Global Journal of Computer Science and Technology
Volume 11 Issue 18 Version 1.0 October 2011
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

A Robust Approach to Find the Control Points for Wide Variety of 3rd Order Bézier Curves

By Alok Kumar Chowdhury, Prithwi Raj Chakraborty, Md. Ibrahim Khan, Sujan Chowdhury

Premier University, Chittagong, Bangladesh

Abstract - This paper represents a new approach that can recover the control points for wide variety of 3rd order Bézier curves. In this regards, the two stage approximation learning algorithm is adopted with some modifications. At 1st stage our key feature is segmentation of the curve which can determine intermediate points of the wide variety of curves. In this respect, an efficient recursive algorithm is used to find out the height of the curve (h) with less iteration. The proposed approach introduced horizontal segmentation rather than vertical segmentation. Different height (H), where the 2nd and 3rd control point are assumed, and also the step-size (∂), at which the control points are moved toward the actual direction, are used to find out the exact location of the control points. Experimental results demonstrate that our proposing method can recover control points for wide variety of curves with minimum error level and less iteration. Wide variety of curve shapes are used to test the proposing approach and results are presented to prove its effectiveness.

Keywords : Bézier curve, curve fitting, segmentation of curve, learning algorithms.

GJCST-B Classification : I.3.5

A ROBUST APPROACH TO FIND THE CONTROL POINTS FOR WIDE VARIETY OF 3RD ORDER BEZIER CURVES

A Robust Approach to Find the Control Points for Wide Variety of 3rd Order Bézier Curves

Alok Kumar Chowdhury[α], Prithwi Raj Chakraborty[Ω], Md. Ibrahim Khan[β], Sujan Chowdhury[ψ]

Abstract - This paper represents a new approach that can recover the control points for wide variety of 3rd order Bézier curves. In this regards, the two stage approximation learning algorithm is adopted with some modifications. At 1st stage our key feature is segmentation of the curve which can determine intermediate points of the wide variety of curves. In this respect, an efficient recursive algorithm is used to find out the height of the curve (h) with less iteration. The proposed approach introduced horizontal segmentation rather than vertical segmentation. Different height (H), where the 2nd and 3rd control point are assumed, and also the step-size (∂), at which the control points are moved toward the actual direction, are used to find out the exact location of the control points. Experimental results demonstrate that our proposing method can recover control points for wide variety of curves with minimum error level and less iteration. Wide variety of curve shapes are used to test the proposing approach and results are presented to prove its effectiveness.

Keywords : Bézier curve, curve fitting, segmentation of curve, learning algorithms.

I. Introduction

The development of Computer Graphics has made computers easier to interact with, and better for understanding and interpreting many types of data which has put a profound impact on many types of media and have revolutionized animation, movies and the video game industry. Generally, Computer Graphics are the representation and manipulation of image data by a computer with help from specialized software and hardware [1].

In Computer Graphics and related fields, a frequently used parametric curve is Bézier curve. Bézier curves are used to model smooth curves that can be scaled indefinitely. As the curve is completely contained in the convex hull of its control points, the points can be graphically displayed and used to manipulate the curve intuitively. When more complex shapes are needed, Bézier curves are patched together. In Animation applications, such as Adobe Flash and Synfig, Bézier curves are used to outline, for example, movement. Besides,True-type Fonts use Bézier curves. Bézier curve is also a very powerful tool for shape approximation and shape comparison. In identifying of actors drawn in ukiyoe pictures, Bézier curves are widely used [2]. They can be used in face recognition and facial emotion detection [3].

A Bézier curve is defined by its order (Linear, Quadratic, Cubic, etc.) and a set of control points. An nth order Bézier curve has $n+1$ control points (P_1 to P_{n+1}). The first and last control points are always the end points of the curve. These two end points are can be called initial and terminating point of the curve respectively. The intermediate control points generally do not lie on the curve; they define the shape and direction of the curve. A 1st order (Linear) Bézier curve is simply a straight line between those two given points P_1 and P_2. A 2nd order (Quadratic) Bézier curve is the path traced by the function $B(t)$, given points P_1, P_2, and P_3. Four points P_1, P_2, P_3 and P_4 in the plane or in three-dimensional space define a 3rd order (Cubic) Bézier curve. The 3rd order curve starts at P_1 going toward P_2 and arrives at P_4 coming from the direction of P_3. P_1 and P_4 are end points [4-5]. Here P_1 and P_4 are end points as well as P_2 and P_3 are intermediate control points. A 3rd order Bézier curve given by the following equation is shown in fig. 1.

$$B(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2 (1-t) P_3 + t^3 P_4 \qquad (1)$$

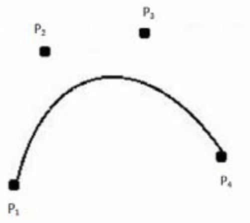


Fig.1 : A sample Bézier Curve

Our discussions will be limited to 3rd order Bézier curve. Though there are several algorithms and approaches to find out control point of 3rd order Bézier curve, these approaches have limitations like incorrect result, requirement of compatible boundary for control points etc. In our previous work [6], we tried to overcome these limitations and it finds control points of 3rd order Bézier curve in an efficient way. But it is still unable to find out control point of 3rd order Bézier curves of some certain shapes. In this paper we are going to propose a modified approach of our previous work to erase the existing limitations. Our new approach is capable of finding out the control points of a large variety of 3rd order Bézier curve shapes efficiently and accurately with minimum error and less iterations.

Author[α] : Department of Computer Science & Engg. Premier University, Chittagong, Bangladesh. Email : alokchy04@yahoo.com
Author[Ω] : Department of Computer Science & Engg. Premier University, Chittagong, Bangladesh.
Email : stranger_prithwi@yahoo.com
Author[β] : Department of Computer Science & Engg. Chittagong University of Engg. & Tech. (CUET), Chittagong, Bangladesh.
Email : muhammad_ikhancuet@yahoo.com
Author[ψ] : Department of Computer Science & Engg. Chittagong University of Engg. & Tech. (CUET), Chittagong, Bangladesh.
Email : sujan_cse_04@yahoo.com

II. Background

There has been several works in the field of recovering control points of Bézier curves. All the familiar algorithms and characteristics are associated with Bézier curves are generation [7], approximation [8], interpolation [9], subdivision [10-12], degree elevation [13-16], blossoming [17], implication [18] etc. X. Ye represented an approach for directly generating Bézier points of curves and surfaces explicitly from the given compatible arbitrary order boundary information of Hermite curves, Coons-Hermite Cartesian sum patches and Coons-Boolean sum patches [19]. But for Hermite curves, this approach requires the end positions, tangents and higher order derivatives at the end points. Moreover it requires the corner points and the compatible arbitrary order partial derivatives at these points for Coons–Hermite Cartesian sum patches.

In determining inner control points of 3rd order rational Bézier curve, the interpolation method developed by J. Chou and L. A. Piegl works good for special type of 3rd order Bézier curves, not all kind of Bézier curves [20]. Besides, this approach relies on the convex hull and on the variation diminishing properties of Bézier curves. And the most important drawback is that if the curve segment is highly curved in one region and relatively flat in another, the approximation of this method is not good.

The previous approach, proposed by us, contains none of these problems. It can successfully find out control points of 3rd order Bézier curve with minimum iteration and error. Fig. 2(b) and 2(c) are showing the way how the previous approach successfully finds out the control points for the curve of a particular shape indicated in fig. 2(a). The details are described at the previous work [6].

However, for some curves of shapes like fig. 3(a) and 4(a), the previous approach is unable to find out the accurate control points. It is because the segmentation process of the previous approach, which uses vertical segmentation, is different from the current one (described at Methodology section).

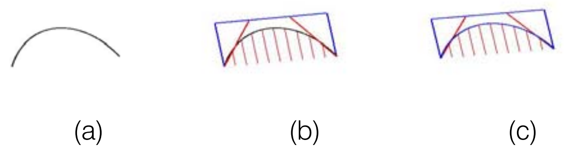


Fig.2 : (a) A typical Bézier curve, (b) Segmentation, (c) Successful-recovery of the curve (previous approach)

Fig. 3 and 4 are showing limitation of the previous approach in finding out the control points for some special shaped curves. The previous approach cannot rescue the whole area for these special shaped curves, indicated in fig. 3(c) and 4(c). Thus the control points found are also not correct for these curves. The curves drawn in blue color are the recovered curves.

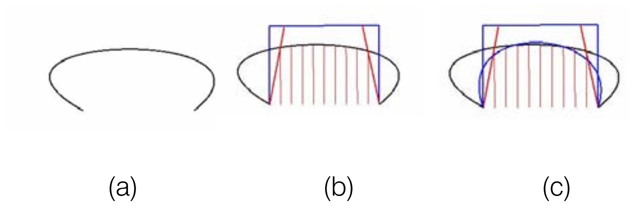


Fig.3 : (a) A typical Bézier curve, (b) Segmentation, (c) Failed-recovery of the curve (previous approach)

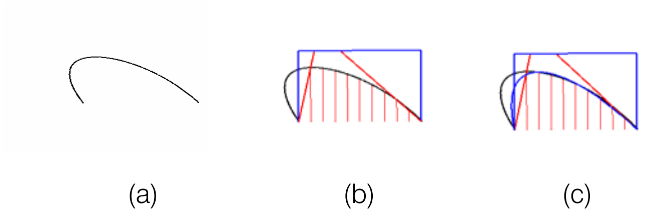


Fig.4 : (a) A typical Bézier curve, (b) Segmentation, (c) Failed-recovery of the curve (previous approach)

But the new approach, with modified segmentation method, is able to find out the actual control points very successfully with minimum iteration and error for all the 3rd order Bézier curves of shapes indicated in fig. 2(a), 3(a) and 4(a). It can rescue the whole area of special shaped curves which is described later in Simulation Results section.

III. Methodology

As it is mentioned before, a 3rd order Bézier curve contains four control points. The objective is to find out them. In order to do this, the new proposing approach is divided into two stages: modified first stage and second stage.

a) Modified first stage

The new proposing approach is actually different from our previous one because of this stage. This stage is described as following:

1) Finding out of the end points: The end points will be lying on the Bézier curve. End points must be those two which have minimum y values. Now it is to be decided which one of these end points is initial point and which one is terminating point. Among these two end points the point which is nearest from Y axis (i.e. x value is minimum) is our initial point, $P_1(x_1, y_1)$. So, the remaining one is for sure the terminating point, $P_4(x_4, y_4)$. These two end points are shown in fig. 5.

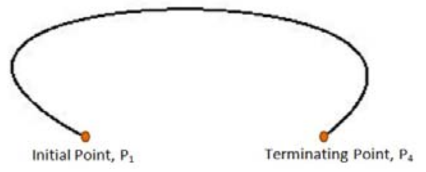


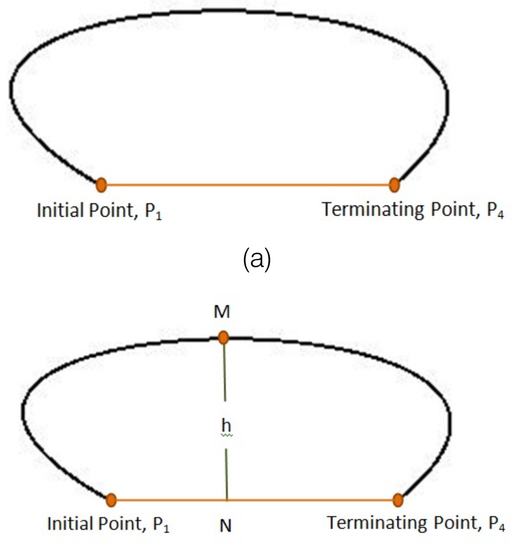
Fig.5 : End points of a Bézier curve

2) Finding out of the height: After finding out the initial and terminating point, the height of the Bézier curve, h is to be found by the following steps:

- A straight line between initial point $P_1(x_1, y_1)$ and terminating point $P_4(x_4, y_4)$ is drawn using the following equation:

$$(y - y_1)(x_4 - x_1) = (x - x_1)(y_4 - y_1) \qquad (2)$$

This line is called base line, P_1P_4 shown in fig. 6(a).



(b)

Fig.6 : (a) Base line, (b) Height and peak intersection point of a Bézier curve

- Now the height, h and the peak intersection point, M of the Bézier curve and the parallel line drawn at distance h from the base line are found by a recursive algorithm which makes the new approach faster than the previous one. This algorithm draws some parallel lines at some assumed distances, d from the base line. If a parallel line intersects the curve at two points, this line is stored and the next parallel line is drawn above the currently stored parallel line at a distance d. If the drawn parallel line does not intersect the curve then that line is not stored and the next parallel line is drawn above at a distance d / 2 from the previously stored parallel line. This algorithm contains three methods – draw_parallel(_line, _d), intersections(_curve, _line), and distance(_line1, _line2). The draw_parallel(_line, _d) method takes a line equation and a distance as inputs and returns the parallel line drawn at a distance _d from the line _line. If the equation ax + by + k = 0 be the value of input variable _line and d is the value of input variable _d, then draw_parallel(_line, _d) returns a parallel line using the following equation:

$$ax + by + [d(\sqrt{a^2 + b^2}) + k] = 0 \qquad (3)$$

The next method intersections(_curve, _line) takes a curve equation and a line equation and returns the number that represents at how many points the line intersects the curve. The peak most segment of the Bézier curve may be appeared as a tiny line rather than a curve. Therefore, the parallel line may intersect the curve at several consecutive intersection points. In that case, the middle one from the consecutive points is taken as the desired single intersection point and the number of intersections is considered as one. The last method distance(_line1, _line2) returns the distance between two lines. Now we should look at the recursive algorithm.

```
a. Set temp_base_line = base_line.
b. Repeat step c to g.
c. Set parallel_line = draw_parallel(temp_base_line, d).
d. Set no_of_intersections = intersections (curve, parallel_line).
e. If no_of_intersections = 1, then: Set M = intersection point of curve and parallel_line. Set h = distance(base_line, parallel_line) and Return h and M.
f. Else If no_of_intersections = 0, then: Set d = d / 2.
g. Else If no_of_intersections = 2, then: Set temp_base_line = parallel_line.
h. Return h and M
```

Thus fig. 6(b) shows the height, h and the peak intersection point, M found by the above algorithm.

3) Segmentation of the Bézier curve: The segmentation process of the new proposing approach follows horizontal approach rather than vertical approach followed by the previous approach. In segmentation, the following steps are followed:

- A normal is drawn on the base line from the point M. This normal obviously intersects the base line or extended base line and the intersection point between normal and the base line can be easily found. Let this point is N. Now the normal (MN) is divided into n equidistant points, $(x_1'', y_1''), (x_2'', y_2'') \ldots (x_n'', y_n'')$. The normal and these n equidistant points are shown by fig. 7(a). If $M(x_1''', y_1''')$ and $N(x_2''', y_2''')$ are the two terminal points of the normal then the normal (MN) is divided using the following equation:

$$x_i'' = (m_1 x_1''' + m_2 x_2''')/(m_1 + m_2) \qquad (4)$$

$$y_i'' = (m_1 y_1''' + m_2 y_2''')/(m_1 + m_2) \qquad (5)$$

The greater value of n is taken, the more accurate segmentation and thus more accurate result is found but in that case computational cost gets higher.

- Now n parallel lines $PL_1, PL_2\ldots PL_n$ of the base line are drawn which are going through n equidistant points, $(x_1'', y_1''), (x_2'', y_2''), \ldots, (x_n'', y_n'')$ on the normal (MN). Each of these n parallel lines of the base line intersects the Bézier curve into two points on both sides of the normal (MN). So, there will be 2n intersection points (except initial point, terminating point and M) lying on the Bézier curve for n parallel lines. Fig. 7(b) shows these 2n points, which are $(x_1', y_1'), (x_2', y_2')\ldots(x_{2n}', y_{2n}')$ These 2n points and M are the desired segmentation points.

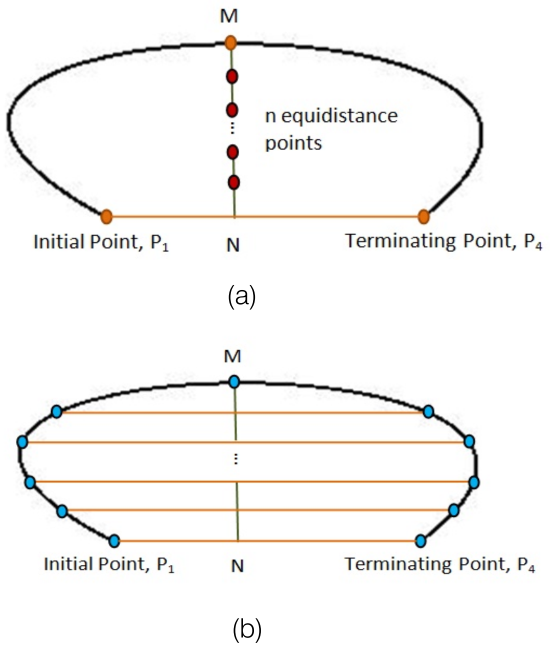


Fig.7 : (a) Normal, equidistant points, (b) Segmentation points of a Bézier curve

4) Assumption of the intermediate control points: This portion of this proposal is same as our previous work. A 3rd order Bézier curve contains two intermediate control points *(P_2 and P_3)*. In this portion these two intermediate control points are assumed. The steps are:

- From the base line, several parallel lines at distance H are drawn. H indicates assumed height with different values - *(3 / 4) h, h, (4 / 3)h* and *1.5h.* Here h is the height of the Bézier curve. Fig. 8 is showing such a parallel line drawn at H = 1.5h distance.
- After each parallel line is drawn, two tangents from initial and terminating points are drawn shown in fig. 8. The tangents must intersect the parallel line at two points. These two points are the assumed intermediate control points. Let one point is P_2', the assumed 2nd control point and other one is P_3', the assumed 3rd control point.
- So, for four values of H (four different distances) - *(3 / 4)h, h, (4 / 3)h* and *1.5h,* four pair of assumed intermediate control points (P_2', P_3') are found. Such a pair of points found at distance 1.5h is shown by fig. 8.

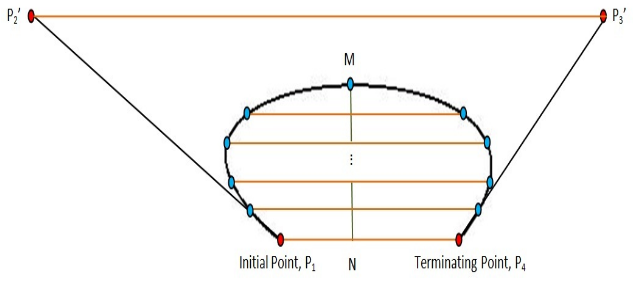


Fig.8 : Parallel line, tangent and assumed intermediate control points of a Bézier curve

From observation it has been found that the original intermediate control points (P_2, P_3) generally lie somewhere around the distances - (3 / 4)h, h, (4 / 3)h and 1.5h. The accurate assumption reduces the number of iterations [6]. It is the reason for drawing parallel lines from the base line at these different values of H. Each distance give us a pair of assumed intermediate control points (P_2', P_3').

b) Second Stage

This stage is same as our previous work [6]. As proposing method focus on minimization of error, so the 2nd and 3rd control point must be in the desired location. In order to find the exact location the approach always calculate the error between the two curves iteratively. If the error becomes minimum, according to algorithm it finds the exact location of 2nd and 3rd control points. The value of the error is the summation of the difference between the points obtained from given curve *Q(t)* and their corresponding points generating from newly found control points in the approximated Bézier curve *Q'(t)* which is shown in fig. 9. Since Bézier curves can be obtained using a parameter t, where $0<=t<=1$, the error between two curves *Q(t)* and *Q'(t)* can be computed by considering the curves in parametric expressions and finding the corresponding points. For *N+1* corresponding points, we are getting locations of corresponding points *Q(t)=(Qx(t),Qy(t))* and *Q'(t)=(Q'x(t),Q'y(t)), (t=0,1/N,2/N,…,1).* The equation that is used for calculating error as follows:

$$Error, E_1 = \sum_{k=0}^{N} |Q(t = k/N) - Q'(t = k/N)|$$

$$= \sum_{k-0}^{N} \sqrt{\{Q_x(k/N) - Q'_x(k/N)\}^2 + \{Q_y(k/N) - Q'_y(k/N)\}^2}$$

The reason why summation is used instead of integration is for computational simplicity. It is applicable when the original Bézier curve is known.

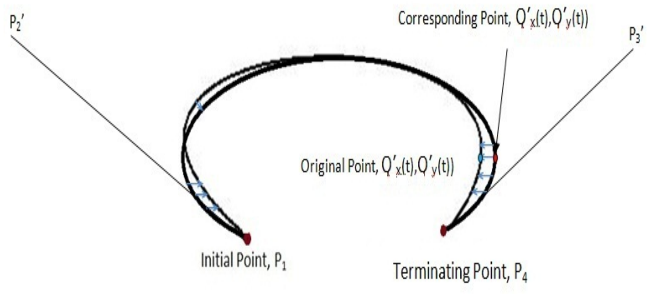

P₂′
Corresponding Point, Q′x(t),Q′y(t))
P₃′
Original Point, Q′x(t),Q′y(t))
Initial Point, P₁
Terminating Point, P₄

Fig.9 : Computation of the sum of the differences between the corresponding points.

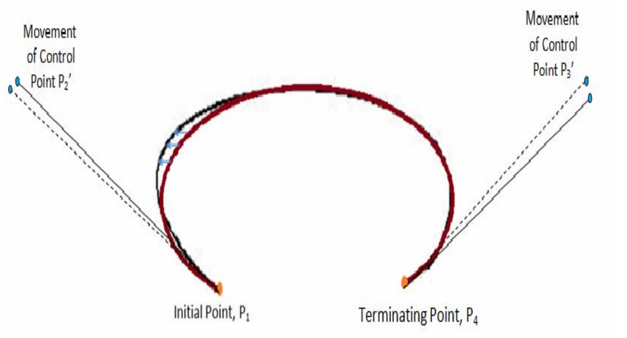


Fig.10 : Movement of the control point to recover error

In order to recover the exact control points the proposing approach try to move those points at a step ∂ towards both X and Y coordinates which shown in fig. 10. The value of the control point will be updated if the calculated error is minimized, otherwise it will not update. If the step fails to minimize error according to algorithm it reduces the step by half and recursively performing the same operation as stated above.

Increasing the efficiency of the overall program was the main contribution as well as overcoming the limitation of the step-size. Although the approach able to get the accurate result for some Bézier curve by considering step size $\partial x=\partial y=1$ [21], but it takes more iterations compared to our propose step size, which reduce the efficiency of the overall performance. Here proposing modification works well for most of the Bézier curve and get the error level less than 0.000001 which increase the dynamicity of overall program because it need less iteration. With variable step $\partial x=\partial y=5$, 50, 75 and 100 in our simulation result we get near to 100% accuracy. The algorithm followed in the second stage is given below.

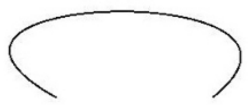
```
[Step 1]
Initialization: allsteps [] = {1, 5, 50, 75,100}
Count=0(Learning time)
MAX=100 (Maximum learning time)
Stepdeterminer = 0 (Variable to determine step)
∂x=∂y=allsteps [stepdeterminer] (Variable displacement for the movement of the control points)
x=assumed_x; y=assumed_y (Initial assumption of control points at 1st stage)
e=0.000001(Minimum permissible error)

[Step 2]
Count++; if (Count>MAX), then goto step 3
if E (x+∂x, y) or E (x-∂x,y ) is minimum then goto step 3
if E (x,y+∂y ) or E (x,y-∂y ) is minimum then goto step 4
if E (x,y) is minimum then ∂x = ∂x /2 , ∂y = ∂y /2, goto step 2

[Step 3]
Searching minimum error for x-direction
While ( E (x+∂x,y) < E (x ,y)) { Count++, x=x+∂x }
While ( E (x-∂x,y) < E (x ,y)) { Count++, x=x-∂x }
if (Count>MAX||E(x,y)<e),then goto Step 5 else goto Step 2

[Step 4]
Searching minimum error for y-direction
While ( E (x,y+∂y) < E (x ,y)) { Count++, y=y+∂y }
While ( E (x,y - ∂y) < E (x ,y)) { Count++, y=y-∂y }
if (Count>MAX||E(x,y)<e),then goto Step 5 else goto Step 2

[Step 5]
If (E(x,y) > e AND stepdeterminer < allsteps.Length) {
Count = 0; stepdeterminer ++; ∂x = ∂y = allsteps [stepdeterminer];
x=assumed_x; y=assumed_y; goto Step 2}
Else { Exit }
```

IV. Simulation Results

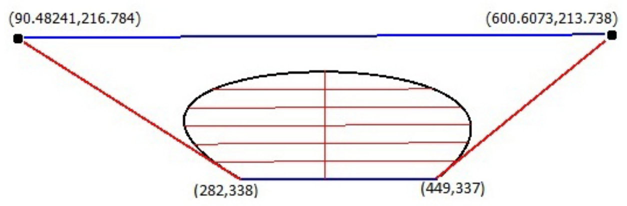
The main contribution of the proposed scheme is to find control points of large number of Bézier curve shapes while keeping the efficiency and accurateness of our existing method. We checked several Bézier curves of different shapes with our computer implemented simulation and got desired results as we expected.

Consider the simulation result shown in fig. 11 and fig. 12. Figures of simulation results are picked directly from our computer implemented simulation. At first stage we segment given curve to know its internal points and assume some control points at some probable height using our new proposed schema. Then using the algorithm of 2nd stage we successfully find the control points. Previous Method [6] was unable to extract control points of those shaped curves because of its limited segmentation technique.

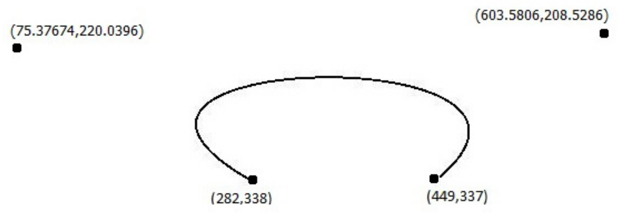
Those shaped curves, whose control points can be found by existing method, can also be found by the newly proposed technique while the accuracy and efficiency of both methods remain same. Example of fig. 13 shows it clearly.



a) Input Image

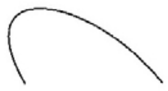


b) Assumption of control points at 1st stage

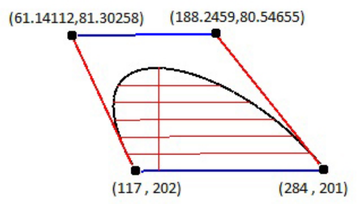


c) Recovered control points

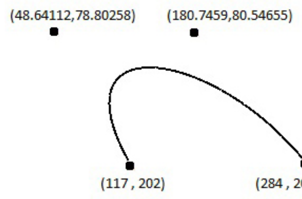
Fig.11 : Simulation Results



a) Input Image

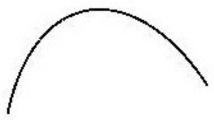


b) Assumption of control points at 1st stage

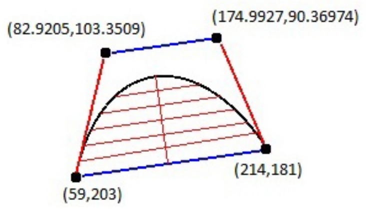


c) Recovered control points

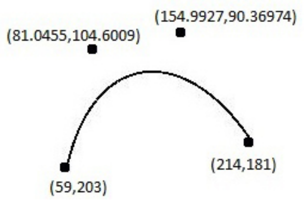
Fig.12 : Simulation Results



a) Input Image



b) Assumption of control points at 1st stage



c) Recovered control points

Fig.13 : Simulation Results

To obtain our desired minimum error we checked our input image of Bézier curve with different combination of H (height) and ∂ (step size). Simulation automatically checks Bézier curve in different combination of H and ∂ until the error < 0.001. Recovered control points for a combination of H and ∂ is chosen as best result if both error and number of iteration are minimum. Table 1 shows recovered control points on each combination with their corresponding error for the example shown in fig. 11.

Table.1 : Calculated error in different combination of height and step-size

Height	Step Size	1st control point(P_1), 2nd control point(P_2), 3rd control point(P_3), 4th control point(P_4)	Iteration	Error
0.75h	5	(282,338),(79,239.6442), (579.5,237.4855), (449,337)	41	0.134
0.75h	50	(282,338),(77.125,216.5192), (606.375,211.2355), (449,337)	26	4E-3
0.75h	75	(282,338),(75.5625,216.9099), (609.5,213.5793), (449,337)	23	4E-3
0.75h	100	(282,338),(77.125,216.5192), (606.375,211.2355), (449,337)	22	4E-3
h	5	(282,338),(78.00368,219.3613), (602.9881,209.3237), (449,337)	40	4E-3
h	50	(282,338),(75.50368,221.8613), (600.4881,206.8237), (449,337)	10	6E-3
h	75	(282,338),(74.72243,225.1816), (598.1443,202.1362), (449,337)	23	14E-3
h	100	(282,338),(75.50368,221.8613), (600.4881,206.8237), (449,337)	11	6E-3
1.33h	5	(282,338),(76.73241,216.784), (609.3573,213.738), (449,337)	10	4E-3
1.33h	50	(282,338), (74.85741,216.784), (606.8573,213.738), (449,337)	8	8E-3
1.33h	75	(282,338),(76.41991,216.784), (607.6385,212.5662), (449,337)	13	4E-3
1.33h	100	(282,338),(74.85741,216.784), (606.8573,213.738), (449,337)	9	8E-3
1.5h	5	(282,338),(73.50174,221.2896), (604.9869,209.2318), (449,337)	16	2E-3
1.5h	50	(282,338),(78.50174,226.2896), (601.2369,204.2318), (449,337)	11	16E-3
1.5h	75	(282,338), (75.37674,220.0396), (603.5806,208.5286), (449,337)	15	2E-3
1.5h	100	(282,338),(78.50174,226.2896), (601.2369,204.2318), (449,337)	12	16E-3

Data from Table 1 shows that minimum error is found at different combination of step-size and height. In this case when height=*1.5h* and step-size= *5*, or height =*1.5h* and step-size=*75* the error is minimum that is *2E-3*. But when height=*1.5h* and step-size=*75*, our simulation takes less iteration. That's why we consider *(282, 338), (75.37674, 220.0396), (603.5806, 208.5286), (449, 337)* as 1st, 2nd, 3rd and 4th control points respectively.

V. Conclusion

The new modified approach can recover the control points of 3rd order Bézier curves more accurately and efficiently than those methods which was proposed earlier. In addition, the emphasis of this paper is on the implementation of a new method that can recover the more different shaped Bézier curve which is not possible according to earlier methods. While conducting the experiment different shapes of 3rd order Bézier curves are taken for simulation and method is proved by the simulation result.

References Références Referencias

1. Computer graphics. [Online]. Available at: http://en.wikipedia.org/wiki/Computer_graphics/
2. MD. Al- Amin Bhuiyan and Hiromitsu Hama, 'Identification of Actors drawn in Ukiyoe picture'. Pattern Recognition, Vol 35, Issue 1, pp. 93-102, 2002.
3. Mohammad Ibrahim Khan and Md. Al-Amin Bhuiyan, 'Facial Expression Recognition for Human-Robot Interface'. IJCSNS International Journal of Computer Science and Network Security, Vol 9, No 4, pp. 300-306, April 2009.
4. G. Farin, 'Curves and surfaces for computer aided geometric design: a practical guide'. Academic Press, 1997.
5. F. Yamaguchi, 'Curves and Surfaces in Computer Aided Geometric Design'. Springer Verlag, 1988.
6. Khan, M.I. Chowdhury, S. Chowdhury, A.K. Deb, K., 'An Efficient Algorithm for Finding the Control Point of 3rd Order Bézier Curve'. International Forum on Strategic Technology, Ulsan. pp. 441-445, 2010.
7. H. Hama and T. Okamoto, 'Fast generation method of Be´zier curves and anti-aliasing'. J. TV Engineers Japan 47(12), 1629–1636, 1993.
8. T. W. Sederberg and R. T. Farouki, 'Approximation by interval Be´zier curves', IEEE Comput. Graphics Appl. 12(5), 87–95, 1992.
9. J. Barry and R. N. Goldman, 'Interpolation and approximation of curves and surfaces using po´lya polynomials'. Graph. Models Image Process. 53(2), 137–148, 1991.
10. E. Catmull, 'A subdivision algorithm for computer display of curved surfaces'. Report UTEC-CSc-74-133, University of Utah, December, 1974.
11. S. Hu, G. Wang, and T. Jin, 'Generalized subdivision of Be´zier surfaces'. Graph. Models Image Process. 58(3), 218–222, 1996.
12. J. Gravesen, 'Adaptive subdivision and the length and energy of Be´zier curves.' Comput. Geometry 8(1), 12–31, 1997.
13. Bogacki, S. E. Weinstein, and X. Ye, 'Degree reduction of Be´zier curves by uniform approximation with endpoint interpolation'. Comput. Aided Design 27(9), 651–661, 1995.
14. S. Lodha and J. Warren, 'Degree reduction of Be´zier simplexes'. Comput. Aided Design 26(10), 735–745, 1994.
15. M. Eck, 'Least squares degree reduction of Be´zier curves'. Comput. Aided Design 27(11), 845–851, 1995.
16. L. Piegl and W. Tiller, 'Algorithm for degree reduction of B-spline curves'. Comput. Aided Design 27(2), 101–110, 1995.

17. L. Ramshaw, 'Blossoming: A connect-the-dots approach to splines'. Digital Systems Research Center Report No. 19, CA, 1987.
18. T. W. Sederberg, J. Zheng, K. Klimaszewski, and T. Dokken, 'Approximate implicitization using monoid curves and surfaces'. Graph. Models Image Process. 61(4), 177–198, 1999.
19. X. Ye, 'Generating Be´zier points for curves and surfaces from boundary information'. Comput. Aided Design 27(12), 875–885, 1995.
20. J. J. Chou and L. A. Piegl, 'Data reduction using cubic rational B-splines'. IEEE Computer Graphics Applications, 12(3), 60–68, 1992.
21. MD. Al- Amin Bhuiyan and Hiromitsu Hama, 'On recovering the control points of Bézier curves for line image indexing'. J. Electron Imaging, Vol 11, Issue 2, pp. 177, 2002.