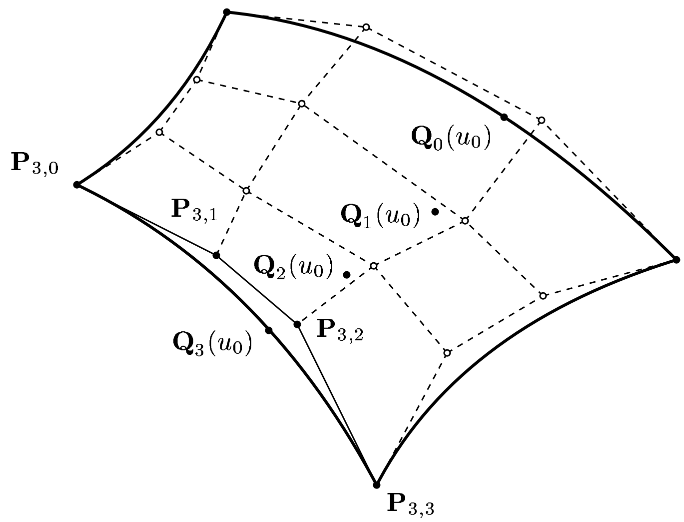
next the point $\mathbf{Q}_1(u_0)$ is calculated as a point on the Bézier curve defined by the control points $\mathbf{P}_{1,0}$, $\mathbf{P}_{1,1}$, $\mathbf{P}_{1,2}$ and $\mathbf{P}_{1,3}$.



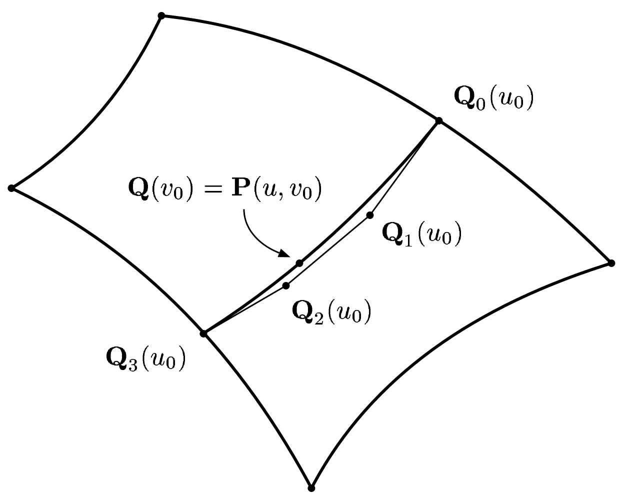
then the point $\mathbf{Q}_2(u_0)$ is calculated as a point on the Bézier curve defined by the control points $\mathbf{P}_{2,0}$, $\mathbf{P}_{2,1}$, $\mathbf{P}_{2,2}$ and $\mathbf{P}_{2,3}$.



and finally, the point $\mathbf{Q}_2(u_0)$ is calculated as a point on the Bézier curve defined by the control points $\mathbf{P}_{2,0}$, $\mathbf{P}_{2,1}$, $\mathbf{P}_{2,2}$ and $\mathbf{P}_{2,3}$.



The point $\mathbf{P}(u_0, v_0)$, on the patch, is calculated as a point on the Bézier curve defined by the control points $\mathbf{Q}_0(u_0)$, $\mathbf{Q}_1(u_0)$, $\mathbf{Q}_2(u_0)$ and $\mathbf{Q}_3(u_0)$,



Calculating with the Other Parameter

If we reverse the order of the sums in the defining equation and regroup, we find that

$$\mathbf{P}(u, v) = \sum_{i=0}^{m} \left[\sum_{j=0}^{n} \mathbf{P}_{i,j} B_{j,n}(v) \right] B_{i,m}(u)$$

which implies, if we do the above construction again, that we can first fix $v = v_0$, define control points $\mathbf{Q}_0(v_0)$, $\mathbf{Q}_1(v_0)$, ..., $\mathbf{Q}_n(v_0)$ and define the equation as

$$\mathbf{P}(u, v_0) \;=\; \sum_{i=0}^{m} \mathbf{Q}_i(v_0) B_{i,m}(u)$$

which is again a Bézier curve lying on the surface.

Thus, we can either do this procedure by fixing u first, or fixing v first, and we obtain the same result.

Summary

The Bézier patch is a direct extension of Bézier curves to surfaces. The definition of the patch follows directly the definition of the curve, with the primary differences being the use of an array of control points and the bivariate Bernstein Polynomials. However, the patch can be treated as a continuous set of Bézier curves, and the calculations to find a point on the patch can be reduced to finding several points on curves. The calculations are parameter independent in that it does not matter whether we start with the u or v parameter.

BÉZIER PATCH SUBDIVISION

Kenneth I. Joy
Visualization and Graphics Research Group
Department of Computer Science
University of California, Davis

Overview

A general method can be specified to subdivide a Bézier patch. This method is specified unlike the matrix methods, as it is based upon the definition of the patch as a set of curves..

The Method for Subdivision

We recall that, if we take the analytic equation of a Bézier patch, fix u and group factors appropriately, we obtain

$$\mathbf{P}(u,v) \;=\; \sum_{j=0}^{m} \left[\sum_{i=0}^{n} \mathbf{P}_{i,j} B_{i,n}(u) \right] B_{j,m}(v)$$

We notice that portion of the equation inside the brackets is the representation of a Bézier curve. If we let $\mathbf{Q}_j(u)$ be the value inside the brackets, i.e.

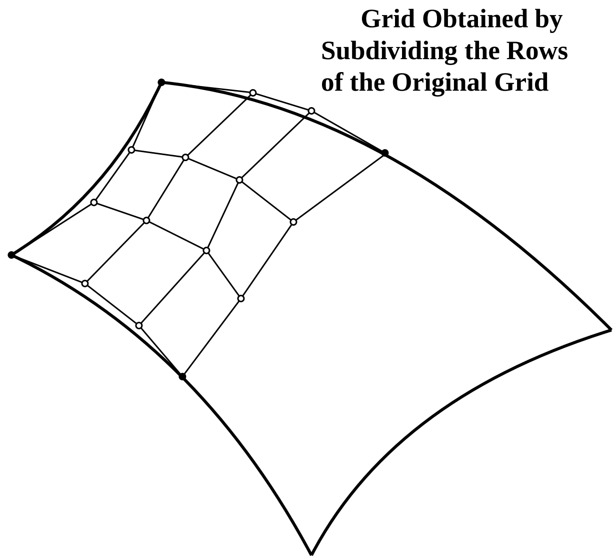
$$\mathbf{Q}_j(u) \;=\; \sum_{i=0}^{n} \mathbf{P}_{i,j} B_{i,n}(u)$$

Then

$$\mathbf{P}(u,v) \;=\; \sum_{j=0}^{m} \mathbf{Q}_j(u) B_{j,m}(v)$$

That is, the quantities $\mathbf{Q}_j(u)$ form the control points of another Bézier curve, and together for all u and v, they form the surface.

If, then, we subdivide each of the m rows of the $\mathbf{P}_{i,j}$ matrix, it implies that the $\mathbf{Q}_j$s in the above equation represent only points from the first half of the patch (with respect to u). The following illustration shows the result of subdividing the rows in the 4×4 case.



The second half of the patch can be obtained in a similar fashion. The first and second half of the patch, with respect to v, can be obtained by subdividing the columns.

Summary

So, using only curve methods, and by subdividing the rows or columns of the control point array, we can effectively subdivide a Bézier patch. This is the most frequently used algorithm in software implementations of subdivision and can be utilized for Bézier patches of arbitrary degree.
