Modified Method for Drawing Geographic Maps by Using Cubic Spline Interpolation

Dr. Abdul-Mohssen J. Abdul-Hossen
Computer Science Department, University of Technology/Baghdad.
Email: AbdulMoohsen53@yahoo.com
Imad Issa Fadhala
Computer Science Department, University of Technology/Baghdad.
Email: Emadessa135@yahoo.com

Received on:23/4/2015 & Accepted on:8/10/2015

ABSTRACT

This paper presents suggested modified algorithm to draw the geographic maps. Cubic spline interpolation had been used to draw these maps by generating intermediate points between each two points from a set of given points, where these intermediate points represent the control points which they are used to constrain the curvature of cubic spline depending on a value named *divisor value* that represents the distance between any two adjacent points at spline knots, and specified by the user. Intermediate points have a great effect on curve shape, through dividing the interval between two adjacent points into equal subintervals. This makes the results of the weights a_i , b_i and c_i in spline function will be equal to zero and d_i has already been determined to be $d_i = y_i$, according the equations determining the weights,. So we can draw lines and polygons that are necessary in drawing geographic maps.

Keywords: cubic spline interpolation, intermediate points, divisor value.

INTRODUCTION

A basic technique for designing curved shapes in the plane is interpolating splines. The designer inputs a sequence of control points, and the computer fits a smooth curve that goes through these points [1]. Geographic information is classified into two types; spatial data and attributes data. Spatial data that describes the location or place, in information includes all of the natural and artificial elements, such as city limits, roads, buildings, railway lines, river path, and so on. Spatial Data is represented by simple spatial components, such as point, line and area, points represent some of the existing sites in nature like trees, wells, and cities, lines like rivers, roads or street that have one dimension, area or polygon, it consists of several lines connected with each, forming closed shape [2]. One of the most important ways to represent information is geographical maps. Maps can be defined as ways to distinguish represent any geographical parameter on the ground and its relation with other parameters. This representation may be a point, line or area (polygons). The lines are one of the main tools that are used in the field of mapping, because most information in digital maps is in the form of lines [3].

In a two-dimensional vector geographic information system (GIS), spatial features can be represented in the form of points, lines and areas. The lines represent the basic representation in the field of mapping and GIS in fact that most information on paper and in digital maps is in the form of lines. However, the representation of linear features need not only be as a series of straight lines passing through the ordered coordinates, but also as curves, defined by the mathematical functions (e.g., a spline or Bezier curve) Therefore, linear features can be considered to include straight-line

https://doi.org/10.30684/etj.34.2B.4
2412-0758/University of Technology-Iraq, Baghdad, Iraq

and curve-line features [4] . Curves are considered one of the most important tools that can be used to create high-resolution graphics. Graphics appear smooth at fixed resolution when using many small polylines, but they cannot maintain smoothness when change in size image (scaled), at same time requires a large space for storing high-resolution images. Curves have many important properties, stored in an easy way, scaling shape that an drawn by curves to any resolution with preserving smoothness, and most important property is to provide an easy way to represent real-world object [5].

The research focuses on solving some unavoidable problems that simply come from interpolating with smooth functions, which inhibit drawing of lines and polygons when using cubic spline interpolation. In this paper, the geographic maps are drawn by using many of lines and curves during a suggested algorithm. This implementation is considered very important in mapping and geographic information system (GIS).

Background

Cubic spline is the most popular. The main aim is to fit a cubic spline to data points. This includes the formation of a single equation for each given points, and therefore would each segment has its own equation. While maintaining the constraining the curve fit to the data features [6],[7].

A spline function is a function, which is used to produce curves from polynomial segments that achieve the terms of continuity at their meeting [8]. A cubic spline is a spline constructed by use piecewise polynomial of the third degree which passes a set of m given data points. The second derivative of each polynomial is usually set to zero at the endpoints since this provides a boundary condition that completes the system of m-2 equations. This produces a so-called "natural" cubic spline and leads to a simple tridiagonal system which can be solved easily to give the coefficients of the polynomials. However, this choice is not the only one possible, and other boundary conditions can be used instead [9],[10].

We can express the function S_i as.

$$S_i(x) = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \qquad \ldots(1)$$

for i = 1, 2… n-1

Where x distance between x_i and x_{i+1}.

Both first and second derivatives of these n - 1 equation are, they are

$$s_i'(x) = 3a_i(x-x_i)^2 + 2b_i(x-x_i) + c_i \qquad \ldots(2)$$
$$s_i''(x) = 6a_i(x-x_i) + 2b_i$$

for $i = 1, 2, \ldots, n-1$. …(3)

From Eq (3), $s_i''(x) = 6a_i(x-x_i) + 2b_i$, so

$$\begin{aligned} s_i''(x) &= 6a_i(x-x_i) + 2b_i \\ s_i''(x_i) &= 6a_i(x_i-x_i) + 2b_i \\ s_i''(x_i) &= 2b_i \end{aligned} \qquad \ldots(4)$$

for i = 2,3,…n-2

To simplify these equations, will put $s''_i(x_i) = M_i$, and can determine the equation coefficients of Eq (4) in terms of M_i and y_i. And therefore can find b_i value, and d_i determined beforehand $d_i = y_i$. [8].

and d_i identified already to be

$$\begin{aligned} s_i''(x_i) &= 2b_i \\ M_i &= 2b_i \\ b_i &= \frac{M_i}{2} \end{aligned} \qquad \ldots(5)$$

$$d_i = y_i$$

Similarly, using equation *ai* can be re-written as

$$\begin{aligned} 2b_{i+1} &= 6a_i h + 2b_i \\ 6a_i h &= 2b_{i+1} - 2b_i \\ a_i &= \frac{2b_{i+1} - 2b_i}{6h} \\ a_i &= \frac{2(\frac{M_{i+1}}{2}) - 2(\frac{M_i}{2})}{6h} \\ a_i &= \frac{M_{i+1} - M_i}{6h} \end{aligned} \qquad \ldots(6)$$

and *ci* can be re-written as

$$d_{i+1} = a_i h^3 + b_i h^2 + c_i h + d_i$$

$$c_i = \frac{y_{i+1} - y_i}{h} - (\frac{M_{i+1} + 2M_i}{6})h. \qquad \ldots(7)$$

We now have our equations for determining the weights for our n-1 aquations

$$\begin{aligned} a_i &= \frac{M_{i+1} - M_i}{6h} \\ b_i &= \frac{M_i}{2} \\ c_i &= \frac{y_{i+1} - y_i}{h} - (\frac{M_{i+1} + 2M_i}{6})h \end{aligned} \qquad \ldots(8)$$

These systems can be handled more conveniently by putting them into matrix form as follows

$$c_{i+1} = 3a_i h^2 + 2b_i h + c_i$$

For i= 1, 2, …,n-1 $M_i + 4M_{i+1} + M_{i+2} = 6(\frac{y_i - 2y_{i+1} + y_{i+2}}{h^2})$ …(9)

Which leads to the folloing matix equation:-

$$\begin{bmatrix} 4 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 4 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 4 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 4 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 4 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} M_2 \\ M_3 \\ M_4 \\ \vdots \\ M_{n-3} \\ M_{n-2} \\ M_{n-1} \end{bmatrix} = \frac{6}{h^2} \begin{bmatrix} y_1 - 2y_2 + y_3 \\ y_2 - 2y_3 + y_4 \\ y_3 - 2y_4 + y_5 \\ \vdots \\ y_{n-4} - 2y_{n-3} + y_{n-2} \\ y_{n-3} - 2y_{n-2} + y_{n-1} \\ y_{n-2} - 2y_{n-1} + y_n \end{bmatrix} \qquad \ldots(10)$$

Piecewise Cubic Splines

The fitting of a polynomial curve to a set of data points has applications in CAD (computer-assisted design), CAM (computer-assisted manufacturing), and computer graphics systems. An operator wants to draw a smooth curve through data points that are not subject to error. Traditionally, it was common to use a French curve or an architect's spline and subjectively draw a curve that looks smooth when viewed by the eye. Mathematically, it is possible to construct cubic functions $S_i(x)$ on each interval $[x_i , x_{i+1}]$ so that the resulting piecewise curve $y = S(x)$ and its first and second derivatives are all continuous on the larger interval $[x_0, x_N]$. The continuity of $S'(x)$ means that the graph $y = S(x)$ will not have sharp corners. The continuity of $S''(x)$ means that the radius of curvature is defined at each point [11].

Cubic Spline Generation Algorithm

Algorithm (1) *Cubic Spline Interpolation***,** *that draws a cubic spline curve from N data points, where N= 4 or above.*

Input: N data points $p(x_i , y_i)$,where $N \geq 4$.

Output: set of interpolated points (x_i , y_i).

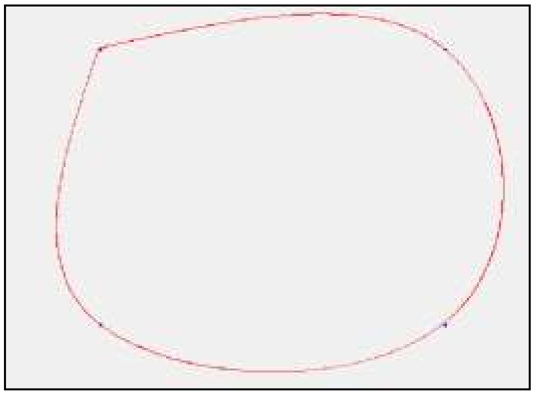
```
Step1
1-Calculate h vector
where      $h[i]=(x_{i+1} - x_i)$
2-Build the matrix A
           for i = 0,.........N-1
   $A[i, i+1] = h[i+1]$

       for i = 0,.........N-2
3-Construct the B vector
   $B[i] = 6 * (((y_{i+2} - y_{i+1}) / h[i+1]) - ((y_{i+1} - y_i) / h[i]))$

        for i = 2......N-1
Step2: Solve the system Eq (10) and store the results in X vector
Step3: Build the M vector
M[j] = X[i, 0]
for i= 2......N-1
Step4: Calculate the coefficients' $a_i$, $b_i$, $c_i$ and $d_i$
          for i=1,...,N-1
Step5: Calculate the results of equation $f(x) = c_n x^n + c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + ..... + c_0 x^0$ shows a form of a polynomial function with a single variable
Step6: Return the points $(x_i, y_i)$
```

Methodology

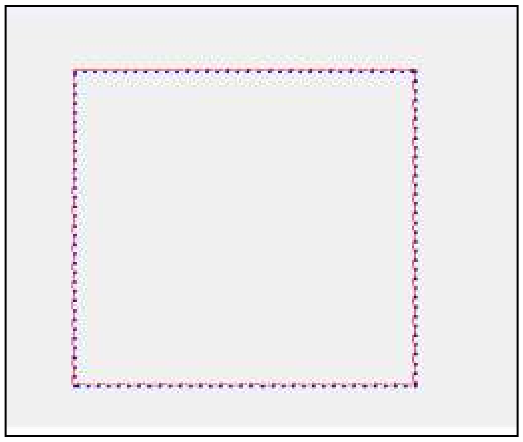
This paper studies how to solve the following problems; First problem with interpolation is that the outcome depends strongly on the parameterization of the interpolation sites. Second, the really nasty side of interpolation by smooth functions becomes apparent when interpolating on lines that enter corners. This problem called overshooting problem.

Our proposed method solves the above problems so we can draw geographic maps. The following example illustrates the difference between natural cubic spline algorithm and suggested algorithm. We will draw polygon(square) by natural cubic spline interpolation, through the following data points entered, p1(300, 300);p2(600, 300); P3(600, 600);p4(300, 600);p5(300, 300);

the result is in Figure (3.1), and then draw same shape and same points but by using the suggested algorithm, as shown in Figure3.2



Figure(1) draw polygons (square) by natural cubic spline interpolation algorithm.



Figure(2) draw polygon (square) by used suggested algorithm

Cubic spline interpolation is stable and smooth characteristics, for that it does not prevent overshooting at intermediate points, which is essential for many drawing geographic maps applications. Therefore, we want controlling the curvature of the spline by generating intermediate points between points that needed into generated points, to be more suitable for drawing letters and numbers. This is achieved by the following steps.

1. Calculating the number of required points that need to be generated between p_i and p_{i+1} points
2. Generating intermediate points between each adjacent pair of data points
3. Merging the generated points with the original data points, and put the result in a new array
4. Performing the steps from 1 to 3, and sent to cubic spline algorithm

Proposed Algorithm to Generate Intermediate Points between Points of Curve

We derive several algorithms for curves represented using normal coordinates.

1. Calculating the number of required points that need to be generated between p_i and p_{i+1} points depending on the divisor value. This achieved by compute the absolute value for difference between the x_i and x_{i+1} divided on divisor value, and the absolute value for difference between y_i and y_{i+1} divided on divisor value, and the large value will be the result that represents the number of required points that need to be generated between p_i and p_{i+1} points.

Algorithm (2) Calculating the number of required points that need to be generated between p_i and p_{i+1} points

```
Input: $p_1$ $(x_1,y_1)$, $p_2(x_2,y_2)$ ,divisor
Output: integer result  (number of intermediates points).
Step1:define an initialize parameters

x. difference = 0, y. difference = 0 , result
where:
    x. difference = different between the $x_i$ and $x_{i+1}$

    y. difference = different between the $y_i$ and $y_{i+1}$
    Result = number of required points that need to be generated between $p_i$ and $p_{i+1}$ points

Step2: calculating the number of required points from the following equations.
  $\text{x.dif} = \left|\frac{\text{p2.x} - \text{p1.x}}{\text{divisor}}\right|$
  $\text{y.dif} = \left|\frac{\text{p2.y} - \text{p1.y}}{\text{divisor}}\right|$

  if (x.dif > y.dif)
      result = x.dif
  OR (x.dif < y.dif)
      result = y.dif
  else
    (x.dif= y.dif)
      result = y.dif
Step3: Return result.
```

2. This algorithm will be as a generator for intermediate points between each adjacent pair of data points. That be detection by algorithm 3.1, the result of this algorithm will be points array, called intermediate points, size it number of intermediate points +2(p_1,p_2). The points (p_1,p_2) that will generate intermediate points between them ,will be the first and last data point, Respectively. And will generate the intermediate points, by some equations. As shown in below.

Algorithm (3) generator for intermediate points between each adjacent pair of data points.

```
Input: $p_1 (x_1,y_1)$, $p_2(x_2,y_2)$ , Nip

Where
        $p_1,p_2$ represent the 2 data points that need to generated

        Nip: number of intermediate points between them.

Output : Nip intermediate points  + $p_1$ and $p_2$ data points

Step1: Define and initialize parameters: intermediate points [Nip+1]
Where
Intermediate points represent the array of points that will be generated.

Step2: add the first point to the first location of the intermediate points array.
Intermediate points[0]=$p_1$

Step3: add the second point to the last location of the intermediate points array.
Intermediate points[Nip +1]=$p_2$
Step4:generate Nip points between $p_1,p_2$
        for i= 1 to  Nip

            step1.1:

                if ($x_2 > x_1$)
                    intermediate points[i].x = intermediate points[i-1].x+ $\left|\frac{X2-X1}{Nip}\right|$
                else if  ( $x_2 < x_1$ or $x_2 = x_1$)
                    intermediate points[i].x=intermediate points[i-1].  x  - $\left|\frac{X2-X1}{Nip}\right|$
            step1.2:comput yi
                if ($y_2 > y_1$)
                    intermediate points[i].y=intermediate points[i-1]. |y + $\frac{y2-y1}{Nip}$
                else if ( $y_2 < y_1$ or $y_2 = y_1$ )
                    intermediate points[i].y=intermediate points[i-1]  y - $\left|\frac{y2-y1}{Nip}\right|$
Next i
Step5: Return intermediates points.
```

3. In this algorithm, the generated points will merge with the original data points, and put the result in a new array that called Newsprints array.

Algorithm (4) merging the generated points with the original data points

```
Input: I points, points, location.
Where:
          I points; represent the array of the generated points.
          Points: represent the array of the original data points.
          Location: integer number represents the location of the points that needed
generation.
Output: New points array.
Step1: Define and initialize parameters.
        Int Nip=I points. length,
             Size= points .length,
             New size=size+(Nip-2),
             New points [New size].
Where:
            Nip: number of the intermediate points.
           Size:  size of the original points array.
           New size: represent the size of the new array that will contain all the points
(New points).
             Track: index for the New points.
             Generated track: index for the I points
.             Original track: index for the points.
Step2:  merging the I points with the points and save the result to the Newpoints.
              for i= 0 to New size
                     If  i = location
            for track = 0 to i+ Nip
                 New points[track] =  I points  [generated track]
                 Generated track ++
Next track
                            i= track-1
                         flag = true
else
             if flag = true New  points[i]=points[original track +2];
                  original track++
      end if
                  If flag = false
      New points [i]=points [original track]
                  original track++
  end if
end else
Step4:Return New points;
```

4. The last algorithm in our methodology it is complete process and draws the any shape by cubic spline curve after modified it depended on the new points and divisor value.

5.

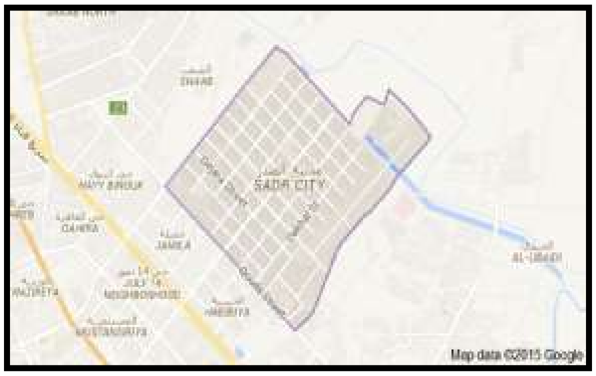
Algorithm (5) completes process and draws the any shape by cubic spline curve after modified it depended on the New points and divisor value.

Input: points , divisor
Output: shapes graphics curves

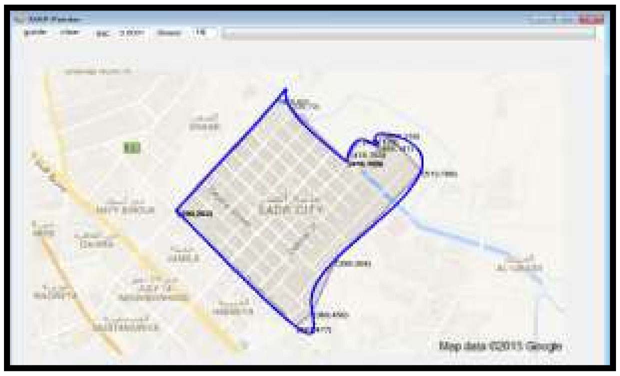
Step1; send the Pointe to the algorithm 1 and receive the result that represent the array of locations that need for generating intermediate points .
Step2: send the points that need generation to the algorithm 2 to retrieve the number of the require point Nip to be generated.
Step3: Send the location and the points to algorithm 3 to retrieve the generated points (intermediate points).
Step 4: Send the original points, generated points and locations to the algorithm 4 to merge all the points into one array and retrieve the result as new points.
Step5: Send the new points to the cubic spline generator and draw the shape.

RESULTS

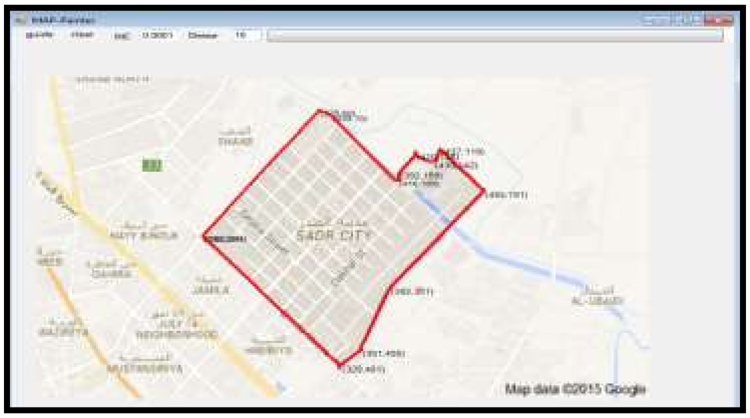
In our proposed method, modified natural cubic spline algorithm is able to draw the lines, by which you can draw a geographical map; this method will be an efficient method at little cost, when compared with the mapping programs that are high-cost programs. The following Figures show the efficiency of the proposed method in drawing geographic maps, especially the shapes that are polygons.



(a)

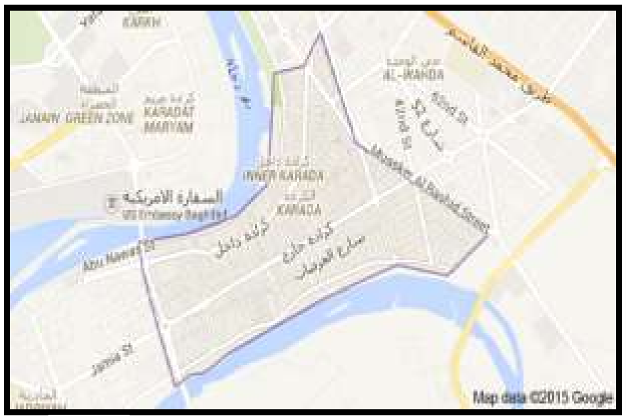


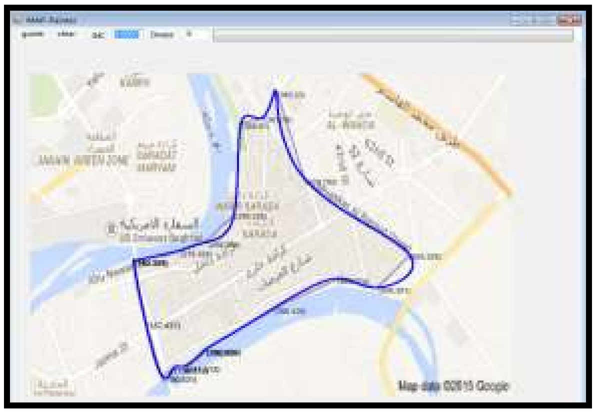
(b)

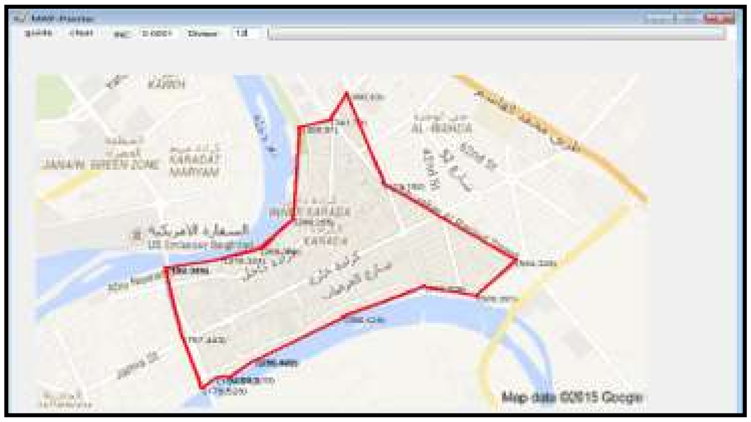


(c)

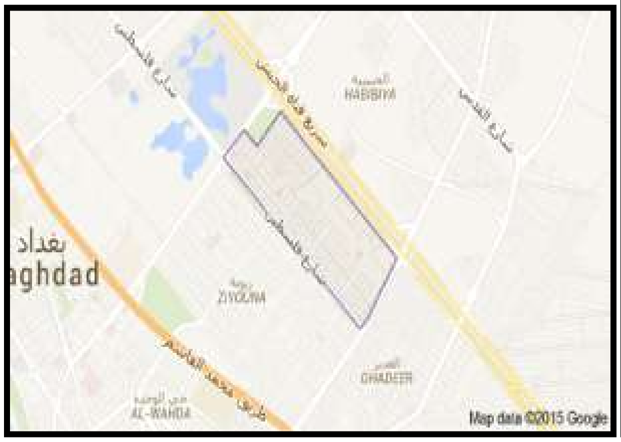
Figure.(3) (a) Original map, (b) by natural cubic spline algorithm, (c) by proposal method


(a)


(b)


(c)

Figure(4). (a) Original map, (b) by natural cubic spline algorithm, (c) by proposal method


(a)

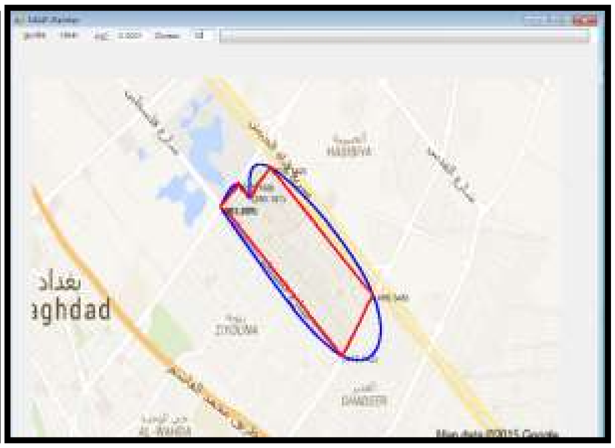

(b)

Figure.(5) (a) Original map, (b) by natural cubic spline algorithm (blue curve), by proposal method (red curve)

Conclusion and future work

The proposed method can be considered easy and low cost method for drawing geographic maps compared with other methods that deal with mapping. This method solves the problems of overshooting and corners that appear with the use of natural cubic spline in an efficient way when used for drawing geographic maps and other fields. Also, the proposed method can be developed to be used in engineering applications and Statistical analysis that needs to be very accurate. This may not be realized by natural spline algorithm because of overshooting problem

REFERENCES:

[1] Levien, R, ” From Spiral to Spline: Optimal Techniques in Interactive Curve Design” Sept. 3, 2009.

[2] John, L” GIS and Decision Making in Local Government ‘Washington, D.C: International City/County Management Association. p. 3-5. 1997. Beyond Maps

[3] Larry, Z,” What is Geographic Information and GIS” Nebraska Guidebook for a Local Government Multipurpose Land Information System. Nebraska Advisory Committee on Standards for Multipurpose Land Information Systems, accessed March 10, 2004.

[4] Tong, X and Shi, W “Measuring positional error of circular curve features in Geographic information systems (GIS)” .2010 Elsevier Ltd. All rights reserved.

[5] Sauer, T” Splines in Industrial Applications” Lecture Notes, Zaragoza, February 2007

[6] O’Neill, C, ” Cubic Spline Interpolation MAE 5093” 28 May 2002.

[7] Rahma, A. M. S. AbdulHossen, A. M. J. and Saeid, A. "Modified BEZIER for Controlling and Generating A Third Degree Curves " Eng. & Tech. Journal, Vol. 28, No. 21, 2010.

[8] Gang, S. Feng, W. Xiuyou, W. Hao, W. and Jing, C. "Application Research on Cubic-Spline Interpolation Based on Particle Swarm Optimization in Mine Pressure Missing Data" IEEE Conference Publication, Vol.1, No.10, 2011.

[9] Hong, S. Wang, L. Truong, T. and Wang, L. "Novel Approaches to the Parametric Cubic Spline Interpolation." IEEE Transactions On Image Processing, Vol. 22, No. 3, March 2013.

[10] Rahma, A. M. S. Abdulla, A. And AbdulHossen, A. M. J. "Modified Method for Generating B-Spline Curves of Degree Three and Their Controlling," *Eng. & Tech. Journal*, Vol. 30, No. 1, 2012.

[11] John, H. M, and Kurtis, K, F” Numerical Methods Using Matlab, 4th Edition, 2004, prentice hall.