Vectorizing Image Outlines using Spline Computing Approaches with Simulated Annealing

MUHAMMAD SARFRAZ
Department of Information Science
Kuwait University
Adailiya Campus, P.O. Box 5969, Safat 13060
KUWAIT
prof.m.sarfraz@gmail.com

Abstract: - This paper is an overview of three spline approaches of degree one, two and three. It represents the review and comparative study of Linear, conic and cubic splines for the vectorization of outlines of the planar images. It has various phases including extracting outlines of images, detecting corner points from the detected outlines, and curve fitting. The idea of simulated annealing has been incorporated to optimize the shape parameters in the description of the conic and cubic splines. In addition, a straightforward approach has also been used for linear spline case because of having no degree of freedom. The methods ultimately produce optimal results for the approximate vectorization of the digital contours obtained from the generic shapes. Demonstrations and a comparative study of linear, conic and cubic splines make the essential parts of the paper.

Keywords: Imaging; optimization; simulated annealing; generic shapes; curve fitting

1 Introduction

Capturing and vectorizing outlines of images is one of the important problems of computer graphics, vision, and imaging. Various mathematical and computational phases are involved in the whole process. This is usually done by computing a curve close to the data point set [3-5, 23-25]. Computationally economical and optimally good solution is an ultimate objective to achieve the vectorized outlines of images for planar objects.

The representation of planar objects in terms of curves has many advantages. For example, scaling, shearing, translation, rotation and clipping operations can be performed without any difficulty. Although a good amount of work has been done in the area [10-16, 35], it is still desired to proceed further to explore more advanced and interactive strategies. Most of the up-to-date research has tackled this kind of problem by curve subdivision or curve segmentation. Curve segmentation is advantageous in a way that it gives a rough geometry of the shape. Approaches used to achieve this task, in the literature, are polygonal approximations [8, 13], circular arc approximations [10, 15, 17, 18, 22] and approximations using cubics or higher order spline functions [2, 14, 24-25].

A non-parametric dominant point detection algorithm was proposed in [8], it used the dominant points for polygonization of digital curves. The problem with polygonal approximation is that these approaches are rarely used for shape analysis. A combination of line segments and circular arcs for object approximation is used in [17, 18]. A scheme to construct a curvature continuous conic spline is proposed in [15]. This approach presented the conic spline curve fitting and fairing algorithm using conic arc scaling. The smoothing is done by removing unwanted curvature extrema. Similar algorithms for data fitting by arc spline curves are presented in [22]. A method for segmentation of curves into line segments and circular arcs by using types of breakpoints is proposed in [10]. Advantage of this technique is that it is threshold free and transformation invariant. Five categories of breakpoints have been defined. The line and conic segmentation and merging is based on these breakpoints.

Least square fitting is mostly adopted in approximations, which uses splines and higher order polynomials. Some approaches are based on active contour models known as snakes. These techniques are also based on parameterization. Enhancement to the scheme by adjusting both number and positions of control points of the active spline curve is shown in [14]. This scheme is based on curve approximation using iterative optimization with B-spline curve by squared distance minimization.

Another way, other than parametric form, is to use implicit form of the polynomial. Curve reconstruction problem is solved by approximating the point clouds using implicit B-spline curve [12]. The authors have used trust region algorithm in optimization theory as minimization heuristics. Techniques described for fitting implicitly defined algebraic spline curves and surfaces to scattered data by simultaneously approximating points and associated normal vectors are proposed in [19, 20, 21].

This work is a presentation of three approaches using linear, conic, and cubic interpolations [42-44]. The linear interpolant approach is straight forward. However, the conic and cubic approaches are inspired by an optimization algorithm based on simulated annealing (SA) by Kirkpatrick et. al. [26]. It motivates the author to an optimization technique proposed for the outline capture of planar images. In this paper, the data point set represents any generic shape whose outline is required to be captured. We present an iterative process to achieve our objective. The algorithm comprises of various phases to achieve the target. First of all, it finds the contour [25-30] of the gray scaled bitmap images. Secondly, it uses the idea of corner points [1-7] to detect corners. That is, it detects the corner points on the digital contour of the generic shape under consideration. These phases are considered as preprocessing steps. Linear, conic, and cubic interpolants are then used to vectorize the outline. The idea of simulated annealing (SA) [26] is used to fit a conic and a cubic spline which pass through the corner points. It globally optimizes the shape parameters in the description of the conic and cubic splines to provide a good approximation to the digital curves. In case of poor approximation, the insertions of intermediate points are made as long as the desired approximation or fit is achieved.

In most of the cases, corner points are not enough to approximate the digital object and hence some more points are also needed. These points are known as break points as they are used to break a segment for better approximation. For onwards discussion, the set of corner points together with the break points will be called as the set of significant points. In the fourth phase of the proposed algorithm, for each iteration, we will insert a point as knot in every piece (if needed) in a manner that the distance, d, of the computed point on the spline curve and its corresponding contour point is greater than a threshold ε. This process increases the set of significant points and hence needs multilevel coordinate search to be employed again for the updated set of significant points to fit an optimal spline curve. This process continues until it rectifies the solution and helps towards the objective optimization in a global fashion. We stop the iterative process when all $d's$ are less than ε. The proposed spline method, using multilevel coordinate search, ultimately produces optimal results for vectorizing the digital contour of the generic shapes. It provides an optimal fit as far as curve fitting is concerned.

The organization of the paper is as follows, Section 2 discusses about pre-processing steps which include finding the boundary of planar objects and detection of corner points. Section 3 is about the interpolant forms of linear, conic, and cubic spline curves. Section 4 briefly introduces about the Simulated Annealing heuristic. Overall methodology of curve fitting is explained in Section 5, it includes the idea of knot insertion as well as the algorithm design for the proposed vectorization schemes. Algorithms for the schemes are devised in Section 6. Demonstration of the schemes as well as comparative study is presented in Section 7. Finally, the paper is concluded in Section 8.

2 Preprocessing

The proposed schemes start with finding the boundary of the generic shape and then using the output to find the corner points. The image of the generic shapes can be acquired either by scanning or by some other mean. The aim of boundary detection is to produce an object's shape in graphical or non-scalar representation. Chain codes [27], in this paper, have been used for this purpose. Demonstration of the method can be seen in Figure 1(b) which is the contour of the bitmap image shown in Figure 1(a).

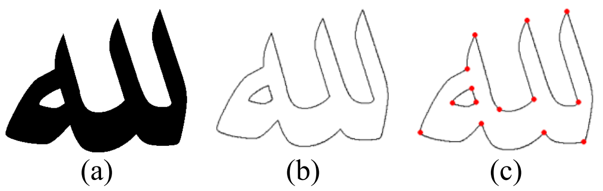

Fig. 1. Pre-processing Steps: (a) Original Image, (b) Outline of the image, (c) Corner points achieved.

Corners, in digital images, give important clues for the shape representation and analysis. These are the points that partition the boundary into various segments. The strategy of getting these points is based on the method proposed in [1]. The demonstration of the algorithm is made on Figure

1(b). The corner points of the image are shown in Figure 1(c).

3 Curve Fitting and Spline

The motive of finding the corner points, in Section 2, was to divide the contours into pieces. Each piece contains the data points in between two subsequent corners inclusive. This means that if there are *m* corner points $cp_1, cp_2, ..., cp_m$ then there will be *m* pieces $pi_1, pi_2, ..., pi_m$. We treat each piece separately and fit the spline to it. In general, the i^{th} piece contains all the data points between cp_i and cp_{i+1} inclusive. After breaking the contour of the image into different pieces, we fit the spline curve to each piece. To construct the parametric spline interpolant on the interval $[t_0, t_n]$, we have $F_i \in R^m$, $i = 0,1,......,n$, as interpolation data, at knots t_i, $i = 0,1,......,n$.

3.1 Linear Spline

The curve fitted by a linear spline is a candidate of best fit, but it may not be a desired fit. This leads to the need of introducing some extra treatment in the methodology. This section deals with a form of linear spline. It introduces parameters *t*'s in the description of linear spline defined as follows:

$$P(t) = P_i(1-\theta) + P_{i+1}\theta \qquad (1)$$

where

$$\theta\big|_{[t_i,t_{i+1})}(t) = \frac{(t-t_i)}{h_i},\ h_i = t_{i+1} - t_i,$$

and P_i and P_{i+1} are corner points of the i^{th} piece.

3.2 Conic Spline

The curve fitted by a conic spline is a candidate of best fit, but it may not be a desired fit. This leads to the need of introducing some shape parameters in the description of the conic spline. This section deals with a form of conic spline. It introduces shape parameters *u*'s in the description of conic spline defined as follows:

$$P(t) = \frac{P_i(1-\theta)^2 + u_i U_i 2\theta(1-\theta) + P_{i+1}\theta^2}{(1-\theta)^2 + 2u_i\theta(1-\theta) + \theta^2}, \qquad (2)$$

where

$$U_i = \frac{(V_i + W_i)}{2},\ V_i = P_i + \frac{h_i D_i}{u_i},\ W_i = P_{i+1} - \frac{h_i D_{i+1}}{u_i}.$$

D_i and D_{i+1} are the corresponding tangents at corner points P_i and P_{i+1} of the i^{th} piece. The tangent vectors are calculated as follows:

$$\left.\begin{aligned} D_0 &= 2(P_1 - P_0) - \frac{(P_2 - P_0)}{2} \\ D_i &= a_i(P_i - P_{i-1}) + (1-a_i)(P_{i+1} - P_i) \\ D_n &= 2(P_n - P_{n-1}) - \frac{(P_n - P_{n-2})}{2} \end{aligned}\right\}, \qquad (3)$$

where

$$a_i = \frac{\|P_{i+1} - P_i\|}{\|P_{i+1} - P_i\| + \|P_i - P_{i-1}\|}.$$

Obviously, the parameters u_i's, when equal to 1, provide the special case of quadratic spline. Otherwise, these parameters can be used to lose or tighten the curve. This paper proposes an evolutionary technique, namely simulated annealing (SA), to optimize these parameters so that the curve fitted is optimal. For the details of SA approach, the reader is referred to [29].

3.3 Generilized Cubic Spline

The cubic spline is defined as follows:

$$P(t) = (1-\theta)^3 F_i + 3\theta(1-\theta)^2 V_i + 3\theta^2(1-\theta)W_i + \theta^3 F_{i+1} \qquad (4)$$

where

$$\theta\big|_{[t_i,t_{i+1})}(t) = \frac{(t-t_i)}{h_i}$$

and

$$V_i = F_i + \frac{h_i D_i}{3},\ W_i = F_{i+1} - \frac{h_i D_{i+1}}{3}$$

Equation (4) can be rewritten as:

$$P\,|_{(t_i,t_{i+1})}(t) = R_{0,i}(t)F_i + R_{1,i}(t)V_i + R_{2,i}(t) + R_{3,i}(t)F_{i+1} \qquad (5)$$

where

$$\left.\begin{aligned} R_{0,i}(t) &= (1-t)^3, \\ R_{1,i}(t) &= 3t(1-t)^2, \\ R_{2,i}(t) &= 3t^2(1-t), \\ R_{3,i}(t) &= t^3, \end{aligned}\right\} \qquad (6)$$

The functions $R_{j,i}, j = 0,1,2,3$ are Bernstein Bézier like basis functions, such that

$$\sum_{j=0}^{3} R_{j,i}(t) = 1 \qquad (7)$$

From the Bernstein-Bézier theory it follows that the curve segment $P|_{[t_i,t_{i+1}]}$ lies in the convex hull of the control points $\{F_i, V_i, W_i, F_{i+1}\}$ and is variation diminishing with respect to the control polygon joining these points.

To get the control points $\{F_i, V_i, W_i, F_{i+1}\}$, we make use of a Bernstein-Bézier representation where we can impose the Hermite interpolation conditions:

$$P(t_i) = F_i \text{ and } P'(t_i) = D_i,\ i = 0,1,......,n \qquad (8)$$

where F_i and F_{i+1} are corner points of i^{th} piece. D_i and D_{i+1} are the corresponding tangents at corner points.

To construct the parametric C^1 cubic spline interpolant on the interval $[t_0, t_n]$ we have $F_i \in R^m$, $i = 0,1,......,n$, as interpolation data, at knots t_i, $i = 0,1,......,n$. The derivatives $D_i \in R^m$ can be found out by the imposition of C^1 constraints on the piecewise cubic form. The C^1 constraints can be written as:

$$P'(t_i^+) = P'(t_i^-) .$$

The tangent vectors are calculated as in Eqn. (3).

The cubic curve fitted is a candidate of best fit, but it may not be a desired fit. This leads to the need of introducing some shape parameters in the description of the cubic spline. Thus, one needs to deal with a more generalized form of cubic spline. Let us introduce two parameters v and w in the description of cubic spline defined as follows:

$$P(t) = (1-\theta)^3 F_i + 3\theta(1-\theta)^2 V_i + 3\theta^2(1-\theta)W_i + \theta^3 F_{i+1} \qquad (9)$$

where

$$V_i = F_i + h_i v_i D_i,\ W_i = F_{i+1} - h_i w_i D_{i+1} \qquad (10)$$

Obviously, the parameters v_i's and w_i's, when equal to 1/3, provide the special case of cubic spline in Eqn. (5). Otherwise, these parameters can be used to lose or tight the curve. This paper proposes an evolutionary technique, namely simulated annealing (SA), to optimize these parameters so that the curve fitted is optimal.

4 Simulated Annealing (SA)

Simulated annealing (SA) [26] is a global optimization method based on the Monte Carlo method. It works on the analogy of the energy in an n-body system where the material is cooled to lower temperatures gradually to result in a perfect crystal structure. The perfect crystal structure is attained by having minimum energy in the material. This analogy translates to the optimization done in simulated annealing in finding a solution that has the lowest objective function value. The solution space is all the possible solutions. The current solution is the present state of the material. The algorithm iteratively tries to change the state of the material and check whether it has improved. The material's state is changed slightly to find a neighboring state i.e. a close solution value in the solution space. It is possible that all neighboring states of current states are worse solutions. The algorithm allows going to a worse state with a certain probability. This probability decreases as the algorithm iterations proceed. Finally, it only allows a change in state if it is strictly better than the current solution. Details of SA theory can be found in [25, 29]. A detailed description of the mapping of the SA technique on the proposed problem is given in the next section.

5 Proposed Approach for Vectorization

The proposed approach to the curve problem is described here in detail. It includes the phases of problem matching with SA using conic and cubic splines, description of parameters used for SA, curve fitting, and the overall designs of algorithms.

5.1 Problem Mapping

This section describes about the SA formulation of the current problem in detail. Our interest is to optimize the values of cubic curve parameters v and w (parameters u in the case of conic curve) such that the defined curve fits as close to the original contour segments as possible. We use SA for the optimization of these parameters for the fitted curves. Hence the dimensionality of the solution space is 2 for cubic curves and 1 for conic curves. Each state in the SA solution space represents a pair of values for v and w for cubics (and value of u for conics).

We start with an initial state that is a given pair of v and w values for cubics (and value of u for conics). A starting temperature is also chosen arbitrarily. This temperature is an inherent internal parameter of SA and has no significance or mapping on our problem. The algorithm maintains a record of the best state ever reached throughout the algorithm run. This is the pair of v and w values for cubics (and value of u for conics) that has given the best curve fitting so far. This best solution gets updated whenever the algorithm finds a better solution. The algorithm iteratively looks for neighboring states that may or may not be better than the current one. These neighboring states are v and w values for cubics (and value of u for conics) that are slightly different from the current pair of v and w values for cubics (and value of u for conics). The cooling rate in SA is the factor affecting the likelihood of selecting a neighboring pair of v and w values for cubics (and value of u for conics) that gives a curve fitting worse than the current pair of v and w values for cubics (and value of u for conics).

Note that we apply SA independently for each segment of a contour that we have identified using corner points. SA is applied sequentially on each of the segments, generating an optimized fitted curve for each segment. The algorithm is run until the maximum allowed time is reached, or an optimal curve fitting is attained.

5.1.1 Initialization

Once we have the bitmap image of a generic shape, the boundary of the image can be extracted using the method described in Section 2. After the boundary points of the image are found, the next step is to detect corner points as explained in Section 2. This corner detection technique assigns a measure of 'corner strength' to each of the points on the boundary of the image. This step helps to divide the boundary of the image into n segments. Each of these segments is then approximated by interpolating splines described in Sections 3.2 and 3.3. The initial solution of spline parameters v and w for cubics (and u for conics) are randomly selected within the range [-1, 1].

5.1.2 Curve Fitting

After an initial approximation for the segment is obtained, better approximations are obtained through SA to reach the optimal solution. We experiment with our system by approximating each segment of the boundary using the conic splines of Section 3.2 and generalized cubic splines of Section 3.3.

The conic spline method is a variation of the quadratic spline. It provides greater control on the shape of the curve and also efficient to compute. The tangents, in the description of the spline, are computed using the arithmetic mean method described in Eqn. (3). Each boundary segment is approximated by the spline. The shape parameter u, in the conic spline, provides greater flexibility over the shape of the curve. These parameters are adjusted using SA to get the optimal fit.

The generalized cubic spline method, explained in Sectionn3.3, is a variation of the well-known Hermite cubic spline. This modified Hermite cubic spline provides greater control on the shape of the curve and is also efficient to compute. The tangents, in the description of the spline, are computed using the arithmetic mean method described in Eqn. (3). Each boundary segment is approximated by the spline. The shape parameters. v and w in the cubic spline provide greater flexibility over the shape of the curve. These parameters are adjusted using SA to get the optimal fit.

Since, the objective of the paper is to come up with optimal techniques which can provide decent curve fit to the digital data. Therefore, the interest would be to compute the curve in such a way that the sum square error of the computed curve with the actual curve (digitized contour) is minimized. Mathematically, the sum squared distance is given by:

$$S_i = \sum_{j=1}^{m_i} \left[P_i(u_{i,j}) - P_{i,j} \right]^2,\ t_{i,j} \in \left[t_i, t_{i+1} \right],\ i = 0,1,\ldots,n-1 \quad , \qquad (11)$$

where

$$P_{i,j} = (x_{i,j}, y_{i,j}), \quad j = 1,2,\ldots,m_i\ , \qquad (12)$$

are the data points of the ith segment on the digitized contour. The parameterization over t's is in accordance with the chord length parameterization. Thus the curve fitted in this way will be a candidate of best fit.

Once an initial fit for a particular segment is obtained, the parameters of the fitted curve v's and w's for cubics (u's for conics) are adjusted to get better fit. Here, we try to minimize the sum squared error of Eqn. (11). Using SA, we try to obtain the optimal values of the curve parameters. We choose this technique because it is powerful, yet simple to implement and as shown in Section 6, performs well for our purpose.

5.1.3 Segmentation using Intermediate Points

For some segments, the best fit obtained through iterative improvement may not be satisfactory. In that case, we subdivide the segment into smaller segments at points where the distance between the boundary and parametric curve exceeds some predefined threshold. Such points are termed as *intermediate points*.

6 The Algorithms

We can summarize all the phases from digitization to optimization discussed in the previous sections. The algorithms of the proposed schemes are contained on various steps. Algorithm 1 of Section 6.1 explains the mechanism for the computation of linear curve. Curve manipulation methodology with conics and cubics, using SA, has been laid down in Algorithm 2 of Section 6.2.

6.1 Algorithm 1 for Linear Interpolant

The summary of the algorithm, designed for optimal curve design using linear interpolant, is as follows:

Step AG1.1: Input the image.
Step AG1.2: Extract the contours from the image in Step AG1.1.
Step AG1.3: Compute the corner points from the contour points in Step AG1.2 using the method in Section 2.
Step AG1.4: Fit the linear spline curve method of Section 3.1 to the corner points achieved in Step AG1.3.
Step AG1.5: IF the curve, achieved in Step AG1.4, is optimal then GO To Step AG1.8, ELSE locate the appropriate intermediate points (points with highest deviation) in the undesired curve pieces.
Step AG1.6: Enhance and order the list of corner and intermediate points achieved in Step AG1.3 and AG1.5.
Step AG1.7: GO TO Step AG1.4.
Step AG1.8: STOP.

6.2 Algorithm 2 for Conic (Cubic) Interpolant

The summary of the algorithm, designed for optimal curve design using conic or cubic interpolants, is as follows:

Step AG2.1: Input the image.
Step AG2.2: Extract the contours from the image in Step AG2.1.
Step AG2.3: Compute the corner points from the contour points in Step AG2.2 using the method in Section 3.2 for conics (Section 3.3 for cubics).
Step AG2.4: Compute the derivative values at the corner / intermediate points.
Step AG2.5: Compute the best optimal values of the shape parameters u_i's for conics (v_i's and w_i's for cubics) using SA.
Step AG2.6: Fit the spline curve method of Section 3.2 for conics (of Section 3.3 for Cubics) to the corner / intermediate points achieved in Step AG2.2.
Step AG2.7: IF the curve, achieved in Step AG2.6, is optimal then GO To Step AG2.10, ELSE locate the appropriate intermediate points (points with highest deviation) in the undesired curve pieces.
Step AG2.8: Enhance and order the list of the corner / intermediate points achieved in Step AG2.3 and AG2.7.
Step AG2.9: GO TO Step AG2.4.
Step AG2.10: STOP.

6.3 SA Parameters Used

SA requires an initial guess for the solution. It is this starting state parameter that affects the performance of the algorithm. If the starting solution is very near the optimal solution, it is more likely to find the optimal solution readily than if the starting solution is distant from the optimal solution. Another important parameter for SA is the temperature. This parameter can be started arbitrarily at any value since the algorithm decreases it gradually until it reaches its minimum value. The rate at which the temperature decreases during the running of the algorithm is determined by another parameter: the cooling rate. This is a constant parameter initialized at the beginning, and is used to update the temperature after a certain interval of time.

The maximum time allowed for the algorithm is a static parameter set in the beginning. Since SA does not terminate unless it exhausts its allotted time, care needs to be given while setting this parameter. If it is too high, then even if the algorithm reaches an optimal solution, it will not return unless the maximum time allowed is reached. There is another internal constant that determines how long it takes before the temperature is updated. This duration is also variable and is adjusted with the updation variable. Table 1 shows the SA parameter settings that we have used for our curve fitting optimization problem.

Table 1. Parameter Settings for SA.

SA parameters	**Values**
Range of input parameters *v* and *w* for cubic (and *u* for conic)	[-1,1]
Fitness Function Optimization Target	0 (function minimization)
Dimension of problem (number of inputs to SA)	2 for cubic (1 for conic)
Weight given to global search vs local search	20
Maximum number of iterations (epochs)	200
Stopping relative error (if distance from optima is less than this, the algorithm terminates)	1e-4
Initial Set of Solution Points	Corner-points and Mid-points of solution space
Number of Steps in Local Search	50

The above mentioned schemes and the algorithms have been implemented and tested for various images. Reasonably quite elegant results have been observed as can be seen in the following Section of demonstrations.

7 Demonstration

The proposed curve scheme has been implemented successfully in this section. We evaluate the performance of the system by fitting parametric curves to different binary images.

Figure 2 shows the implementation results of the two algorithms for the image “Plane” in Figure 1(a). Figures 2(a) and 2(b) are the results for the linear scheme, respectively, without and with insertion of intermediate points. Similarly, Figures 2(c) and 2(d) are the results for the conic scheme, respectively, without and with insertion of intermediate points. Cubic Curve Fitting without Intermediate Points and with Intermediate Points are shown in Figures 2(e) and 2(f) respectively.

Figures 3 shows the implementation results of an image of an Arabic Language word “Lillah (لله)”. Figures 3(a) and 3(b) are respectively its outline and outline together with the corner points detected. Figures 3(c) and 3(d) are the results for the linear scheme, respectively, without and with insertion of intermediate points. Similarly, Figures 3(e) and 3(f) are the results for the conic scheme, respectively, without and with insertion of intermediate points. Cubic Curve Fitting without Intermediate Points and with Intermediate Points are shown in Figures 3(g) and (h) respectively.

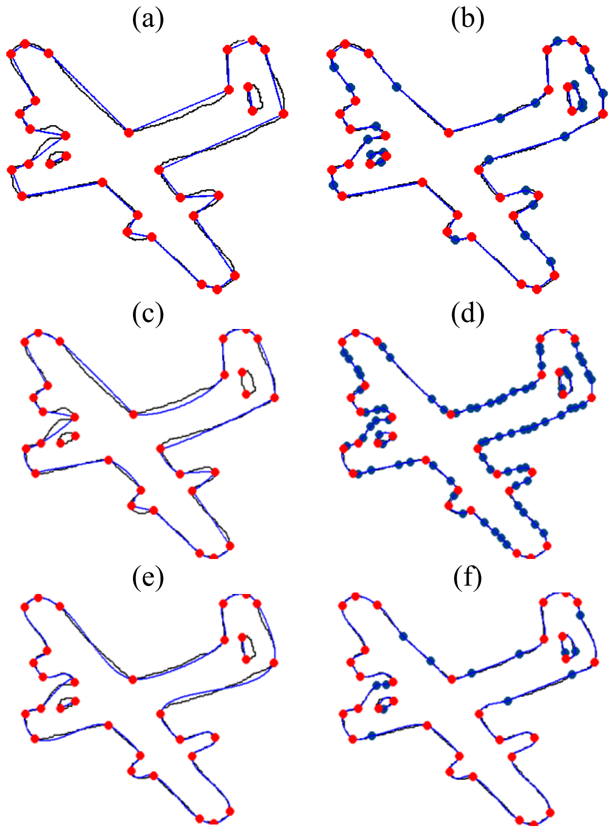

Fig. 2. Curve Fittings on an extracted outline of a plane (a) Linear Curve Fitting Without Intermediate Points (b) Linear Curve Fitting With Intermediate Points (c) Conic Curve Fitting Without Intermediate Points (d) Conic Curve Fitting With Intermediate Points (e) Cubic Curve Fitting Without Intermediate Points (f) Cubic Curve Fitting With Intermediate Points.

Table 2. Test images, contours and corner point details of outlines.

Image	**Name**	**# of Contours**	**# of Contour Points**	**# of Initial Corner Points**
	Lillah	2	[1522+161]	14
	Plane	3	[1106+61+83]	31

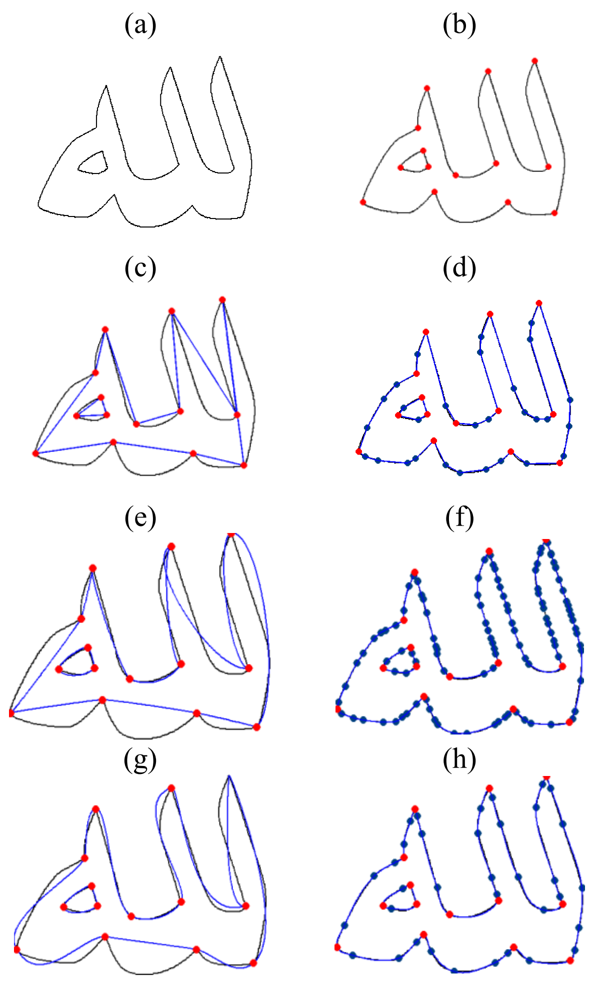

Fig. 3. Curve Fittings on an extracted outline of arabic word ‘Lillah’ font image (a) Extracted Outline (b) Initial Corner Points (c) Linear Curve Fitting Without Intermediate Points (d) Linear Curve Fitting With Intermediate Points (e) Conic Curve Fitting Without Intermediate Points (f) Conic Curve Fitting With Intermediate Points (g) Cubic Curve Fitting Without Intermediate Points (h) Cubic Curve Fitting With Intermediate Points

One can see that the approximation is not satisfactory when it is achieved over the corner points only. This is specifically due to those segments which are bigger in size and highly curvy in nature. Thus, some more treatment is required for such outlines. This is the reason that the idea to insert some intermediate points is demonstrated in the algorithms. It provides excellent results. The idea of how to insert intermediate points is not explained here due to limitation of space. It will be explained in a subsequent paper.

Table 3. Comparison of number of initial corner points, intermediate points and total time taken (in seconds) for conic interpolation approaches.

Image	**# of Intermediate Points in Linear Interpolation**	**Total Time Taken For Linear Interpolation**	
		Without Intermediate Points	**With Intermediate Points**
Lillah.bmp	29	2.827	3.734
Plane.bmp	24	7.735	8.39

Table 4. Comparison of number of initial corner points, intermediate points and total time taken (in seconds) for conic interpolation approaches.

Image	**# of Intermediate Points in Conic Interpolation**	**Total Time Taken For Conic Interpolation**	
		Without Intermediate Points	**With Intermediate Points**
Lillah.bmp	14	19.485	48.438
Plane.bmp	31	20.953	49.356

Table 5. Comparison of number of initial corner points, intermediate points and total time taken (in seconds) for cubic interpolation approaches.

Image	**# of Intermediate Points in Cubic Interpolation**	**Total Time Taken For Cubic Interpolation**	
		Without Intermediate Points	**With Intermediate Points**
Lillah.bmp	35	308.479	749.511
Plane.bmp	13	179.786	286.033

Tables 2, 3, 4 and 5 summarize the experimental results for different bitmap images. These results highlight various information including contour details of images, corner points, intermediate points, total time taken for linear, conic interpolation approaches. Table 2 demonstrates test images, their contour and corner point details of outlines. Tables

3, 4 and 5 exhibit comparisons of number of initial corner points, intermediate points and total time taken (in seconds) for linear, conic, and cubic interpolation approaches respectively. One can have the following observations here:

- Some of the outlines of images need more intermediate points than the others to have satisfactory fit. It happens when initial (default) outline fit is too loose and away from its actual outline. Or, it happens when initial (default) outline fit is much bigger in size from its actual outline.
- Increase in intermediate points is also proportionally related to the higher accuracy of the approximation of outline fit.
- Achievement of higher accuracy of the approximation of outline fit is also proportionally related to high amount of time consumed for each of the linear, conic, and cubic approaches.

Table 6. Comparison of objective function values for linear, conic and cubic interpolation approaches.

Image	Objective Function Values for:					
	Linear Interpolation		Conic Interpolation		Cubic Interpolation	
	Without Intermedia	With Intermedia	Without Intermedia	With Intermedia	Without Intermedia	With Intermedia
Lillah.bmp	22340.4223	1240.7778	28192.1557	1654.4514	18087.7440	1384.4751
Plane.bmp	3302.2542	1137.1815	4153.5356	1250.6142	2911.4841	1205.1647

Table 7. Comparison of number of function calls taken by SA for conic and cubic interpolation approaches with and without intermediate points.

Image	# of Function Calls taken by SA for:			
	Conic Interpolation Without Intermediate Points	Conic Interpolation With Intermediate Points	Cubic Interpolation Without Intermediate Points	Cubic Interpolation With Intermediate Points
Lillah.bmp	10611	165864	22703	53062
Plane.bmp	27392	150035	38211	53108

Tables 6 and 7 summarize some further experimental results. These results explain comparison of objective function values (Table 6) and number of iterations (Table 7). It is convenient to see, in Table 6, that higher objective function values take place when the schemes are implemented without inserting the intermediate points. This also indicates that the accuracy of approximation is not high without inserting the intermediate points. Hence, the higher accuracy of the outline capture is proportionally related to lower value of objective functions in each of the linear, conic, and cubic approaches.

Table 7 reflects that the higher number of function calls take place when each of the approach is implemented with inserting the intermediate points. This also indicates that when the accuracy of approximation is high, the number of function calls is also high. Hence, the higher accuracy of the outline capture is proportionally related to higher number of function calls in each of the linear, conic, and cubic approaches.

One can observe from the Tables 2-7 that linear spline fit is computationally most economical as compared to its conic and cubic spline techniques. This is mainly due to the fact that SA methodology is part of the conic and cubic approximations, but not the linear approximation. However, it is worth noting that conic and cubic approximations are comparatively smoother than their linear interpolation counterpart. It is also mentionable that cubic approach, with or without intermediate points, provides more accurate approximation as compared to its conic counterpart.

8 Conclusion

Two optimization techniques are proposed for the outline capture of planar images. First technique uses simply a linear interpolant and a straight forward method based on distribution of corner and intermediate points. Second technique uses the simulated annealing to optimize a conic spline to the digital outline of planar images. By starting a search from certain good points (initially detected corner points), an improved convergence result is obtained. The overall technique has various phases including extracting outlines of images, detecting corner points from the detected outline, curve fitting, and addition of extra knot points if needed. The idea of simulated annealing has been used to optimize the shape parameters in the description of a conic spline introduced. The methods ultimately produce optimal results for the approximate vectorization of the digital contours obtained from the generic shapes. The schemes provide optimal fits with efficient computation cost as far as curve fitting is concerned. The proposed algorithms are fully automatic and require no human intervention. The two proposed approaches of linear and conic splines have been compared to the cubic spline which also uses the idea of simulated annealing to optimize the shape parameters in the description of a cubic spline. A detailed comparative study and analysis have been developed for the three approaches. As a future study and continuation of the problem, the author is also thinking to apply the proposed methodologies for surface models in 3D. This work is in progress to be published as a subsequent work.

Acknowledgement
The author is thankful for the helpful comments of Referees. This work was supported by Kuwait University, Research Grant No. [WI 05/12].

References:

[1] D. Chetrikov, S. Zsabo, A Simple and Efficient Algorithm for Detection of High Curvature Points in Planar Curves, The Proceedings of the 23rd Workshop of the Australian Pattern Recognition Group, 1999, pp. 1751-2184.

[2] M. Sarfraz, Representing Shapes by Fitting Data using an Evolutionary Approach, International Journal of Computer-Aided Design & Applications, Vol. 1(1-4), 2004, pp 179-186.

[3] A. Goshtasby, Grouping and Parameterizing Irregularly Spaced Points for Curve Fitting, ACM Transactions on Graphics, 2000, pp. 185-203.

[4] M. Sarfraz, M. A. Khan, An Automatic Algorithm for Approximating Boundary of Bitmap Characters, Future Generation Computer Systems, 2004, pp. 1327-1336.

[5] M. Sarfraz , Some Algorithms for Curve Design and Automatic Outline Capturing of Images, International Journal of Image and Graphics, 2004, pp. 301-324.

[6] Z. J. Hou, G.W.Wei, A New Approach to Edge Detection, Pattern Recognition, 2002, pp. 1559-1570.

[7] P. Reche, C. Urdiales, A. Bandera, C. Trazegnies, F. Sandoval, Corner Detection by Means of Contour Local Vectors, Electronic Letters, Vol. 38, No. 14, 2002.

[8] M. Marji, P. Siv, A New Algorithm for Dominant Points Detection and Polygonization of Digital Curves, Pattern Recognition, 2003, pp. 2239-2251.

[9] M. Sarfraz, Designing Objects with a Spline, International Journal of Computer Mathematics, Taylor & Francis, Vol. 85, No. 7, 2008.

[10] Wu-Chih Hu, Multiprimitive Segmentation Based on Meaningful Breakpoints for Fitting Digital Planar Curves with Line Segments and Conic Arcs, Image and Vision Computing, 2005, pp. 783-789.

[11] H. Kano, H. Nakata, C. F. Martin, Optimal Curve Fitting and Smoothing using Normalized Uniform B-Splines: A Tool for Studying Complex Systems, Applied Mathematics and Computation, 2005, pp. 96-128.

[12] Z. Yang, J. Deng, F. Chen, Fitting Unorganized Point Clouds with Active Implicit B-Spline Curves, Visual Computer, 2005, pp. 831-839.

[13] G. Lavoue, F. Dupont, A. Baskurt, A New Subdivision Based Approach for Piecewise Smooth Approximation of 3D Polygonal Curves, Pattern Recognition, 2005, pp. 1139-1151.

[14] H. Yang, W. Wang, J. Sun, Control Point Adjustment for B-Spline Curve Approximation, Computer Aided Design, 2004, pp. 639-652.

[15] X. Yang, Curve Fitting and Fairing using Conic Spines, Computer Aided Design, 2004, pp. 461-472.

[16] M. Sarfraz, Computer-Aided Reverse Engineering using Simulated Evolution on NURBS, International Journal of Virtual & Physical Prototyping, Vol. 1, No. 4, 2006, pp. 243-257.

[17] J. H. Horng, An Adaptive Smooting Approach for Fitting Digital Planar Curves with Line Segments and Circular Arcs, Pattern Recognition Letters, 2003, pp. 565-577.

[18] B. Sarkar, L. K. Singh,, D. Sarkar, Approximation of Digital Curves with Line Segments and Circular Arcs using Genetic Algorithms, Pattern Recognition Letters, 2003, pp. 2585-2595.

[19] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, T. R.

Evans, Reconstruction and Representation of 3D Objects with Radial Basis Functions, Proceedings of SIGGRAPH 01, 6776, 2001.

[20] B. Juttler, A. Felis, A Least Square Fitting of Algebraic Spline Surfaces, Advance Computer Mathematics, 2002, pp. 135-152.

[21] B. S. Morse, T. S. Yoo, D. T. Chen, P. Rheingans, K. R. Subramanian, Interpolating Implicit Surfaces from Scattered Surface Data using Compactly Supported Radial Basis Functions, SMI 01 Proceedings of the International Conference on Shape Modeling and Applications, 8998, IEEE Computer Society, Washington DC, 2001.

[22] X. N. Yang, G. Z. Wang, Planar Point Set Fairing and Fitting by Arc Splines, Computer Aided Design, 2001, pp. 35-43.

[23] M. Sarfraz, M. Riyazuddin, M. H. Baig, Capturing Planar Shapes by Approximating their Outlines, International Journal of Computational and Applied Mathematics, Vol. 189, No. 1-2, 2006, pp. 494 – 512.

[24] M. Sarfraz, A. Rasheed, A Randomized Knot Insertion Algorithm for Outline Capture of Planar Images using Cubic Spline, The Proceedings of The 22th ACM Symposium on Applied Computing (ACM SAC-07), Seoul, Korea, 2007, pp. 71 – 75, ACM Press.

[25] M. Sarfraz, Outline Capture of Images by Multilevel Coordinate Search on Cubic Splines, Lecture Notes in Artificial Intelligence: Advances in Artificial Intelligence, A. Nicholson, X. Li (Eds.): Vol. 5866, Springer-Verlag Berlin Heidelberg, 2009, pp. 636–645.

[26] S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi, Optimization by Simulated Annealing, Science, Vol. 220(4598), 1983, pp. 671-680.

[27] H. Freeman, L.S. Davis, A corner finding algorithm for chain-coded curves, IEEE Trans. Computers, Vol. 26, 1977, pp. 297-303.

[28] M. Sonka, V. Hlavac, R. Boyle, Image processing, analysis, and machine vision. Brooks/Cole publication, 2001, pp. 142-143.

[29] N. Richard, T. Gilbert, Extraction of Dominant Points by estimation of the contour fluctuations, Pattern Recognition, Vol. (35), 2002, pp. 1447-1462.

[30] W. Huyer, A. Neumaier, Global Optimization by Multilevel Coordinate Search, Journal of Global Optimization, Vol. 14, 1999, pp. 331-355.

[31] Y. Kumar, S. K. Srivastava, A. K. Bajpai, N. Kumar, Development of CAD Algorithms for Bezier Curves/Surfaces Independent of Operating System, WSEAS Transactions on Computers, Vol. 11, No. 6, 2012, pp. 159-169.

[32] K. Thanushkodi, K. Deeba, On Performance Analysis of Hybrid Intelligent Algorithms (Improved PSO with SA and Improved PSO with AIS) with GA, PSO for Multiprocessor Job Scheduling, WSEAS Transactions on Computers, Vol. 11, No. 5, 2012, pp. 159-169.

[33] M. Sarfraz, N. Al-Dabbous, "Curve Representation for Outlines of Planar Images using Multilevel Coordinate Search", WSEAS Transactions on Computers, Vol. 12(3), WSEAS, 2013, pp. 62–73.

[34] M. Sarfraz, M. Z. Hussain, M. Irshad, Reverse Engineering of the Digital Curve Outlines using Genetic Algorithm, International Journal of Computers, Vol. 7(1), NAUN, 2013, pp. 1–10.

[35] M. Sarfraz, "Generating Outlines of Generic Shapes by mining Feature Points", Recent Advances in Computer Engineering Series, Vol. (16), Segio Lopes. (Eds.), 978-960-474-336-0, WSEAS, 2013, pp. 183 – 190.

[36] J. Kennedy, R. Eberhart, Particle swarm optimization, Proc. IEEE Intl. Conf. Neural Networks, 4, Nov/Dec 1995, pp. 1942 –1948.

[37] R. Eberhart and J. Kennedy, A new optimizer using particle swarm theory, Proc. the Sixth Intl. Symposium on Micro Machine and Human Science, MHS '95, 4-6 Oct 1995, pp. 39 -43.

[38] Y. Shi, R. Eberhart, A modified particle swarm optimizer, The 1998 IEEE Intl. Conf. on Evolutionary Computation Proc., IEEE World Congress on Computational Intelligence, 4-9 May 1998, pp. 69 – 73.

[39] Sarfraz, M., Swati, Z.N.K., (2013), Mining Corner Points on the Generic Shapes, Open Journal of Applied Sciences, Vol. 3(1B), pp. 10 – 15.

[40] Neri F. (2008). "PIRR: a Methodology for Distributed Network Management in Mobile Networks". WSEAS Transactions on Information Science and Applications, WSEAS Press (Wisconsin, USA), issue 3, vol. 5, pp. 306-311.

[41] Bojkovic, Z., Neri, F. (2013) An introduction to the special issue on advances on interactive multimedia systems WSEAS Transactions on Systems, 12 (7), pp. 337-338.

[42] M. Sarfraz, Vectorizing Outlines of Generic Shapes by Cubic Spline using Simulated Annealing, International Journal of Computer Mathematics, Taylor & Francis, Vol. 87(8), 2010, pp. 1736 – 1751.

[43] M. Sarfraz, Intelligent Approaches for Vectorizing Image Outlines, International Journal of Software Engineering and Applications, Vol. 5(12B), 2012, pp. 78 – 83.