INVERSE POINT SOLUTION OF BEZIER CURVE

Syed Belal[1], Shashank .Kr.Tripathi[1], Shashank Kaswdhan[1], Md. Nadeem Akhtar[1], Satish Kumar Dwivedi[2] ,Rahul Singh[2]

1-Student of B. tech (mechanical engineering), Buddha institute of technology, GIDA, Gorakhpur

2- Asst. professor (mechanical engineering), Buddha institute of technology, GIDA, Gorakhpur

ABSTRACT

Nowadays there are various implementations regarding the surface and curve sharpening or smoothening are in progress. We also worked on it with the reference of Bezier curve, in most of the case, either cubic Bezier curve or 4- degree Bezier curve is used. The existing solution prevails to find the parametric value for a lower degree curve and the method is reverse engineering. The aim of this paper is to present a software system for drawing a Bezier curve of any n-degree and to find out the parametric value t for a known point P (t) on any n-degree Bezier curve. We called it as inverse point solution of Bezier curve. We implemented this technique in Matlab. The inverse point solution is the mathematical technique to make n-degree curve tracing advance and simple. The use of metamodels or surrogate approximations in place of actual simulation models makes analysis realistic by reducing computational burden.

Key words - Bezier curve, Matlab, parametric value, reverse engineering, and simulation.

1 INTRODUCTION

The concept of Bezier curve was developed by **Pierre Bezier** in the 1970's while working for **RENAULT.** The Bezier curve is the parametric curve which is defined by minimum of three control points consisting of an origin, an endpoint and at last a control point. Unlike straight lines and circles where one of the variables can be set to discover a point on the line, with Bezier curves you sample as many times as required from ***t*** in **[0,1]** to obtain that many points. Each of these points returned is a function of the control points on the origin and endpoint. [6]-[7]

Bezier curve originally introduced by Paul De Casteljan in 1959. But it becomes famous shape only when Pierre Bezier, a French engineer at RENAULT, used it to design in automobiles in the 1970's. Bezier curve is widely used in many field such as industrial and computer-aided design, vector-based drawing, font design (especially in postscript font) and 3D modeling.

Mathematically, a parametric Bezier curve is defined by

$$B(t) = \sum_{i=0}^{n} P_i J_{n,i}(t) \qquad 0 \le t \le 1 \quad \ldots\ldots\ldots..(1)$$

Where nth order Bernstein equation or blending function is

$$J_{n,i}(t) = \binom{n}{i} t^i \,(1-t)^{(n-i)} \qquad \ldots\ldots(2)$$

Where $\binom{n}{i} = \frac{n!}{i!(n-i)!}$, & $i = 0 \ldots\ldots\ldots\ldots n$

The most commonly used Bezier curve of third order are fully defined by four control points, which represent as,

$$\mathbf{B}(t) = (1-t)^3\,\mathbf{p0} + 3(1-t)^2\,t\,\mathbf{p1} + 3(1-t)\,t^2\,\mathbf{p2} + t^3\,\mathbf{p3} \qquad \textbf{...... (3)}$$

Where **p0**, **p1**, **p2**, **p3** are the control points.

So the interesting thing about Bezier curve is that it is easy to work with theoretical program. There's only one problem is that the curve does not passes through control point [1]. The curve actually lies in convex hull of the control point. This means that the control points may not lie on the curve, which makes calculating tangent and normals (for use in 3D trigonometry) tedious. [6]

2 PROBLEM DESCRIPTIONS

What we want to do is to define four points and have a Bezier curve passing through all four points. Basically, given the four original points **q0**, **q1**, **q2** and **q3**, we will find four points **p0**, **p1**, **p2** and **p3** such that the Bezier curve calculated using points **p(i)**, will pass through the points **q(i)**. So going back to the equation **(3)** above, when **t** is **zero**, the equation effectively collapses into just **p0**. When t is one, the equation gives **p3**. When t is between zero and one, the resulting point lies on the curve itself, so iterating t from zero to one will give the Bezier curve. Since we know the curve will pass through **p0** and **p3**, we need to find **p1** and **p2**.

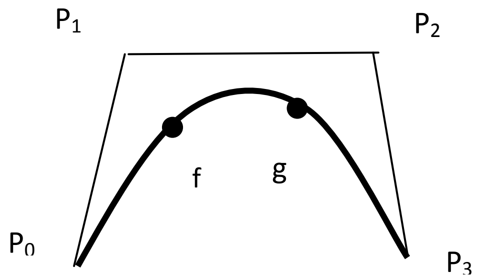


Fig.1 3 degree Bezier curve

Suppose we want the curve to pass through **p0** when t=0, **f** when t=u, **g** when t=v and **p3** when t=1, Where **f** and **g** are the points to be passed through. Next, we make sure that 0 < u, v < 1 and u not equal to v. These conditions will ensure a solution can be found. Next, we substitute the desired points into the equation:

$$\mathbf{f} = (1-u)^3\,\mathbf{p0} + 3(1-u)^2\,u\,\mathbf{p1} + 3(1-u)\,u^2\,\mathbf{p2} + u^3\,\mathbf{p3}$$

$$\mathbf{g} = (1-v)^3\,\mathbf{p0} + 3(1-v)^2\,v\,\mathbf{p1} + 3(1-v)\,v^2\,\mathbf{p2} + v^3\,\mathbf{p3}$$

The two equations are then simplified into

$$3(1-u)^2\,u\,\mathbf{p1} + 3(1-u)\,u^2\,\mathbf{p2} = \mathbf{c}$$

$$3(1-v)^2\,v\,\mathbf{p1} + 3(1-v)\,v^2\,\mathbf{p2} = \mathbf{d}$$

Where

$$\mathbf{c} = \mathbf{f} - (1-u)^3\,\mathbf{p0} - u^3\,\mathbf{p3}$$

$$\mathbf{d} = \mathbf{g} - (1-v)^3\,\mathbf{p0} - v^3\,\mathbf{p3}$$

We are assuming that u = 1/3 and v = 2/3, but they can be any value as long as 0 < u, v < 1 and u not equal to v (and logically u < v). It is likely that **f** is somewhere 1/3 of the way between **p0** and **p3**, and that **g** is somewhere 2/3 of the way between **p0** and **p3**. But it's not a given, so we still need to determine that. 1/3 and 2/3 just happens to be the "logical, and common-sensical" default.

This set of equations has a unique solution when 0 < u, v < 1 and u not equal to v, and assuming **c** and **d** aren't both zero vectors. The equations have a unique solution because the ***determinant*** is not zero. Let's transform the set of equations into matrix form before explaining what a determinant is

$$\begin{bmatrix} 3u(1-u)^2 & 3(1-u)u^2 \\ 3v(1-v)^2 & 3(1-v)v^2 \end{bmatrix} \begin{pmatrix} p1 \\ p2 \end{pmatrix} = \begin{pmatrix} c \\ d \end{pmatrix}$$

The determinant for the above 2 by 2 matrix on the left-most side is

$$3(1-u)^2\,u * 3(1-v)\,v^2 - 3(1-u)\,u^2 * 3(1-v)^2\,v$$

Factorising this, we get

= 9uv (1-u) (1-v) [(1-u) v – u (1-v)]

= 9uv (1-u) (1-v) [v –u v -u +u v]

= 9uv (1-u) (1-v) [v - u]

Since 9 obviously is not equal to 0, and 0 < u, v < 1 (so u, v not equal to 0 and (1-u), (1-v) not equal to 0) and u not equal to v (so v-u is not equal to 0), therefore, the determinant is not equal to 0. By a theorem in linear algebra, this means the set of (linear) equations has a unique solution. For a 2 by 2 matrix, the determinant can be obtained by taking the product of the top-left element and bottom-right element, then subtract the product of the top-right element and bottom-left element, like drawing a cross.

Next, we multiply the *inverse* of the 2 by 2 matrix on the left of both sides of the equation and we get

$$\begin{pmatrix} p1 \\ p2 \end{pmatrix} = \frac{1}{det} \begin{bmatrix} 3(1-v)v^2 & -3(1-u)u^2 \\ -3v(1-v)^2 & 3u(1-u)^2 \end{bmatrix} \begin{pmatrix} c \\ d \end{pmatrix}$$

Note that inverse will cancel the matrix on left side. The inverse of(2 by 2 matrix) is obtained by swapping of top left and bottom right element, and then divide each element by the determinant. The determinant is non zero so division by zero is not a problem. A non- zero determinants means an inverse actually exist (by another theorem in linear algebra), so all of this works out.

2.1 PROBLEM WITH EXISTING SOLUTION: -

The method is appropriate for lower degree curves, but it becomes a very tedious task to solve the equation if degree of curve is higher.

The determinant of an n by n matrix is generally difficult to find, as is the inverse of one. Refer to a linear algebra text book for the theories (they usually use a method called **Gaussian elimination**). The programmatic approach uses a slightly modified version to reduce computational errors and easy to demonstrate a **(n) degree** Bezier curve having **(n+1)** control points and an easiest way to find out the parametric values of **t** which specifies in above equation **(3).[5]**

2.2 PROGRAMMABLE APPROACH:-

Fig.3 genera

We have developed a programme in Mat lab to solve a problem regarding the drawing of any degree of Bezier curve and also after drawing we

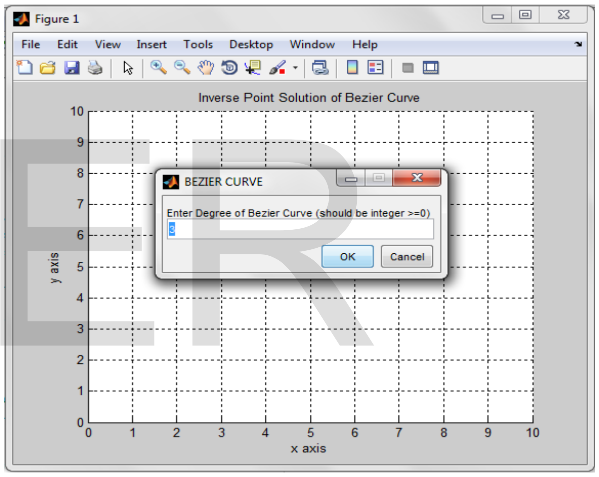


Figure 1
File Edit View Insert Tools Desktop Window Help
Inverse Point Solution of Bezier Curve
y axis
x axis
0
1
2
3
4
5
6
7
8
9
10
OK
Cancel

BEZIER CURVE

Enter Degree of Bezier Curve (should be integer >=0)

have to find out the parametric values, so we develop a programme in a way that after drawing Bezier curve everyone can also find out the

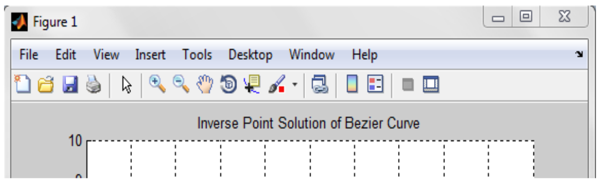
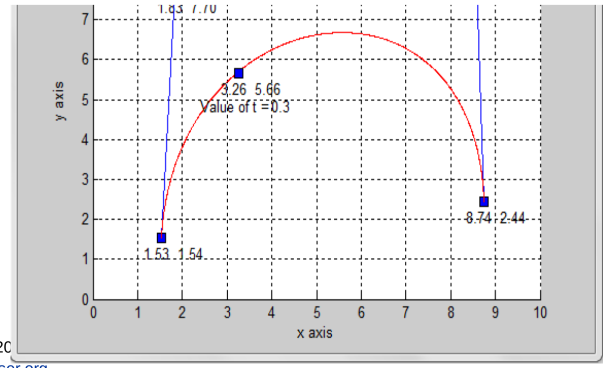
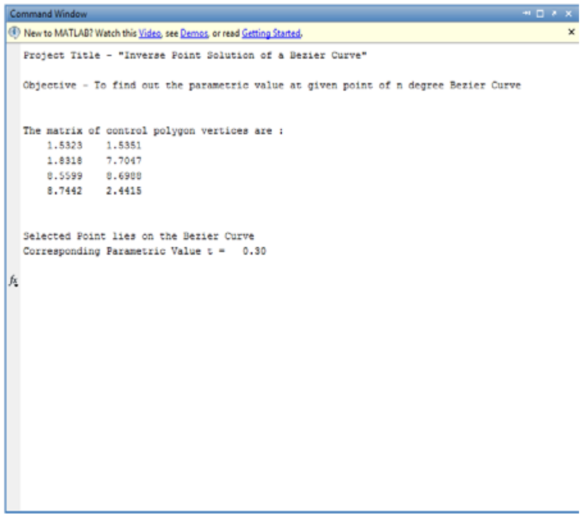


Figure 1
File Edit View Insert Tools Desktop Window Help
Inverse Point Solution of Bezier Curve
10

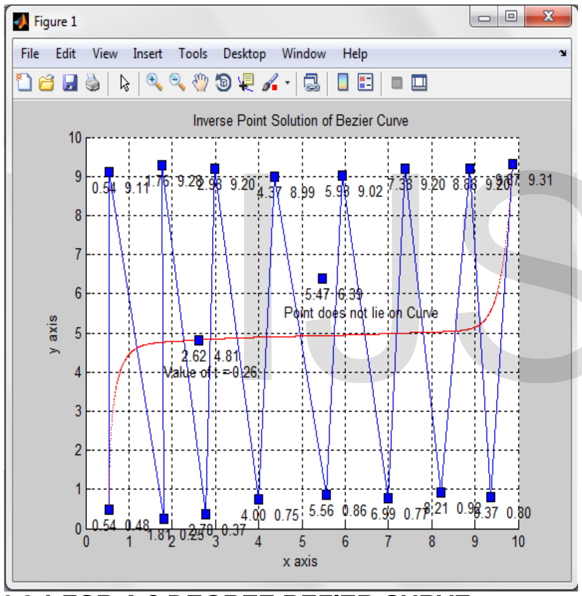
Fig.2 graphic space with degree specification dialogue box



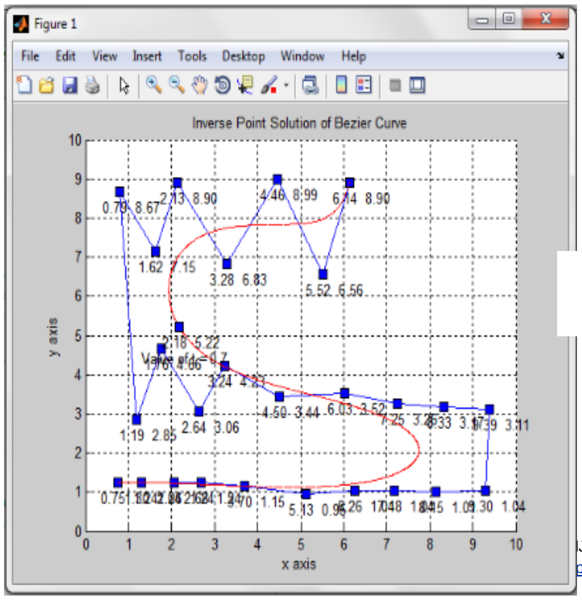

Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.

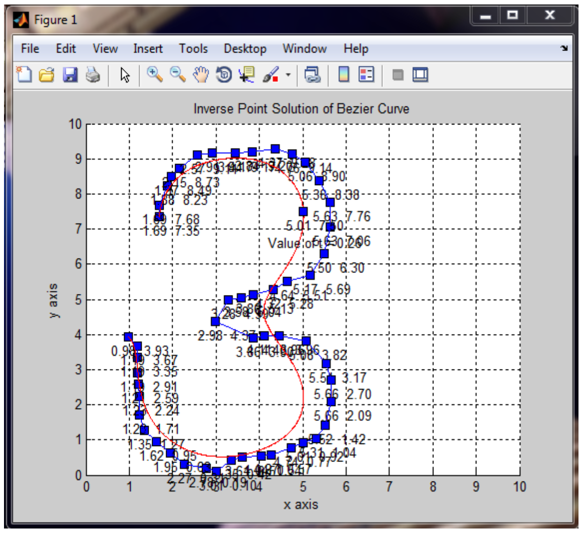
```
Project Title - "Inverse Point Solution of a Bezier Curve"

Objective - To find out the parametric value at given point of n degree Bezier Curve


The matrix of control polygon vertices are :
    1.5323    1.5351
    1.8318    7.7047
    8.5599    8.6988
    8.7442    2.4415


Selected Point lies on the Bezier Curve
Corresponding Parametric Value t =   0.30
```

parametric value.



2.2.1 FOR A 3 DEGREE BEZIER CURVE: -





Here we draw a Bezier curve of 3 degree using Matlab. Higher degree Bezier curves with their parametric values are shown on next page. In fig.2 we can see the graphic window with degree specification dialogue box. In fig.3, 3 degree Bezier

curve gets generated after running the program the red curve shows the Bezier curve and the blue dot on it shows the nearest parametric value of that curve at that point, dot which appear outside the red curve shows that the point does not exist on curve in fig.4 respected values of control polygon vertices are shown in the form of matrix which is program part. [2][4]

Fig.4 data sheet saved in matrix form with parametric values

iod

Fig.7 50 degree Bezier curve

such as Bezier Curves, B-Spline curves etc. For finding the value of parametric variable corresponding to a given point on the curve. This programme make

Fig.5 15 degree Bezier curve

tric value of

vas a main problem with existing solution using reverse engineering technique and also helpful in surface generation

4 ACKNOWLEDGEMENT:- It gives us an immense pleasure to present the technical paper on topic "***Inverse Point Solution of a Bezier Curve"*** undertaken during Bachelor of Technology final year. We would like to express our deep and sincere gratitude to "International Journal of Scientific and Engineering Research" Journal for the support to develop this document, our project guide for providing us excellent guidance, encouragement and inspiration throughout the project work. We would like to extend our special thanks to Head of Department of mechanical engineering & Project In charge for their immense support in all kind of work.

REFERENCES:-

[1]. F. yamaguchi. Curves and surface in computer aided geometric design, Springer, Verlag, 1988.
[2]. Introduction to Matlab, Dr. Sikander M. Mirza, department of physics and applied mathematics, Pakistan.
[3]. Yang, H.-M., Lu, J.-J., and Lee, H.-J.: 'A Bezier curve based approach to shape description of Chinese, calligraphy character', proceeding of sixth international conference on document analysis and recognition, 2001, pp.276-280.
[4]. Numerical Methods Using Mat lab, 4th Edition, 2004, John H. Mathews and Kurtis K. Fink
[5]. Reverse engineering Bezier curves, June 27, 2007 By Vincent.
[6]. Thomas W. Sederberg 'An Introduction to Bezier Curves' January 6, 2003.
[7]. Bezier curve fitting by Tim Andrew Pastva, September, 1998, naval post graduate school, Monterey, California.