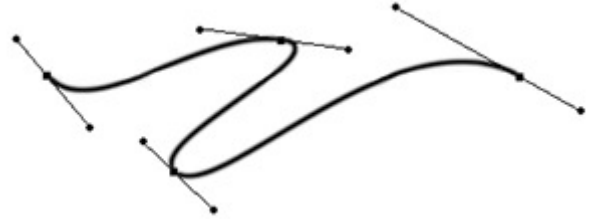


QS Bézier Curves and Gradient Palettes

Michael Sargent – November 2019



Bézier curves are determined by sets of control points and are derived from the parametric equations:

($x = B_x(t)$, $y = B_y(t)$) for increments of t from 0 to 1.

$B_x()$ and $B_y()$ are weighted binomial expansions of $((1-t) + t)^n$, in which the weighting factors **W** are the coordinates of the control points. The number of control points is $(n + 1)$. The coefficients of the binomial expansions for each value of **n** form Pascal's triangle.

Thus, the equations for the first few degrees of Bézier curves are:

linear (straight line segments): $n = 1$

$$x = W_{1x} * 1 * (1-t) + W_{2x} * 1 * t$$

$$y = W_{1y} * 1 * (1-t) + W_{2y} * 1 * t$$

quadratic: $n = 2$

$$x = W_{1x} * 1 * (1-t)^2 + W_{2x} * 2 * (1-t) * t + W_{3x} * 1 * t^2$$

$$y = W_{1y} * 1 * (1-t)^2 + W_{2y} * 2 * (1-t) * t + W_{3y} * 1 * t^2$$

cubic: $n = 3$

$$x = W_{1x} * 1 * (1-t)^3 + W_{2x} * 3 * (1-t)^2 * t + W_{3x} * 3 * (1-t) * t^2 + W_{4x} * 1 * t^3$$

$$y = W_{1y} * 1 * (1-t)^3 + W_{2y} * 3 * (1-t)^2 * t + W_{3y} * 3 * (1-t) * t^2 + W_{4y} * 1 * t^3$$

$$\begin{array}{ccccccc} & & & & 1 & & & \\ & & & 1 & & 1 & & \\ & & 1 & & 2 & & 1 & \\ & 1 & & 3 & & 3 & & 1 \\ & & 1 & & 4 & & 6 & & 4 & & 1 \\ & 1 & & 5 & & 10 & & 10 & & 5 & & 1 \\ & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{array}$$

Pascal's triangle

For the sake of completeness, the generalized formula is:

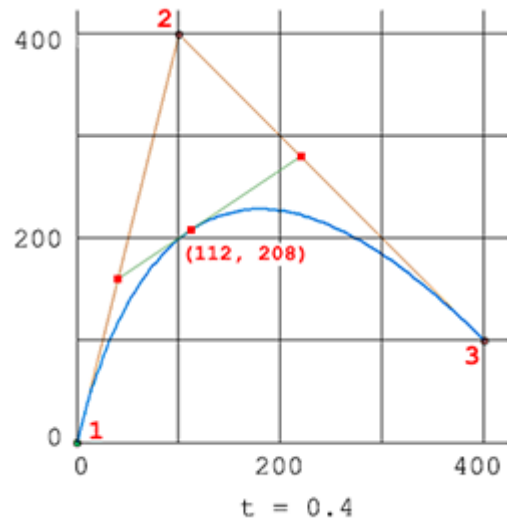
$$\begin{aligned} (x+y)^n &= \sum_{k=0}^n \binom{n}{k} x^{n-k} y^k = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k}, \quad \binom{n}{k} = \frac{n!}{k!(n-k)!} \\ &= \binom{n}{0} x^n y^0 + \binom{n}{1} x^{n-1} y^1 + \binom{n}{2} x^{n-2} y^2 + \cdots + \binom{n}{n-1} x^1 y^{n-1} + \binom{n}{n} x^0 y^n \end{aligned}$$

The binomial coefficient (“n choose k”) is the number of combinations of size **k** in a set of **n** items, and it can be read as item **k** in row **n** of Pascal's triangle (count starts at 0).

Here is a quadratic Bézier curve with 3 control points:

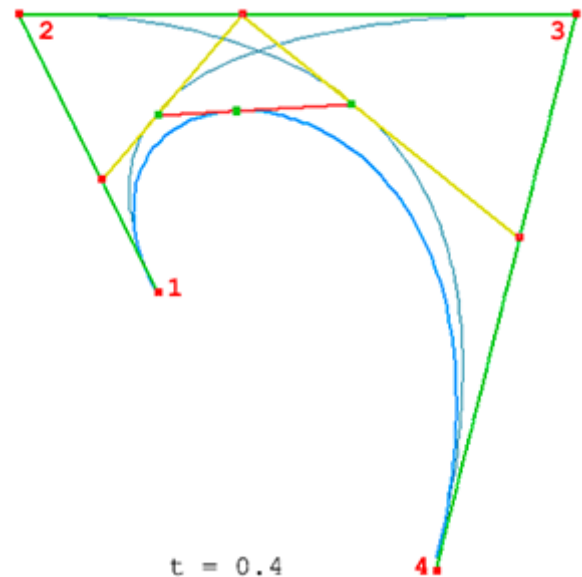
1: (0, 0) ; 2: (100, 400); 3: (400, 100)

Inserting these values as weighting factors into the Bézier equations with $t = 0.4$ yields the point (112, 208). The graph also demonstrates a geometric correlation: the curve can be formed by interpolation. Points are inserted at 0.4 of the lengths of the line segments connecting control points 1 and 2, and control points 2 and 3. Then a segment is drawn to connect those 2 points. The point at 0.4 of the length of that segment, (112, 208), lies on the curve, and the segment itself, being tangent to the curve, indicates the curve's derivative at that point. This geometric construction is the basis of deCasteljau's algorithm, which is demonstrated in the following illustration.



This cubic Bézier curve can be constructed in 3 steps.

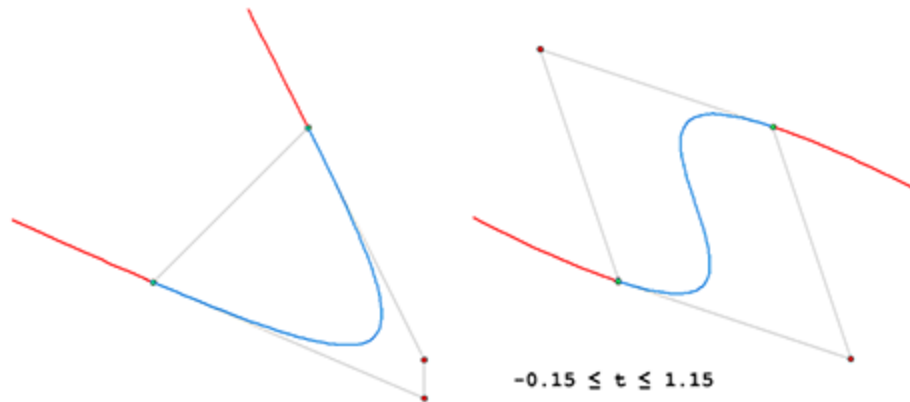
- 1) Draw 3 linear Bézier curves (*i.e.* line segments) connecting the pairs of control points (1-2, 2-3, 3-4), and insert points at 0.4 of the length of each segment.
 - 2) Connect the interpolated points from step 1 with line segments and insert points at 0.4 of the length of each of those segments. Those points lie on the 2 quadratic curves derived from the control point triplets (1-2-3, 2-3-4). Those curves approximate the final curve more closely than the previous linear curves.
 - 3) Connect the interpolated points from step 2 with a line segment and insert a point at 0.4 of the length of that segment. That point lies on the final cubic curve.
- Repeat for each value of t from 0 to 1.



For higher degree curves, an analogous series of steps (the number of which equals the degree of the curve) is followed until only one line segment is drawn. The interpolated point along that final segment lies on the final curve.

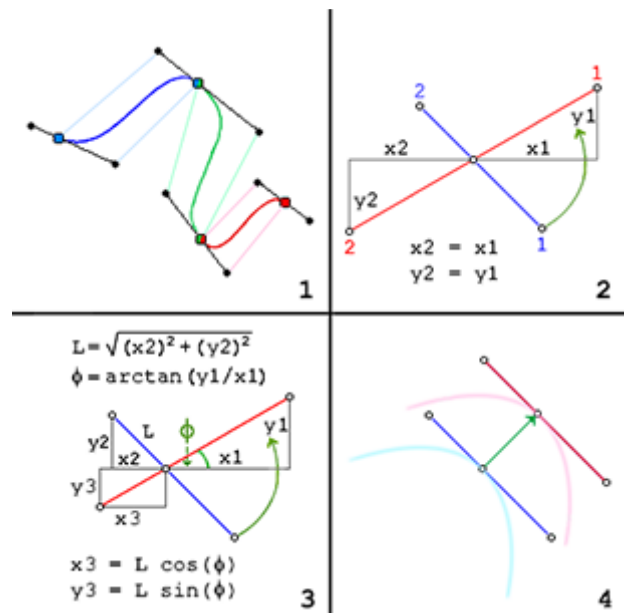
The algorithm can be programmed as a recursive (self-calling) function. The curve can be split by treating points that occur before and after a selected t value separately.

The binomial is set to $((1 - t) + t)^n$ so that the curve always lies between the first control point ($t = 0$, so the curve point = $(1 - 0)^n$ * starting point) and the last control point ($t = 1$, so the curve point = 1^n * ending point). The value of t is limited to the range $0 - 1$ so that the curve stays within the bounding polygon (called the “hull”) formed by the control points. But there is no theoretical restriction to extending a curve beyond the control points by using t values outside of this range.



In the above examples, the portions of the curves within the range $0 \leq t \leq 1$ are shown in blue, and the portions outside of that range are shown in red.

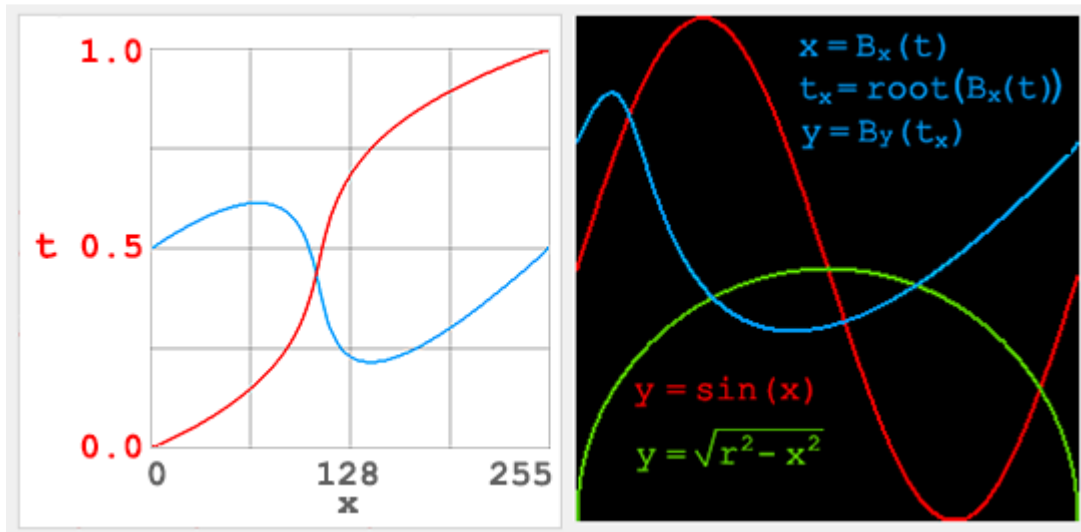
The “paths” used in image editing and vector drawing software are poly-Bézier curves: series of curves in which the end point of one curve is the start point of the next. Image 1 at the right shows a curve formed by 3 cubic Bézier curves. Typically the junctions, or on-curve points, are connected to their adjacent control points by line segments to form “handles” used to adjust the curve.



To avoid sharp kinks at the junctions, the derivatives (indicated by the slopes the handle segments) of each curve should be equal at the junctions. Image 2 shows point #1 on the blue segment being moved to point #1 on the new red segment, increasing the length of segment #1. The new point #2 can simply be mirrored across the on-curve point. But now the length of segment #2 also has been altered.

To preserve the original length of segment #2, the Pythagorean theorem and some basic trigonometric functions can be used, as shown in image 3. Finally, if the on-curve point is moved, its associated points must be translated correspondingly, as shown in image 4.

Creating palettes from **Bézier curves** presents a challenge not faced when using **sine curves** or **circular arcs**. In the latter situations, the X-axis can be divided by the number of palette entries and the y value can be calculated directly and transformed into the corresponding palette value. But since **Bézier curves** are parametric, the y values cannot be calculated directly from the x values. It is even possible for multiple y values to be associated with a single x value, as can be seen in the upper illustration on the page 3. (That can be avoided by keeping the x values of the intermediate control points within the range of the start and end points.)



In addition, incrementing the value of x by a fixed amount results in variations in the intervals between corresponding t values, and therefore in curve segments of different lengths, disrupting the smoothness of the palette gradient. The left image shows a blue Bézier curve. The red curve shows the inconstant variability of t with respect to x. This is summarized in the table.

x range	0 - 64	64 - 128	128 - 192	192 - 255
t range	0.0 – 0.149	0.149 – 0.678	0.678 – 0.892	0.892 – 1.0
dt	0.149	0.529	0.214	0.108

There is a solution to finding values of y corresponding to selected values of x: Divide the X-axis as usual, and then, as implied by the right image, let $x = B_x(t)$, solve for t_x as the dependent variable, and use that value of t_x to calculate y. The equation is converted from binomial expansion to standard polynomial form

$$a_n t^n + a_{(n-1)} t^{(n-1)} + a_{(n-2)} t^{(n-2)} + \dots + a_2 t^2 + a_1 t + a_0 - x = 0$$

and solved for a root between 0 and 1. For quadratic equations, the well-known quadratic formula can be used.

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Unfortunately, the analogous formula for cubic equations is much more daunting, and can involve complex numbers, even if none of the roots are complex.

$$x = \sqrt[3]{\left(\frac{-b^3}{27a^3} + \frac{bc}{6a^2} - \frac{d}{2a}\right) + \sqrt{\left(\frac{-b^3}{27a^3} + \frac{bc}{6a^2} - \frac{d}{2a}\right)^2 + \left(\frac{c}{3a} - \frac{b^2}{9a^2}\right)^3}} + \sqrt[3]{\left(\frac{-b^3}{27a^3} + \frac{bc}{6a^2} - \frac{d}{2a}\right) - \sqrt{\left(\frac{-b^3}{27a^3} + \frac{bc}{6a^2} - \frac{d}{2a}\right)^2 + \left(\frac{c}{3a} - \frac{b^2}{9a^2}\right)^3}} - \frac{b}{3a}$$

By using some clever manipulations originally published by Cardano in 1545, a reasonably straightforward method of obtaining the roots of a cubic equation can be coded. This allows the *QS Gradient* program to update cubic Bézier curves in real time, so that palette values can be calculated and displayed over an evenly-spaced range of palette indices on the X-axis. Of course, Newton's method could be used to find acceptable approximations as an alternative.

Historical Notes

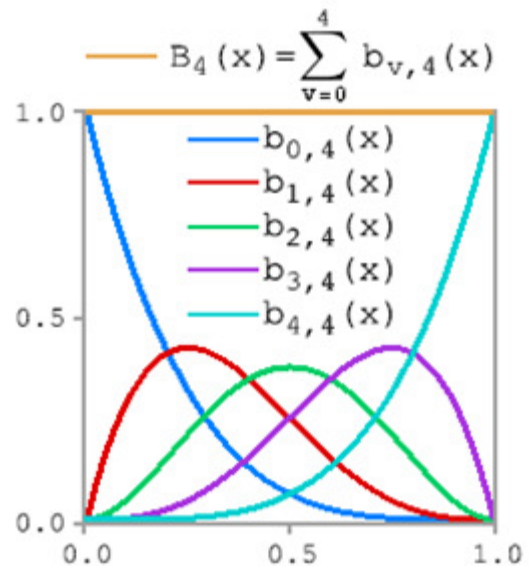
The Bézier interpolation functions are based on Bernstein polynomials, which were first developed in 1911 by the Russian mathematician Sergei Bernstein (1880 – 1968). His doctoral thesis, *About the Best Approximation of Continuous Functions by Polynomials of Given Degree* led to a proof of the Weierstrass theorem that continuous functions can be approximated as closely as desired by polynomials. Bernstein “basis” polynomials take the form:

$$b_{\nu,n}(x) = \binom{n}{\nu} x^{\nu} (1-x)^{n-\nu}, \nu = 0, \dots, n$$

A set of basis polynomials can be combined into a Bernstein polynomial

$$B_n(x) = \sum_{\nu=0}^n \beta_{\nu} b_{\nu,n}(x)$$

where β_{ν} represents the Bernstein coefficients (now sometimes called Bézier coefficients) used as weighting factors. This image shows a set of quartic basis polynomials ($n = 4, 0 \leq \nu \leq 4$), the combination of which yields a Bernstein polynomial with a constant value of 1.



Bézier curves were developed in the mid-20th century to facilitate CAD in the automotive industry. Paul de Casteljau (b. 1930) was the first person to utilize Bernstein polynomials for this purpose in 1958, while working for Citroën. He developed the geometry-based algorithm to compute the curves which bears his name. As Pierre Bézier himself was quoted:

There is no doubt that Citroën was the first company in France that paid attention to CAD, as early as 1958. Paul de Casteljau, a highly gifted mathematician, devised a system based on the use of Bernstein polynomials. ... the system devised by de Casteljau was oriented towards translating already existing shapes into patches, defined in terms of numerical data. ... Due to Citroën's policy, the results obtained by de Casteljau were not published until 1974, and this excellent mathematician was deprived of part of the well deserved fame that his discoveries and inventions should have earned him.

Pierre Bézier (1910-1999) studied parametric polynomial curves in graduate school and was hired by Renault in 1933. The company initially resisted his recommendation to pursue numerical control and computer-assisted design. But he persisted and developed a system he called "UNISURF". Bézier patented his system, and published his work in 1962. When Renault and Peugeot initiated a cooperative research program in 1966, some Peugeot engineers supported and helped advance the system, which ultimately won widespread acceptance. This led to the association of his name with the curves that his system used.

Bézier devised the familiar control system in which a common control point joining 2 cubic Bézier curves is connected to the 2 adjacent and opposite control points by line segments forming "handles", which are manipulated interactively in computer software to adjust the shape of the curves. Ultimately, Bézier curves have been adopted by increasing numbers of software companies like Adobe and Corel for use in vector design programs. Among numerous applications, they are used for outline fonts such as PostScript and TrueType.

The most intriguing figure connected to the *QS Gradient* program is Gerolamo Cardano (1501-1576). He was a prominent physician and mathematician whose monumental treatise of 1545, *Ars Magna*, contained solutions to cubic and quartic equations, which he attributed to Scipione del Ferro and Lodovico Ferrari. He also studied hypocycloids, designing the "Cardan gear" mechanism, in which the inner circle's radius is half that of the outer circle, causing a point on its circumference to move in a linear oscillating fashion. He introduced the use of negative numbers and binomial coefficients, and recognized imaginary numbers. As an accomplished gambler, he developed the first systematic treatment of probability. He developed versions of the combination lock, the gimbal and the universal joint, now used in automobile transmissions. He also proposed that deaf people could be intelligent and learn to read and write.



Acknowledgement

Much of the information in this article is presented with clarity and depth in an excellent online book entitled *A Primer on Bézier Curves*, written and updated by an eclectic from Vancouver Island who calls himself “Pomax”. The primer offers detailed descriptions of a wealth of procedures involving Bézier curves, with pseudocode, including Cardano’s cubic root method. It also contains many interactive Java script applets which aid in understanding the concepts.

<https://pomax.github.io/bezierinfo/>

© Michael Sargent
November 2019
msargent@uvm.edu



My Honda Fit: a tribute to deCasteljau, Bézier and all other automotive designers