Approximate a Curve by Circular Arcs

David Eberly, Geometric Tools, Redmond WA 98052
https://www.geometrictools.com/

Created: May 6, 2023

Contents

1 Introduction

Given a continuous parametric curve $\boldsymbol{X}(t)$ for $t \in [t_{\min}, t_{\max}]$, approximate the curve by a collection of circular arcs. The approximation is a continuous function; that is, the arcs share endpoints. The derivative is not guaranteed to be continuous at the shared endpoints.

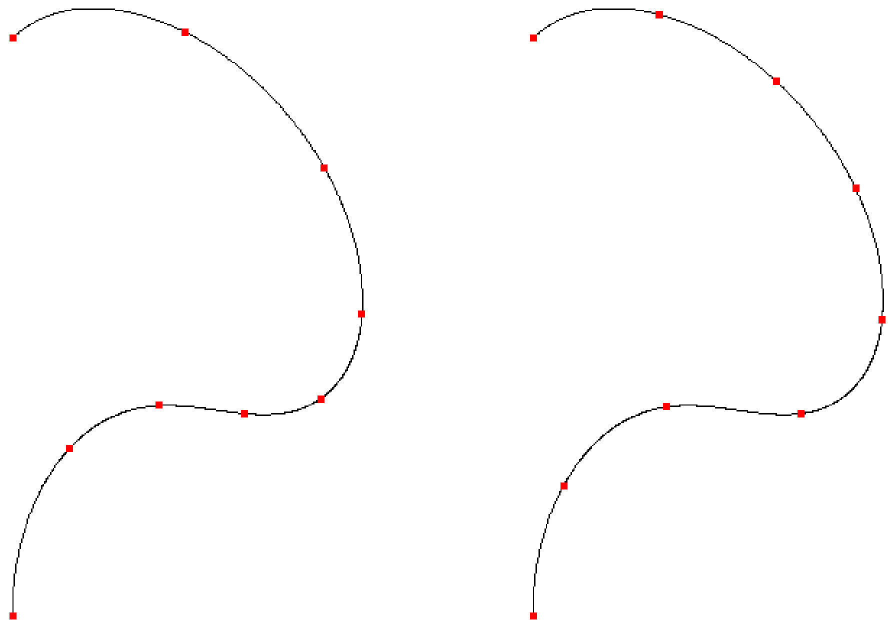
2 Algorithm

2.1 Subdivision of the Curve

Choose the n arcs for the approximation. The parametric curve is evaluated at $n+1$ parameters t_i for $0 \leq i \leq n$. The sequence must be increasing, $t_{\min} = t_0 < t_1 < \ldots < t_{n-1} < t_n = t_{\max}$. Define $\boldsymbol{P}_i = \boldsymbol{X}(t_i)$; these points are the endpoints of the arcs. Two consecutive arcs share an endpoint. For an open parametric curve, $\boldsymbol{P}_0 \neq \boldsymbol{P}_n$. The point $\boldsymbol{P}_0$ is an endpoint of the first arc and is not shared by another arc. The point $\boldsymbol{P}_n$ is an endpoint of the last arc and is not shared by another arc. For a closed parametric curve, $\boldsymbol{P}_0 = \boldsymbol{P}_n$. The common point is a shared endpoint of the first and last arc, but in an implementation two copies of the shared point are managed, one an endpoint of the first arc and one an endpoint of the last arc.

Any such sequence of parameters is allowed, but to obtain a reasonable distribution of the points, choose the parameters so that the arclengths of the subcurves on $[t_i, t_{i+1}]$ are a constant. If L is the total arclength of the curve, the arclengths of the subcurves are all L/n. Figure 1 shows a curve with a subdivision uniform in the curve parameter t and a subdivision uniform in the arclength parameter s.

Figure 1. The left image shows the subdivision uniform in the curve parameter t. The right image shows the subdivision uniform in the arclength parameter s.



The images show a Bézier curve of degree 7 with control points $(0,0)$, $(0,2)$, $(2,2)$, $(1,1)$, $(3,0)$, $(4,3)$, $(1,5)$, and $(0,4)$. The subdivisions each have 9 points. The total length of the curve is $L \doteq 7.4794$ (rounded to 4 fractional digits).

Table 1 shows the subdivision uniform in t.

Table 1. The subdivision in curve parameter t where $t_i = i/8$ for $0 \le i \le 8$. The s-values are the corresponding arclengths for the subcurves with $t \in [0, t_i]$ and are rounded to 4 fractional digits.

i	0	1	2	3	4	5	6	7	8
t-value	0.0000	0.1250	0.2500	0.3750	0.5000	0.6250	0.7500	0.8750	1.0000
s-value	0.0000	1.2571	1.9636	2.5579	3.1050	3.7765	4.8420	6.2059	7.4794

Table 2 shows the subdivision uniform in s.

Table 2. The subdivision in arclength parameter s where $s_i = L/8$ for total length $L \doteq 7.4794$ and $0 \le i \le 8$. The t-values are the corresponding curve parameters for the subcurves with $s \in [0, s_i]$ and are rounded to 4 fractional digits.

i	0	1	2	3	4	5	6	7	8
s-value	0.0000	0.9349	1.8699	2.8048	3.7397	4.6746	5.6098	6.5545	7.4794
t-value	0.0000	0.0841	0.2310	0.4304	0.6195	0.7332	0.8216	0.9061	1.0000

The arclength of a curve $\boldsymbol{X}(\tau)$ for $\tau \in [t_{\min}, t]$ is

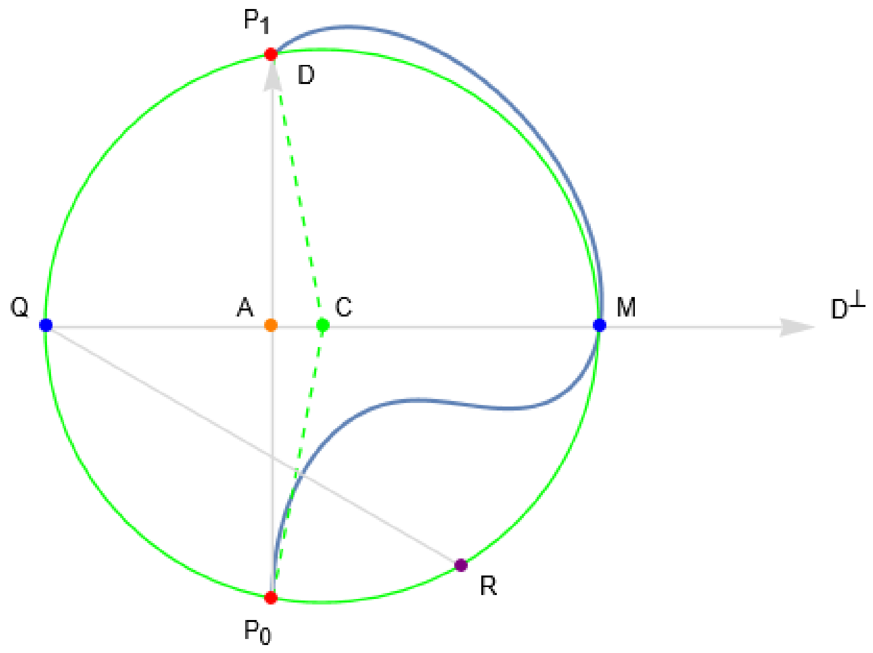
$$L(t) = \int_{t_{\min}}^{t} |\boldsymbol{X}'(\tau)|^2 \, d\tau \tag{1}$$

Generally, the integration must be implemented using a numerical method such as the Trapezoid rule, Simpson's rule, Gaussian quadrature, or other method. Subdivision by arclength requires specifying a length ℓ and determining what value of t produces this length; that is, we need to invert the arclength function $\ell = L(t)$ to obtain $t = L^{-1}(\ell)$. This also requires a numerical method and is referred to as *reparameterization by arclength*. A detailed description is found in Moving Along a Curve with Specified Speed.

2.2 Fit an Arc to a Subcurve

Figure 2 illustrates the concept for fitting an arc to a subcurve whose known endpoints are $\boldsymbol{P}_0$ and $\boldsymbol{P}_1$. Let the curve corresponding curve parameters be $\tau_i \in [t_{\min}, t_{\max}]$ for $i = 0, 1$. The midpoint of the line segment $\langle \boldsymbol{P}_0, \boldsymbol{P}_1 \rangle$ is the average $\boldsymbol{A} = (\boldsymbol{P}_0 + \boldsymbol{P}_1)/2$. The direction vector for the line segment is $\boldsymbol{D} = \boldsymbol{P}_1 - \boldsymbol{P}_0 = (d_0, d_1)$, which is not generally unit length. The vector perpendicular to $\boldsymbol{D}$ and of the same length is $\boldsymbol{D}^{\perp} = (d_1, -d_0)$. The aforementioned geometric quantities are all known. The points $\boldsymbol{M}$, $\boldsymbol{C}$, and $\boldsymbol{Q}$ are unknown quantities. These points lie on the bisector line for the line segment $\langle \boldsymbol{P}_0, \boldsymbol{P}_1 \rangle$.

Figure 2. Illustration of the various geometric quantities that occur when fitting an arc to a subcurve.



The algorithm involves computing a point $\boldsymbol{M}$ on the parametric curve that also lives on the bisector line. The parametric curve is continuous, so as you traverse the subcurve from $\boldsymbol{P}_0$ to $\boldsymbol{P}_1$, you must cross the bisector line. It is possible that the curve oscillates back and forth across the bisector line, so there can be many such crossings. It is sufficient to choose one point, call it $\boldsymbol{M}$. If there are multiple crossings, you need to choose more points for the subdivision. As the number n becomes sufficiently large, there will be only one crossing.

The crossing point is $\boldsymbol{M} = \boldsymbol{X}(\bar{t})$ for some $\bar{t} \in (\tau_0, \tau_1)$. Construction of $\boldsymbol{X}(\bar{t})$ uses bisection of the function $F(t) = \boldsymbol{D} \cdot (\boldsymbol{X}(t) - \boldsymbol{A})$ for $t \in [\tau_0, \tau_1]$. Notice that $F(t) > 0$ when $\boldsymbol{X}(t)$ is above the bisector line. In particular, $F(\tau_1) = +|\boldsymbol{D}|^2/2$. Also, $F(t) < 0$ when $\boldsymbol{X}(t)$ is below the bisector line. In particular, $F(\tau_0) = -|\boldsymbol{D}|^2/2$. Finally, $F(t) = 0$ when $\boldsymbol{X}(t)$ is on the bisector line. Sufficiently many bisection iterations are applied to locate $\bar{t}$ where $F(\bar{t}) = 0$. If floating-point arithmetic is used, the bisection can be implemented without requiring the caller to specify the maximum number of iterations. Listing 1 contains pseudocode for the bisection of $F(t)$ for $t \in [\tau_0, \tau_1]$.

Listing 1. Bisection of $F(t)$ for $t \in [\tau_0, \tau_1]$ using floating-point arithmetic. The loop is guaranteed to terminate because a sufficient number of iterations will either find a $\bar{t}$ where $F(\bar{t}) = 0$ using floating-point computations, or the interval to bisect has consecutive floating-point endpoints and the interval midpoint rounds to one of those endpoints. The root produces $\boldsymbol{M} = \boldsymbol{X}(\bar{t})$.

```
Point P0 = X(tau0), P1 = X(tau1), A = (P0+P1)/2, D = P1−P0;
Float tMin = tau0, tMax = tau1, tBar;

while (true)
{
    tBar = (tMin + tMax) / 2;
    Float fAtTBar = Dot(D, X(tBar) − A);
    int signFAtTBar = Sign(fAtTBar);
    if (signFAtTBar == 0 or tBar == tMin or tBar == tMax)
    {
        break;
    }

    if (signFAtTBar == −1)
    {
        tMin = tBar;
    }

    else // signFAtTBar = +1
    {
        tMax = tBar;
    }
}

Point M = X(tRoot);
```

We have 3 points that lie on an arc: $\boldsymbol{P}_0$, $\boldsymbol{P}_1$, and $\boldsymbol{M}$. A circle containing the points can be generated knowing that $\boldsymbol{M}$ is on the bisector line. Algebraically,

$$|\boldsymbol{C} - \boldsymbol{M}|^2 = |\boldsymbol{C} - \boldsymbol{P}_0|^2, \quad |\boldsymbol{C} - \boldsymbol{M}|^2 = |\boldsymbol{C} - \boldsymbol{P}_1|^2 \tag{2}$$

Define $\boldsymbol{C} = (c_0, c_1)$, $\boldsymbol{\Delta}_i = \boldsymbol{P}_i - \boldsymbol{M} = (\Delta_{i0}, \Delta_{i1})$ and $\boldsymbol{H}_i = (\boldsymbol{P}_i + \boldsymbol{M})/2$ for $i = 0, 1$. Expanding the equalities and cancelling the common $|\boldsymbol{C}|^2$ terms,

$$\boldsymbol{\Delta_i} \cdot \boldsymbol{C} = (\boldsymbol{P}_i - \boldsymbol{M}) \cdot \boldsymbol{C} = \left(|\boldsymbol{P}_i|^2 - |\boldsymbol{M}|^2\right)/2 = \boldsymbol{\Delta}_i \cdot \boldsymbol{H}_i = \lambda_i \tag{3}$$

where the last equality defines the λ_i. We have a linear system of two equations in two unknowns,

$$\begin{bmatrix} \Delta_{00} & \Delta_{01} \\ \Delta_{10} & \Delta_{11} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} \lambda_0 \\ \lambda_1 \end{bmatrix} \tag{4}$$

which has solution

$$\begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \frac{1}{\Delta_{00}\Delta_{11} - \Delta_{01}\Delta_{10}} \begin{bmatrix} \Delta_{11} & -\Delta_{01} \\ -\Delta_{10} & \Delta_{00} \end{bmatrix} \begin{bmatrix} \lambda_0 \\ \lambda_1 \end{bmatrix} \tag{5}$$

whenever the determinant $\delta = \Delta_{00}\Delta_{11} - \Delta_{01}\Delta_{10}$ is not zero. If not zero, the approximation produces a true arc. If zero, the arc represents a line segment whose endpoints are the arc endpoints. The center and radius components of the arc are set to the maximum floating-point numbers as a signal to the caller that the arc is a segment. In the implementation, a threshold test $|\delta| \geq \epsilon$ for $\epsilon \geq 0$ is provided. By setting $\epsilon > 0$ according to the needs of your application, the hope is to avoid inaccurate center and radius caused by floating-point rounding errors.

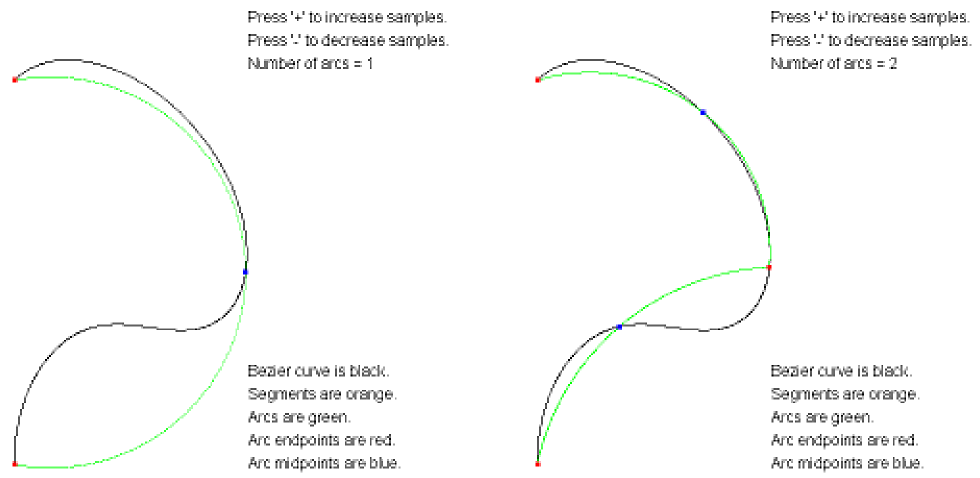
3 Visualization

A sample application that illustrates the arc fitting is

GeometricTools/GTE/Samples/Mathematics/ApproximateBezierCurveByArcs

The Bézier curve is degree 7, but as mentioned previously, any continuous curve allows for the arc fitting. Figure 3 shows the screen captures for 1 and 2 arcs.

Figure 3. The screen captures for 1 arc (left) and for 2 arcs (right). The text in the image indicates what the various rendered objects are.



Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 1

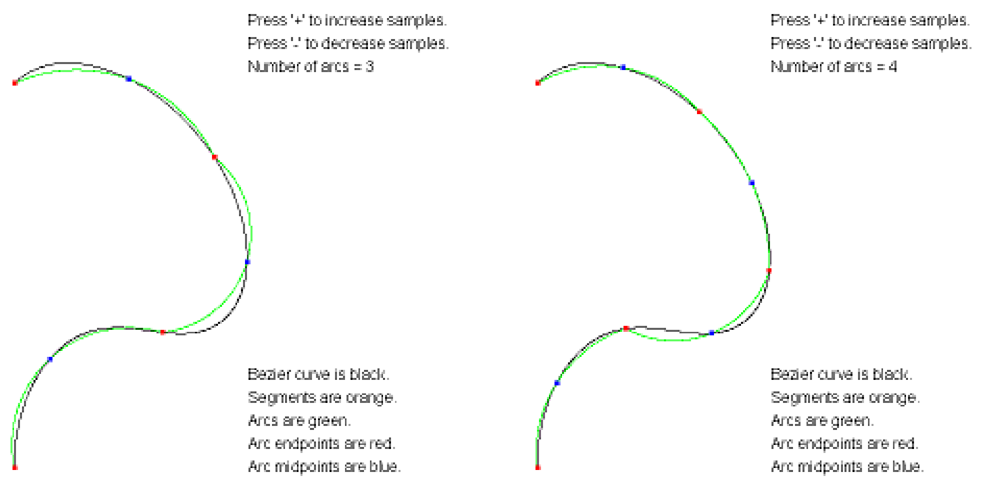
Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 2

Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

You can press the '+' key to increase the number of arcs one at a time up to 32 arcs. You can press the '-' key to decrease the number of arcs one at a time down to 1 arc. Figure 4 shows the screen captures for 3 and 4 arcs.

Figure 4. The screen captures for 3 arcs (left) and for 4 arcs (right).



Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 3

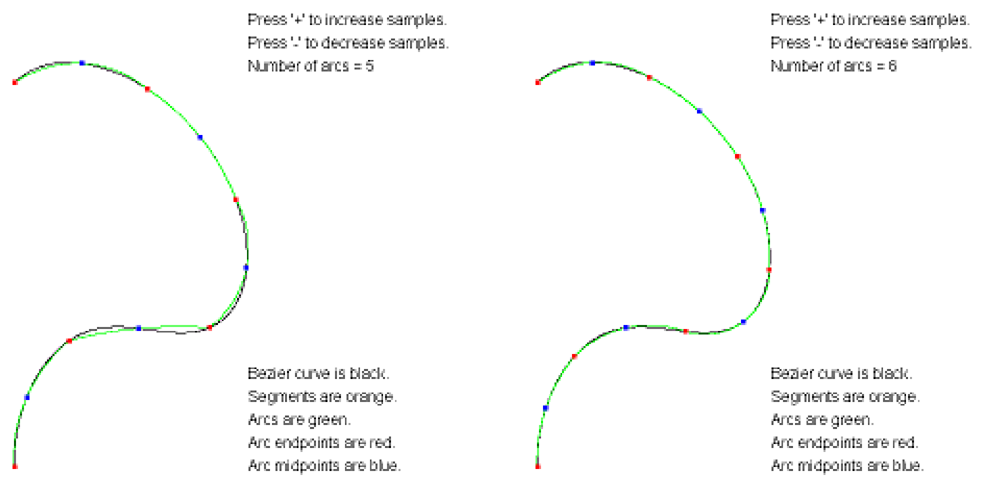
Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 4

Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Figure 5 shows the screen captures for 5 and 6 arcs.

Figure 5. The screen captures for 5 arcs (left) and for 6 arcs (right).



Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 5

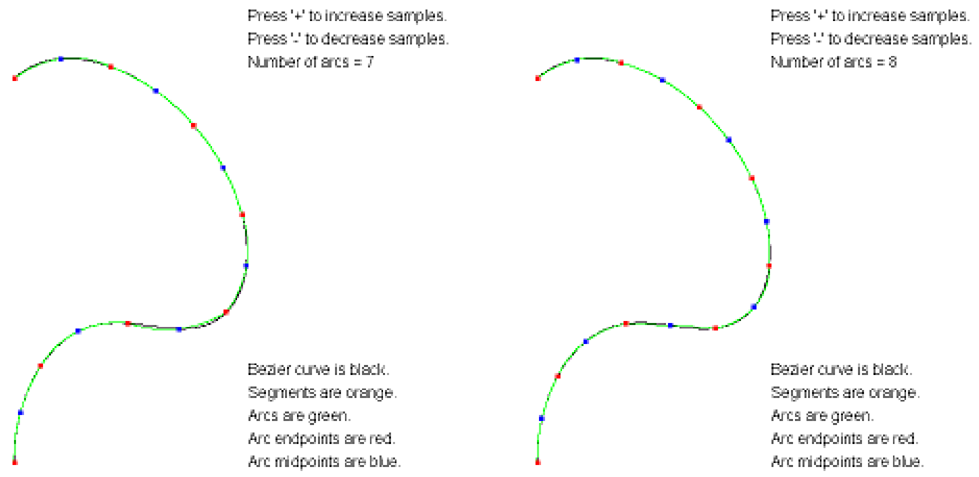
Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 6

Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Figure 6 shows the screen captures for 7 and 8 arcs.

Figure 6. The screen captures for 7 arcs (left) and for 8 arcs (right).



Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 7

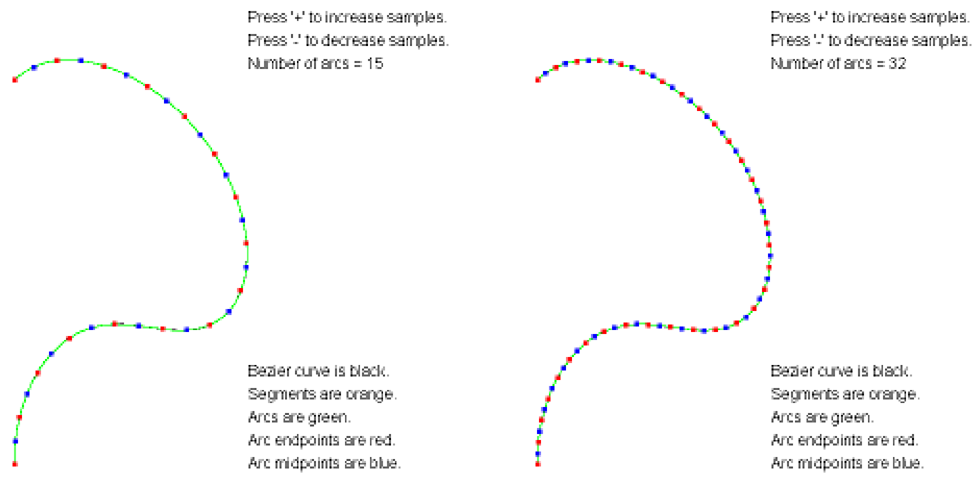
Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 8

Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Figure 7 shows the screen captures for 16 and 32 arcs.

Figure 7. The screen captures for 16 arcs (left) and for 32 arcs (right).



Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 15

Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Press '+' to increase samples.
Press '-' to decrease samples.
Number of arcs = 32

Bezier curve is black.
Segments are orange.
Arcs are green.
Arc endpoints are red.
Arc midpoints are blue.

Generally, as the number of samples in the subdivision increases, the error of approximation decreases.

The arc center can be subtracted from the arc endpoints to form endvectors. To render the arcs, one needs to distinguish between an acute angle between the endvectors and an obtuse angle between the endvectors. One way to do this without having separate algorithms for the two cases is the following. Figure 2 shows a point $\boldsymbol{Q}$ which is the antipodal point of $\boldsymbol{M}$ on the circle containing the arc. It is the case that $\boldsymbol{C} - \boldsymbol{Q} = \boldsymbol{M} - \boldsymbol{C}$, so $\boldsymbol{Q} = 2\boldsymbol{C} - \boldsymbol{M}$.

The line segment connecting the arc endpoints can be parameterized as $\boldsymbol{S}(u) = \boldsymbol{P}_0 + u\boldsymbol{D}$ for $u \in [0, 1]$. The ray with origin $\boldsymbol{Q}$ and containing $\boldsymbol{S}(u)$ intersects a point $\boldsymbol{R}$ on the arc. As such, we also know $|\boldsymbol{R} - \boldsymbol{C}|^2 = r^2$. Define $\boldsymbol{W}(u) = \boldsymbol{S}(u) - \boldsymbol{Q}$. We have $Q + v\boldsymbol{W}(u) = R$ for some $v > 0$; then

$$\begin{aligned} r^2 &= |\boldsymbol{R} - \boldsymbol{C}|^2 \\ &= |\boldsymbol{Q} - \boldsymbol{C} + v\boldsymbol{W}(u)|^2 \\ &= |\boldsymbol{Q} - \boldsymbol{C}|^2 + 2\boldsymbol{W}(u) \cdot (\boldsymbol{Q} - \boldsymbol{C})v + |\boldsymbol{W}(u)|^2 v^2 \\ &= r^2 + 2\boldsymbol{W}(u) \cdot (\boldsymbol{Q} - \boldsymbol{C})v + |\boldsymbol{W}(u)|^2 v^2 \end{aligned} \tag{6}$$

The common r^2 terms can be cancelled. The remaining expression has a factor v which can be divided out because we know $v > 0$. At this time we have a linear equation in v with solution

$$v = -\frac{2\boldsymbol{W}(u) \cdot (\boldsymbol{Q} - \boldsymbol{C})}{|\boldsymbol{W}(u)|^2} \tag{7}$$

The result is that we have a parameterization of the arc points in terms of u,

$$\boldsymbol{R}(u) = \boldsymbol{Q} - \frac{2\boldsymbol{W}(u) \cdot (\boldsymbol{Q} - \boldsymbol{C})}{|\boldsymbol{W}(u)|^2} \boldsymbol{W}(u) \tag{8}$$