Cubic Bézier interpolation algorithm for rendering smoothly curves on virtual paper

Salah J. Mohamed
salah.j.mohamed@almamonuc.edu.iq

Nibras W. Jawad
nibras.w.jawad@almamonuc.edu.iq

Auras K. Hameed

Abstract

This paper presents suggested modified algorithm that enables the rendering of smoothly flowing curves on digital paper is presented in order to provide Providing quality in drawings in many applications.

To draw these curves, cubic Bézier interpolation had been used by generating intermediate points between each two points from a set of given points. A comparison will be made with the functions used in the MATLAB program to clarify the difference when using the proposed algorithm through drawings showing illustrate the advantages of this algorithm.

الخلاصة

يقدم هذا البحث خوارزمية معدلة مقترحة تمكن من عرض منحنيات تتدفق بسلاسة على الورق الرقمي من أجل توفير الجودة في الرسومات في العديد من التطبيقات.

لرسم هذه المنحنيات، تم استخدام طريقة بيزير الثلاثية عن طريق توليد نقاط وسيطة بين كل نقطتين من مجموعة نقاط معينة. سيتم إجراء مقارنة مع الدوال المستخدمة في برنامج MATLAB لتوضيح الفرق عند استخدام الخوارزمية المقترحة من خلال رسومات توضح مزايا هذه الخوارزمية.

1. Background:

Drawing by hand is a particularly appealing way to provide an easy-to-use design interface for modeling. Using a stylus, light pen, mouse, or wand in the same way one would draw with a pencil, the designer sketches a curve using this method. Usually, all the data points are interpolated or some other approach is used to estimate the data [2].

Pen and paper had been the standard writing implements in the past. This trend started to progressively shift with the recent introduction of electronic tablets. For touchscreen devices, note-taking apps were created to make taking notes more convenient. These applications use a variety of methods, including line thinning, object mixing, and line anti-aliasing, to as closely replicate the traditional handwriting experience as possible. While these programs make an effort to develop an interface that can take the place of a pen and paper, it is presently exceedingly challenging to accurately imitate the feeling of writing on paper. As a result, there is a lack of documentation on the strategies used to develop such an interface, particularly for the iOS. This paper describes an iPad application with a unique pen-and-paper writing interface to do this.. When drawing by coordinates with mouse events, you get just connected lines. We looked at several algorithms that make a smooth curve, and we choose Cubic Bézier interpolation algorithm for post-processing, when the user has finished drawing, and even then, the lines turn into a curve. This technique used in computer graphics and design software to create curves that are aesthetically pleasing and visually appealing [5].

2. Cubic Bézier interpolation algorithm:

Cubic Bézier interpolation is a mathematical algorithm used to render smooth curves on virtual paper.

The algorithm works by defining four control points that determine the shape of the curve. The first and last control points are the endpoints of the curve, while the two middle control points determine the curvature of the curve.

To calculate the position of any point on the curve, the algorithm uses a parametric equation that takes into account the position of each control point and a parameter value between 0 and 1. The parameter value determines how far along the curve a point should be placed.

Here we have four control points p_0; p_1; p_2; p_3 and let $t \in R$. We calculate a curve point using the construction below:

$$\begin{aligned}
\mathbf{p}_0^1(t) &= (1-t)\mathbf{p}_0 + t\mathbf{p}_1 \\
\mathbf{p}_1^1(t) &= (1-t)\mathbf{p}_1 + t\mathbf{p}_2 \\
\mathbf{p}_2^1(t) &= (1-t)\mathbf{p}_2 + t\mathbf{p}_3 \\
\mathbf{p}_0^2(t) &= (1-t)\mathbf{p}_0^1(t) + t\mathbf{p}_1^1(t) \\
\mathbf{p}_1^2(t) &= (1-t)\mathbf{p}_1^1(t) + t\mathbf{p}_2^1(t) \\
\mathbf{p}_0^3(t) &= (1-t)\mathbf{p}_0^2(t) + t\mathbf{p}_1^2(t).
\end{aligned} \tag{1}$$

Where:

- p(t) is the position of a curve point at the value of the parameter t
- p_0 is the starting point (first control point)
- p_1 is the first middle control point
- p_2 is the second middle control point
- p_3 is the ending point (last control point)

By combining the first three equations with the following two, we get:

$$\begin{aligned}
\mathbf{p}_0^2(t) &= (1-t)^2\mathbf{p}_0 + 2(1-t)t\mathbf{p}_1 + t^2\mathbf{p}_2 \\
\mathbf{p}_1^2(t) &= (1-t)^2\mathbf{p}_1 + 2(1-t)t\mathbf{p}_2 + t^2\mathbf{p}_3.
\end{aligned} \tag{2}$$

Again, we insert these two equations into the last one, and get this:

$$\mathbf{p}_0^3(t) = (1-t)^3\mathbf{p}_0 + 2(1-t)^2t\mathbf{p}_1 + (1-t)t^2\mathbf{p}_2 + (1-t)^2t\mathbf{p}_1 + 2(1-t)t^2\mathbf{p}_2 + t^3\mathbf{p}_3.$$

(3)

The formula for calculating a point on a cubic Bézier curve is as follows:

$$\mathbf{p}_0^3(t) = (1-t)^3\mathbf{p}_0 + 3(1-t)^2 t\mathbf{p}_1 + 3(1-t)t^2\mathbf{p}_2 + t^3\mathbf{p}_3. \tag{4}$$

The construction is much the same as in the quadratic case as , p_0^3 is our point on the curve at parameter value t. Figure 1 illustrates the geometric arrangement for t = 1/2:

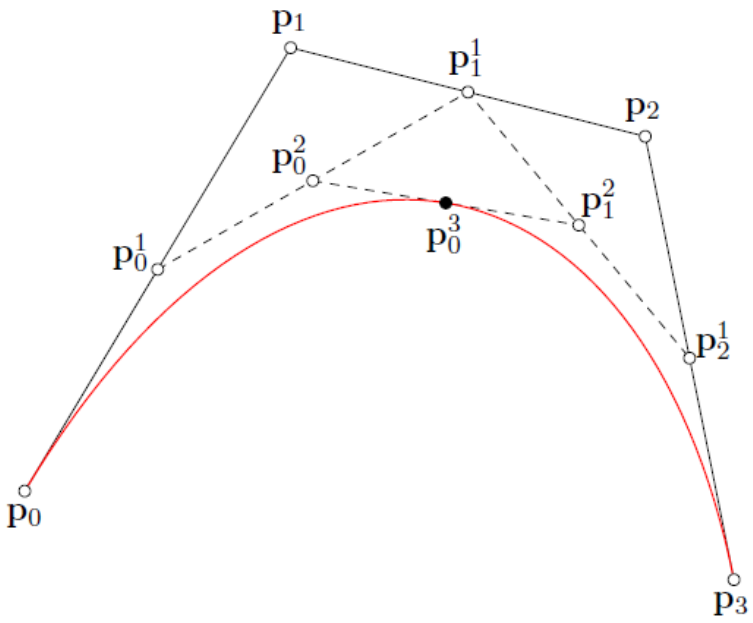

Figure 1: Line interpolation over and over on a cubic

As this is a cubic expression in t, as shown by equation (4), the resulting curve is also cubic. Because four control points can exist in space as well as on a plane, this is the first curve type that can construct space curves.

The point p_0^3 on this curve, which is affine invariant as well, needs to be computed using 6 linear interpolations. If we examine the curve shape, we can see that the curve is perpendicular to the line p_0p_1 for t = 0, and to the line p_1p_2 for t = 1. For the parabola, we have already mentioned this point. The four control points' convex hull always contains a cubic curve.

Compared to parabolas, parametric cubic curves have a significantly wider range of possible forms, including inflection points, nodes (points of self-intersection), and even a cusp (point, in which the curve has two tangents). As a result, Postscript, PDF, or vector drawing and CAD systems employ these cubic curves as the primary curve shapes [3].

By varying the positions of these four control points, different shapes and curves can be created. The algorithm ensures that these curves are smooth by ensuring that they have continuous first and second derivatives at each endpoint.

Overall, cubic Bézier interpolation is an effective way to create smooth curves in virtual environments. Its versatility makes it useful for many applications, including graphic design, animation, and gaming.

3. Experimental Results

After executing the program, it is possible to draw using the mouse and using the pen, the following Figures obtained

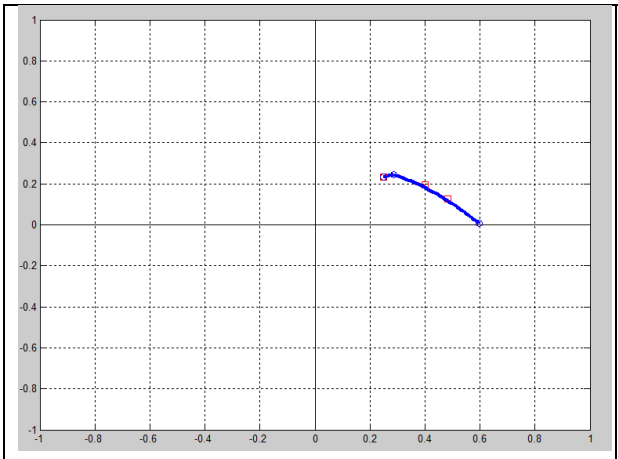

Figure 2

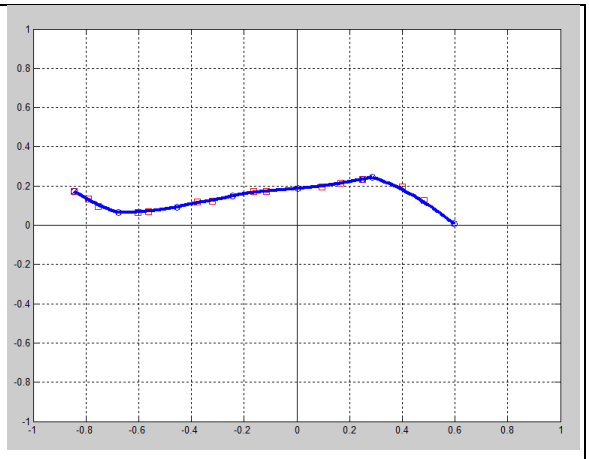

Figure 3

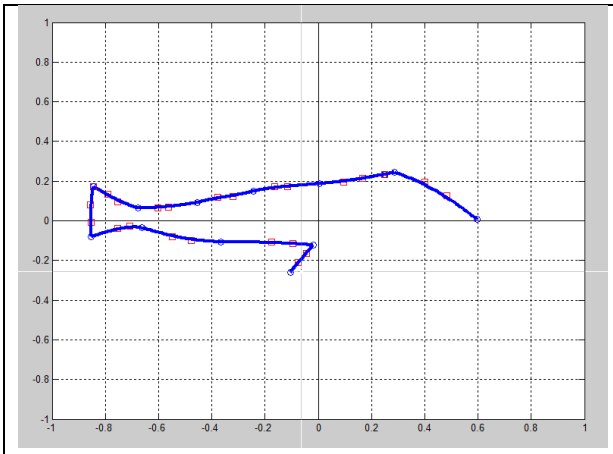

Figure 4

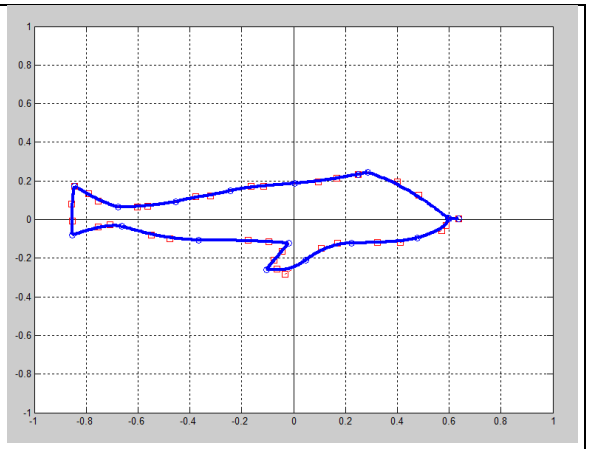

Figure 5

The results were compared with FREEHANDDRAW function (Matlab) to obtain smooth curves using hand drawing

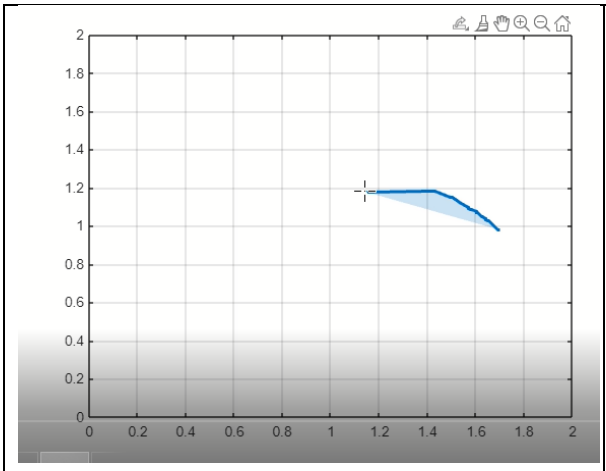

Figure 6

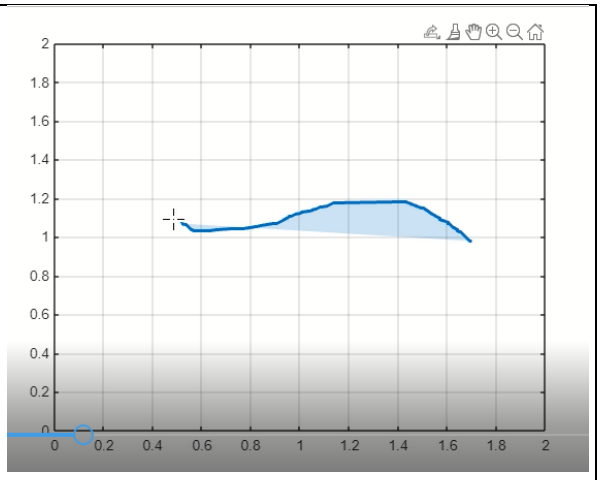

Figure 7

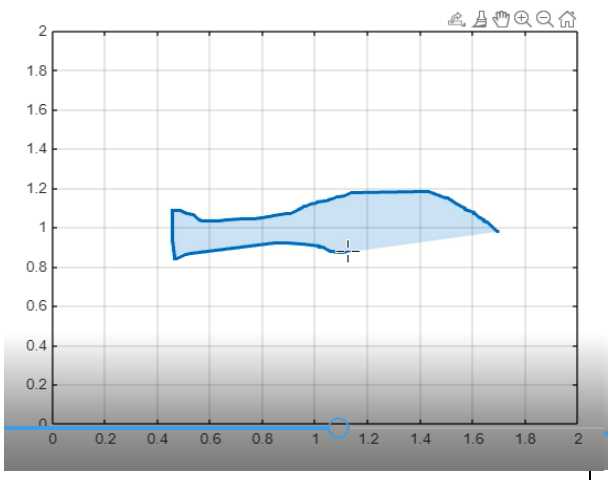

Figure 8

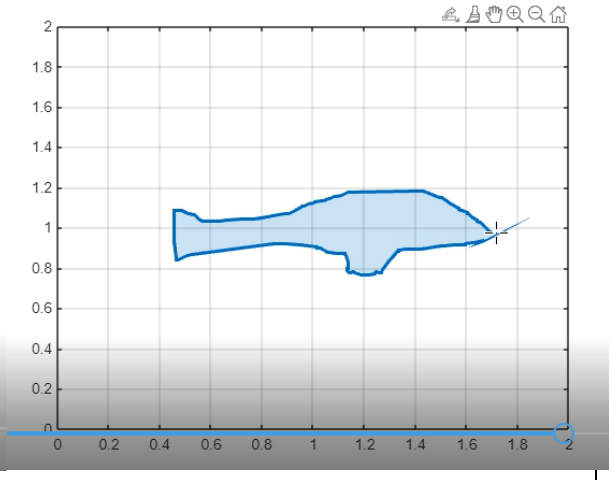

Figure 9

4. Conclusion

The results showed the preference of the suggested modified algorithm over the method used by MATLAB function FREEHANDDRAW.

The proposed method can be considered easy and low-cost method for drawing curves compared with other methods for this application. This method solves the problems of overshooting and corners that appear, if it is used for interactive curves. Also, the proposed method can be developed to be used in many applications that needs for rendering of smoothly flowing curves on digital paper is presented in order to provide documentation for the application's implementation. According to the test results, Writability succeeds in offering a competitive natural virtual pen-and-paper solution.

References

[1]. **J. Park and Y. Kwon, "An Efficient Representation of Hand Sketch Graphic Messages Using Recursive Bezier Curve Approximation", 2004**

[2]. **M. J. Bankst and E. Cohen, "Real time Pipelined Spline Data Fitting for Sketched Curves",1991**

[3]. **D. Reimers, "An Introduction Bezier Curves", 2011**

[4]. **B. Watson and L. F. Hodges, "Fast Algorithms For Rendering Cubic Curves",1992**

[5]. **K. S. Silvoza, R. A. Blonna, R. O. Atienza, "A Natural Handwriting Algorithm for Tablets", 2014**

[6]. **M. F. baharuddin, "Bézier method for image processing", 2010**

[7]. **T. K. and A. Yamaguchi, "A Method to Generate Freeform Curves from a Hand-drawn Sketch",2013**

[8]. **S. T. Arasteh, A. Kalisz, "Conversion Between Cubic Bezier Curves and Catmull–Rom Splines", 2021**

[9]. **Z. Sun, W. Wang, Li. Zhang, and J. Liu," Sketch Parameterization Using Curve Approximation",2014**