Quadratic bezier offsetting with selective subdivision

Gabriel Suchowolski, Jul 10, 2012

Abstract

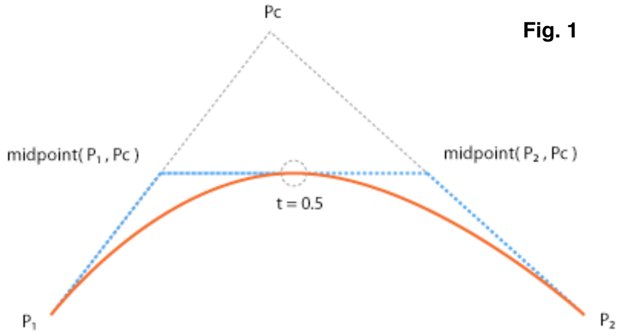


Fig. 1

Computer graphics made use of the quadratic bezier curves as a fast method to draw vector curves. Even if the curve is a cubic bezier, in the last term, it's represented by approximation using quadratic bezier paths or line segments. In different vector art applications, it's possible to draw bezier curves with different strokes widths, and this is done by using an offsetting algorithm.

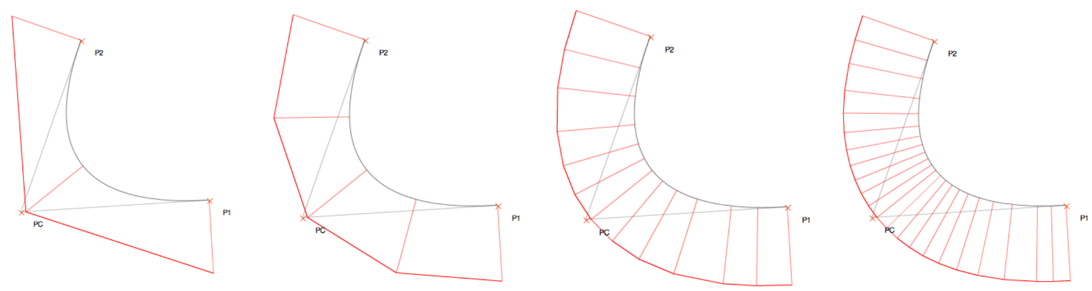
This document, **"Quadratic bezier offsetting with selective subdivision"** covers a method to offset quadratic beziers using a criterion that set the parametric value on which the quadratic bezier is subdivided at the start in order to generate an offset approximation with other quadratic beziers segments. This method, obviously, may not be the most perfect approximation of a hypothetical "real" offset, but a fast algorithm for drawing strokes that can be performed on different quality levels by using a **non recursive algorithm**.

The trouble with the midpoint

Paul de Casteljau discovers that any bezier curve can be subdivided into two bezier curve segments at an arbitrary parametric location (ex. midpoint t=.5, **Fig.1**) resulting two segments of bezier curves of the same degree. The algorithm is a very fast way for subdivide a bezier and is wide used on the computer graphics like vectorial design applications of today.

In other hand one of the most used algorithm to perform the quadratic bezier's offsetting consist in recursively subdivision of the curve at the **midpoint** (using Casteljau) and offset this points by a distance. The new resulting points finally are joined with rects or curves that approximate the real offset. **(Fig.2)**

Fig. 2



Quadratic bezier offsetting with selective subdivision

Gabriel Suchowolski, Jul 10, 2012

But one of the troubles with using this **midpoint** (t = .5) algorithm for offsetting, is that it needs, in some cases, a lot of iterations in order to get a good approximation of the real offset. (**Fig.3**)

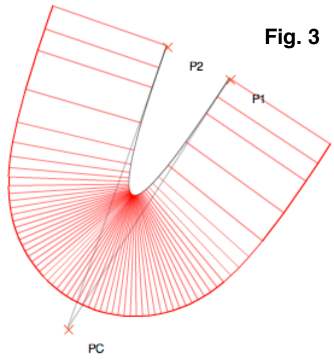


Fig. 3

The second troubles is that in some cases just the first iteration of the algorithm does a terrible result that can't be fixed on subsequent passes of the algorithm (**Fig.4a** with rects, **Fig.4b** with curve joints). This is because most of the offsetting algorithm fix this situation with a cusp, that can be an arc of a circle or a cut line. For example an offset (stroke) curve with Flash CS6 show some anomalies. (**Fig.4c**)

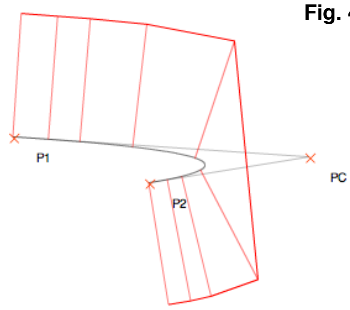


Fig. 4a

There exist ways to fix that anomalous situations, and can be "corrected" on subsequent iterations of the algorithm, for example checking on each pass of the subdivision to various conditions, like "flatness", that give us information about the quality of the approximation. But again there is a need to subdivide several times in order to solve the cusp anomalies.

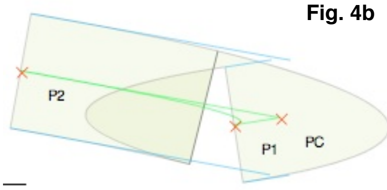
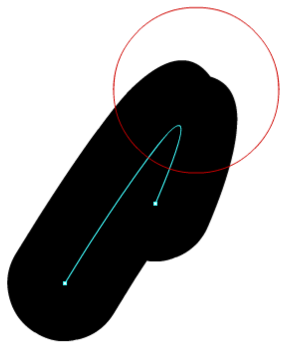


Fig. 4b

So the first subdivision on the **midpoint** ($t = 1/2$) maybe not the best selection to start.

The current paper **"Quadratic bezier offsetting with selective subdivision"** will explain an alternative way to get to the offsetting of a quadratic bezier not using the **midpoint** approach.

Fig. 4c



Instead we will start the subdivision on a different parametric location that will help to avoid the anomalous situations from the beginning.

Quadratic bezier offsetting with selective subdivision

Gabriel Suchowolski, Jul 10, 2012

The nearest point to the control point Pc

Be a quadratic bezier defined by three points $P_1,\ P_c,\ P_2$. The curve start at P_1 and ends on P_2. At this two locations the tangents have the directions of the vectors $(P_c - P_1)\ and\ (P_2 - P_c)$ respectively. P_c is a control point.

As the bezier “bends” towards P_c and P_c is inside the triangle formed by $P_1,\ P_c,\ P_2$ must be a point P_t at the quadratic bezier curve that is the most near point to P_c. This idea is more evident when the triangle $P_1\ P_c\ P_2$ gets more closed. (**Fig. 5**)

So a good point to choose to start the subdivision looks to be that P_t

Because this point will be the local cusp of our quadratic bezier towards P_c and will subdivide the quadratic bezier where the curve bends.

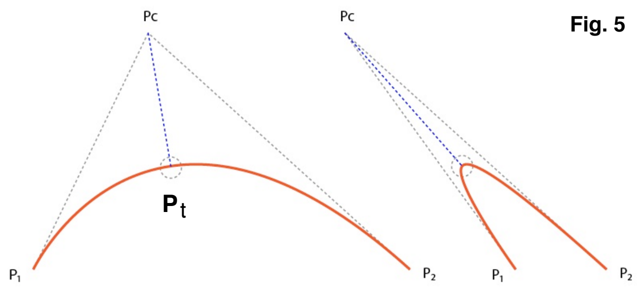


Fig. 5

Quadratic bezier offsetting with selective subdivision

Gabriel Suchowolski, Jul 10, 2012

P_t calculations

Let be the function of a quadratic bezier in the Bernstein polynomial form:

$$P(t) = (1-t)^2 P_1 + 2t(1-t)P_c + t^2 P_2 \quad where \quad t \in [0,1]$$

The nearest point $P_t \in P(t)$ with $t \in [0,1]$ to P_c will be the one that the vector from P_c to $P(t)$ is perpendicular to the tangent at this t, so:

$$(P(t) - P_c) \cdot P'(t) = 0$$

Where $P'(t)$ is the first derivate of $P(t)$ and the · is dot/scalar product.

Where

$$P'(t) = 2\ [\ t\ [\ (P_2 - P_c) - (P_c - P_1)\ ] + (P_c - P_1)\ ]$$

Be

$$V_0 = (P_c - P_1)$$
$$V_1 = (P_2 - P_c)$$

Then

$$P'(t) = 2\ [\ t\ [\ V_1 - V_0\ ] + V_0\ ]$$

And

$$P(t) - P_c = (1-t)^2 P_1 + 2t(1-t)P_c + t^2 P_2 - P_c = t^2(V_1 - V_0) + t(2V_0) + (-V_0)$$

So

$$(P(t) - P_c) \cdot P'(t) = 0$$

That is a third degree polynomial

$$t^3[\ a\ ] + t^2[\ b\ ] + t[\ c\ ] + [\ d\ ] = 0$$

where

$$a = (V_1 - V_0) \cdot (V_1 - V_0)$$
$$b = 3(V_1 \cdot V_0 - V_0 \cdot V_0)$$
$$c = 3V_0 \cdot V_0 - V_1 \cdot V_0$$
$$d = -V_0 \cdot V_0$$

This will give us at last one solution for $t \in [0,1]$ that will give the nearest point $P_t \in P(t)$ to P_c that we are looking for.

Quadratic bezier offsetting with selective subdivision

Gabriel Suchowolski, Jul 10, 2012

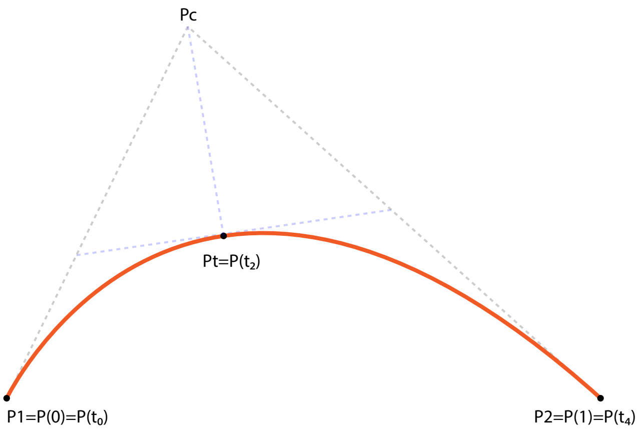
Quadratic bezier offsetting algorithm using Pt

This algorithm is a non recursive algorithm, so it's executed the number of times the subdivision of the quadratic bezier will be occur. The minimum subdivision of the curve is one (divided only at P_t)

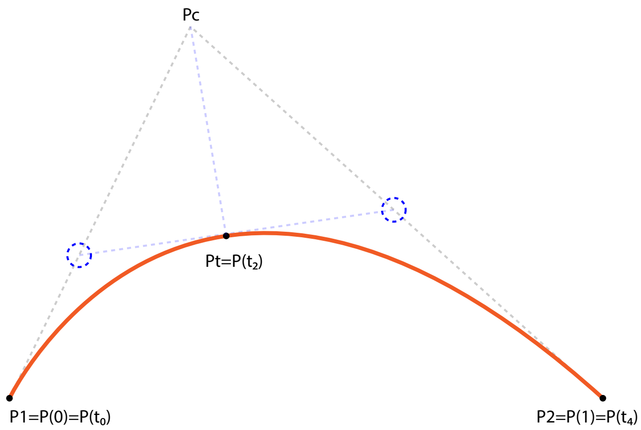
This is an example of the algorithm with **two pass of subdivision**:

---

1. Get P_t that is nearest point of the quadratic bezier to P_c

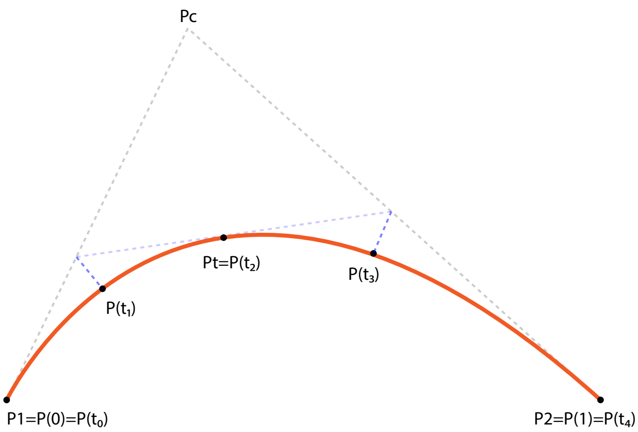


2. Get "control points" of subdivided bezier throw P_t

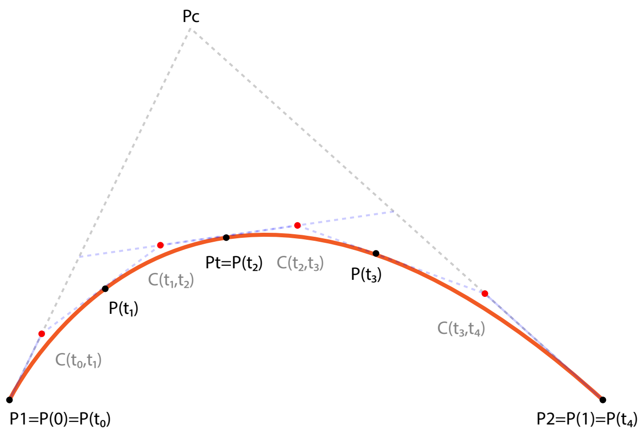


---

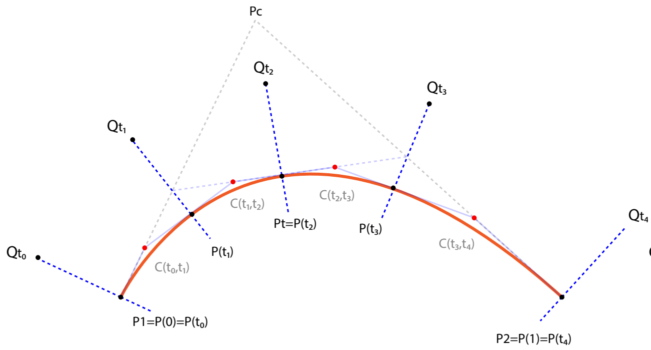
3. Get new subdivision points (Nearest points of the quadratic bezier to the new "control points")



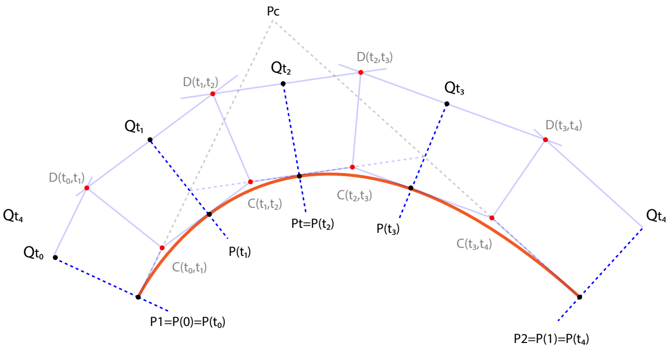
4. Get "control points" of the quadratic bezier segments at $P(t_i),\ P(t_{i+1}) \rightarrow C(t_i,\ t_{i+1})$



5. Get “offset points” Q_{t_i} that are offset from $P(t_i)$ at a distance in the perpendicular direction at each point.



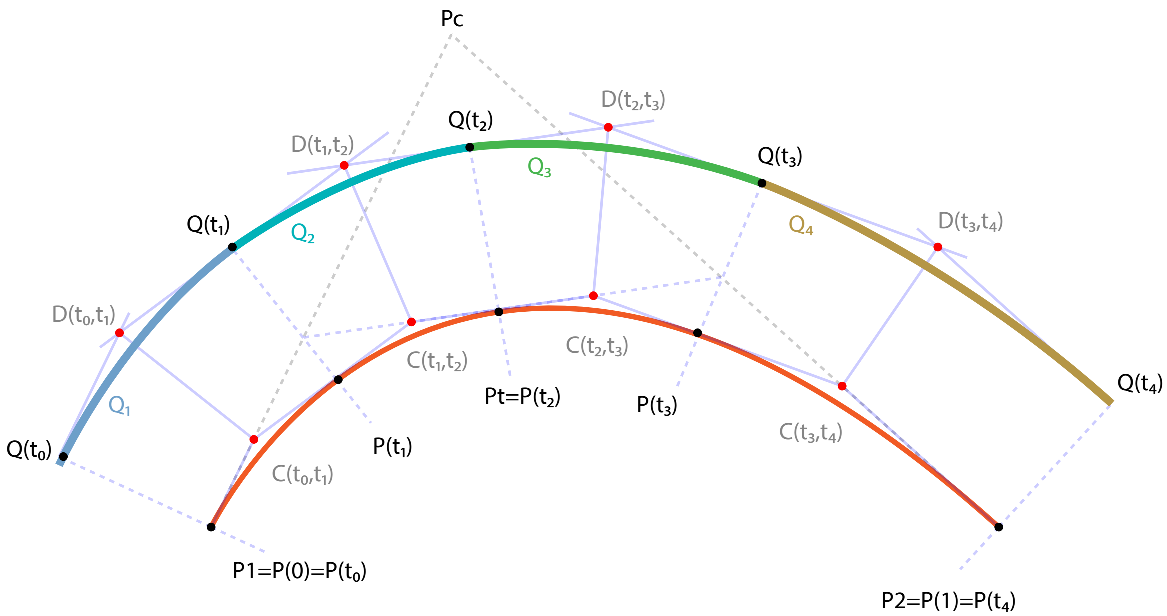
6. Get the $D(t_i,t_{i+1})$ points that will be “control points” of each bezier segment for the offset approximation of the quadratic bezier.



7. Draw the offset approximation path by sewing segments of quadratic beziers, each segment (colored segments in the picture) is represented as:

$$Q_i(t) = S(Q(t_i),\ D(t_i,t_{i+1}),\ Q(t_{i+1}))$$

Where $S(S_1,\ S_C,\ S_2)\ =\ (1-u)^2 S_1\ +\ 2u(1-u)S_C\ +\ u^2 S_2\quad where\ \ u \in [0,1]$



Note: Of course the process is the same for the other side of the offset.

Quadratic bezier offsetting with selective subdivision

Gabriel Suchowolski, Jul 10, 2012

Notes on 4th step of the algorithm.

We need control points $C(t_i,t_{i+1})$ of the quadratic bezier segment between two consecutive $P(t_i)$, $P(t_{i+1})$ points of the quadratic bezier. $C(t_i,t_{i+1})$ lie on the intersection of the two tangent lines at this points. So using the Casteljau's bezier curve subdivision algorithm, the control point of the quadratic bezier segment is:

$$C(t_i,\ t_{i+1}) = (1-t_{i+1})C_1(t_i) + t_{i+1}\,C_2(t_i)$$

Where

$$C_1(t_i) = (1-t_i)\,P_1 + t_i\,P_c$$

$$C_2(t_i) = (1-t_i)\,P_c + t_i\,P_2$$

Notes on 6th step of the algorithm.

We need control points $D(t_i,t_{i+1})$ where **Fig.6** shows a segment.

With

$$k = \frac{d}{cos(\propto/2)}$$

$$\vec{k} = \frac{m}{|m|}\,k$$

$$|m_1| = |m_2| = 1$$

(unit vectors perpendicular to the tangents)

$$m = m_1 + m_2$$

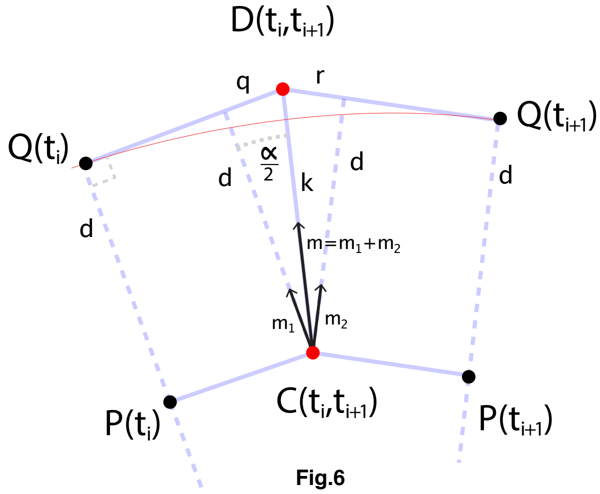


Fig.6

By applying the Law of Cosines:

$$|m_2|^2 = |m|^2 + |m_1|^2 - 2|m||m_1|\,cos(\propto/2)$$

So

$$|m|\ [\ |m| - 2\,cos(\propto/2)\ ] = 0$$

$$|m| = 2\,cos(\propto/2)$$

$$cos(\propto/2) = |m|\ /\ 2$$

Then

$$k = \frac{d}{cos(\propto/2)} = \frac{2d}{|m|}$$

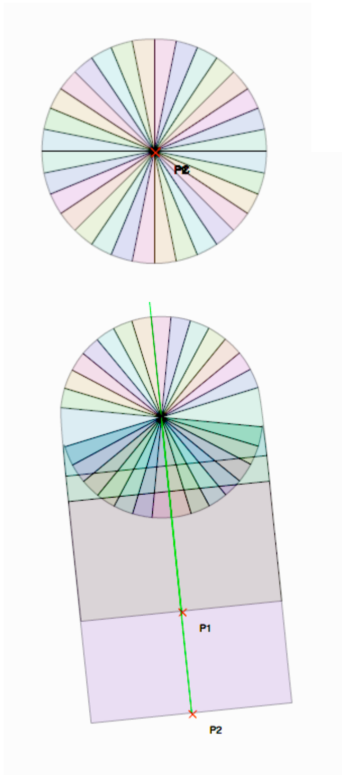
Then vector $\vec{k}$ from $C(t_i,t_{i+1})$ to $D(t_i,t_{i+1})$ is

$$\vec{k} = \frac{m}{|m|}\,k = \frac{2d}{|m|}\,\frac{m}{|m|} = m\,\frac{2d}{|m|^2} = m\,\frac{2d}{m\cdot m} \longrightarrow \boxed{D(t_i,t_{i+1}) = C(t_i,t_{i+1}) + m\,\frac{2d}{m\cdot m}}$$

Notes in 3rd step of the algorithm. Improvement of the subdivision

We can improve this step in some cases, for example when the angle at P_c formed by the triangle $P_1 \; P_c \; P_2$ is less than a limit (for example π/4): When this angle is more and more closed, the offsetting around P_t get more and more near to a circle arc **Fig.7**, so taking this in consideration we can do some adjustments.

Fig. 7



At this step we get the subdivision $P(t_2)$ point, then we check if the angle between tangents at P_t and $P(t_2)$ is bigger than an angle limit α (for example π/4). Then we will select another point P_m at $P(t)$ for the subdivision that satisfy that the angle between tangents at P_t and P_m is equal to α. So if angle of the tangent w_m at P_m is a_m and the angle of the tangent w_t at P_t is α_t, then the P_m we are looking for is the one that the angle of tangent at P_n is $\alpha_m = \alpha_t - \alpha$. So in order to get the new subdivision P_m we need to know the t in which the slope of the tangent vector $P'(t)$ the same slope of w_m (parallel vectors).

The slope of $P'(t)$ is

$$\frac{P'_y(t)}{P'_x(t)} = \frac{2\,[\; t\,[\,(P_{2y} - P_{Cy}) - (P_{Cy} - P_{1y})\,] + (P_{Cy} - P_{1y})\,]}{2\,[\; t\,[\,(P_{2x} - P_{Cx}) - (P_{Cx} - P_{1x})\,] + (P_{Cx} - P_{1x})\,]}$$

The slope of the tangent w_m at P_m is

$$s_m = \frac{w_{my}}{w_{mx}}$$

Then we are looking for the t that

$$s_m = \frac{P'_y(t)}{P'_x(t)} \quad \text{then}$$

$$t = \frac{s_m\,(P_{Cx} - P_{1x}) - (P_{Cy} - P_{1y})}{[\,(P_{1y} - P_{Cy}) - (P_{Cy} - P_{2y})\,] + (P_{Cx} - P_{1x})\,] - s_m[\,(P_{1x} - P_{Cx}) - (P_{CX} - P_{2x})\,]}$$

References

1. *Algorithme de De Casteljau, P. de Casteljau. Outillages methodes calcul. Technical report, A Citroen, Paris 1959*
2. *Piecewise Linear Approximation of Bezier Curves Kaspar Fischer October 16, 2000*
3. *Recursive Subdivision, Jim Amstrong 2112 F/X TN-06-005, Octiver 2006*
4. *Gernot Hoffmann Bézier Curves*
5. *Hausdorff distance between the offset curve of quadratic bezier curve and its quadratic approximation, Young Joon Ahn 2007*
6. *Antigrain.com – Interpolation with Bezier Curves, A very simple method of smoothing polygons*