

# Two Approaches for Vectorizing Image Outlines

Muhammad Sarfraz, *Member, IACSIT*

**Abstract**—Two approaches, based on linear and conic splines, are proposed here for vectorizing image outlines. Both of the approaches have various phases including extracting outlines of images, detecting corner points from the detected outlines, and curve fitting. Interpolation splines are the bases of the two approaches. Linear spline approach is straight forward as it does not have a degree of freedom in terms of some shape controller in its description. However, the idea of the soft computing approach, namely simulated annealing, has been utilized for conic splines. This idea has been incorporated to optimize the shape parameters in the description of the generalized conic spline. Both of the linear and conic approaches ultimately produce optimal results for the approximate vectorization of the digital contours obtained from the generic shapes. Demonstrations and a comparative study also make the essential parts of the paper.

**Index Terms**—Vectorization, corner points, generic shapes, curve fitting.

## I. INTRODUCTION

Vectorizing outlines of images is required in various real life problems including font design, cartooning, pattern recognition, etc. It 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. Computationally economical and optimally good solution is an ultimate objective to achieve the vectorized outlines of images for planar objects.

Designing and modeling some appropriate curve scheme [1]-[3] is one of the important phases of capturing and vectorizing outlines of images. It plays a significant role in various applications. 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 [4]-[16], 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.

The proposed work is the extension of the work in [17]. This work is a presentation of two approaches using linear and conic interpolations. The linear interpolant approach is straight forward. However, the conic approach is inspired by an optimization algorithm based on simulated annealing (SA) by Kirkpatrick et.al [18]. 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 [19] – [22] of the gray scaled bitmap images. Secondly, it uses the idea of corner points [23] – [29] 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 and conic interpolants are then used to vectorize the outline. The idea of simulated annealing (SA) is used to fit a conic spline which passes through the corner points. It globally optimizes the shape parameters in the description of the conic 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.

The organization of the paper is as follows, Section 2 discusses about preprocessing steps which include finding the boundary of planar objects and detection of corner points. Section 3 is about the interpolant forms of linear and conic spline curves. Overall methodology of curve fitting is explained in Section 4, it includes the idea of knot insertion as well as the algorithm design for the proposed vectorization schemes. Demonstration of the proposed schemes as well as comparative study is presented in Section 5. Finally, the paper is concluded in Section 6.

## II. 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 [21], in this paper, have been used for this purpose. Demonstration of the method can be seen in Fig. 1(b) which is the contour of the bitmap image shown in Fig. 1(a).

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 [23]. The demonstration of the algorithm is made on Fig. 1(b). The corner points of the image are shown in Fig. 1(c).

## III. 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  $cp_1, cp_2, \dots, cp_m$  then there will be  $m$  pieces  $pi_1, pi_2, \dots, pi_m$ . We treat each piece separately and fit the spline to it. In general, the  $i^{th}$  piece

Manuscript received March 10, 2012; revised April 26, 2012. This work was supported by Kuwait University, Research Grant No. [WI 05/10].

M. Sarfraz is with the Department of Information Science, Kuwait University, Adailiya Campus, Kuwait (e-mail: muhamad.sarfraz@ku.edu.ke; prof.m.sarfraz@gmail.com).

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  $F_i \in R^m$ ,  $i = 0, 1, \dots, n$ , as interpolation data, at knots  $t_i$ ,  $i = 0, 1, \dots, n$ .

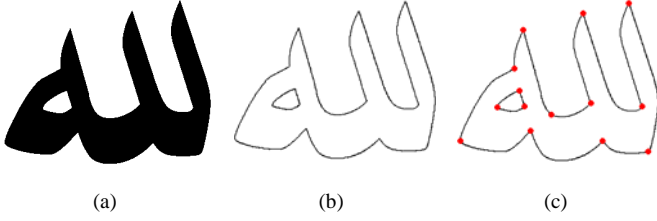


Fig. 1. Pre-processing steps: (a) Original image, (b) Outline of the image, (c) Corner points achieved.

#### A. 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:

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

where

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

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

#### B. Final Stage

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:

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

where

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

$D_i$  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:

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

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 loose 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 [30].

#### IV. SIMULATED ANNEALING (SA)

Simulated annealing (SA) [19], [30] 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 [19], [30] and an example applet can be seen in [17]. A detailed description of the mapping of the SA technique on the proposed problem is given in the next section.

#### V. PROPOSED APPROACH

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

##### A. Problem Mapping

This section describes about the SA formulation of the current problem in detail. Our interest is to optimize the values of conic curve parameters  $u$  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 1 for conic curves. Each state in the SA solution space represents a value of  $u$  for conics.

We start with an initial state that is a given 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 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 values of  $u$  for conics that are slightly different from the values of  $u$  for conics. The cooling rate in SA is the factor affecting the likelihood of selecting neighboring values of  $u$  for conics that give a curve fitting worse than the values 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.

### B. 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 spline described in Sections 3.2. The initial solution of spline parameters  $u$  values for conics are randomly selected within the range  $[-1, 1]$ .

### C. 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.

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.

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:

$$S_i = \sum_{j=1}^{m_i} [P_i(u_{i,j}) - P_{i,j}]^2, t_{i,j} \in [t_i, t_{i+1}], i = 0, 1, \dots, n-1, \quad (4)$$

where

$$P_{i,j} = (x_{i,j}, y_{i,j}), \quad j = 1, 2, \dots, m_i, \quad (5)$$

are the data points of the  $i$ th 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  $u$ 's for conics are adjusted to get better fit. Here, we try to minimize the sum squared error of Eqn. (4). 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.

### D. 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*. A new parametric curve is fitted for each new segment.

## VI. OVERALL APPROACHES AND 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, using SA, has been laid down in Algorithm 2 of Section 6.2. A more detailed description for conics, describing the whole system with step by step flow, is shown in the flowchart demonstrated in Fig. 2.

### A. 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.

### B. Algorithm 2 for Conic Interpolant

The summary of the algorithm, designed for optimal curve design using conic interpolant, 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.

**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 using SA.

**Step AG2.6:** Fit the spline curve method of Section 3.2 for conics 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.

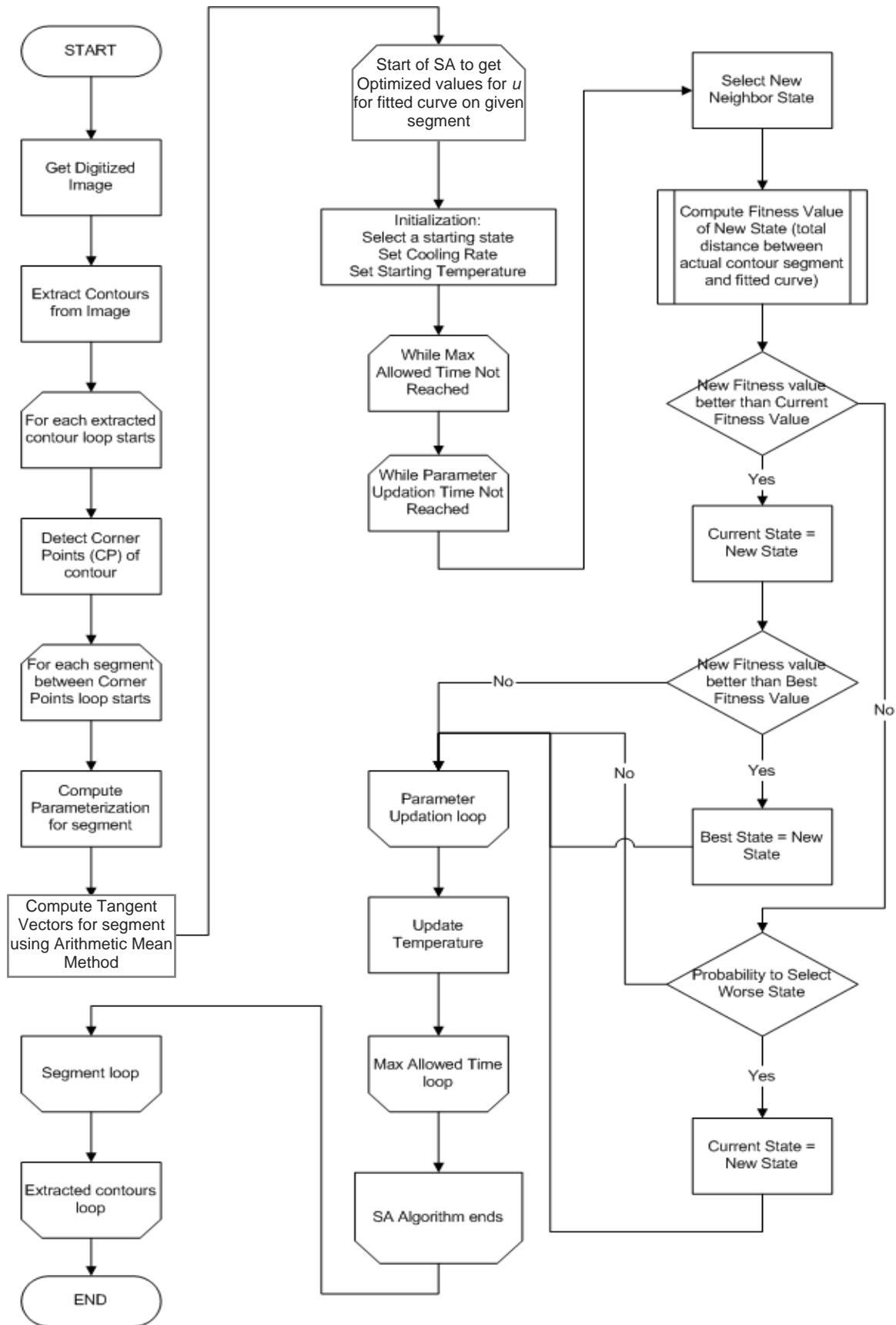


Fig. 2. Flowchart of the system computation of conic curves using SA.

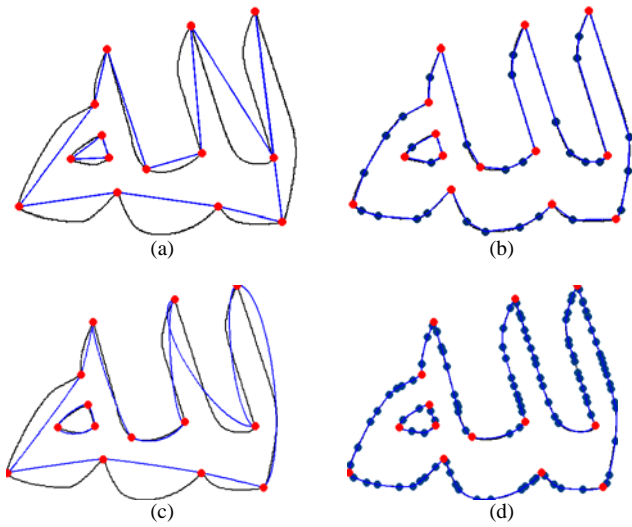


Fig. 3. Results of linear scheme: (a) Fitted outline of the image, (b) Fitted outline of the image with intermediate points. Results of conic scheme: (c) Fitted outline of the image, (d) Fitted outline of the image with intermediate points.

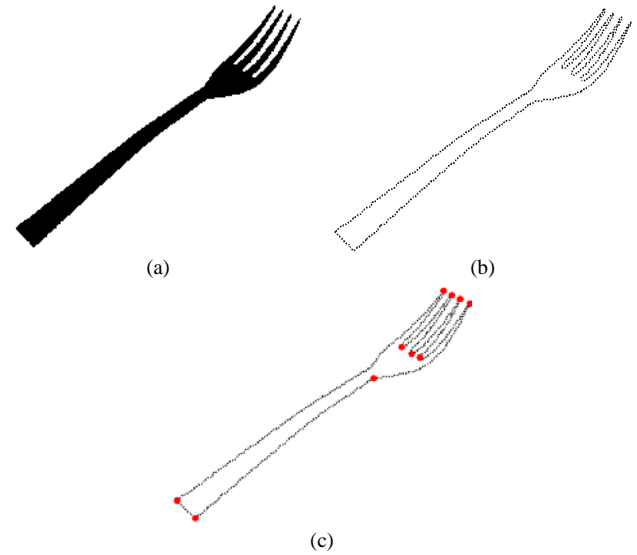


Fig. 6. Pre-processing steps for curve fitting (a) Image of a plane, (b) Extracted outline (c) Initial corner points.

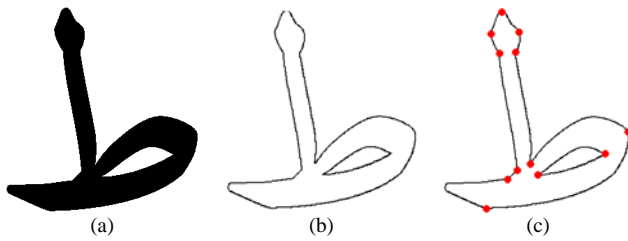


Fig. 4. Pre-processing Steps: (a) Original Image, (b) Outline of the image, (c) Corner points achieved, (d) Fitted Outline of the image.

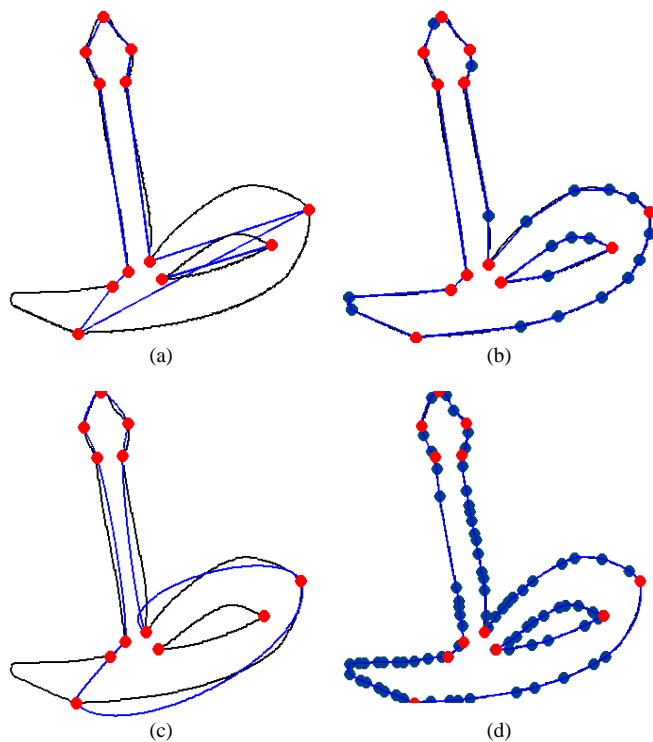


Fig. 5. Results of curve fitting, linear curve fitting: (a) Fitted outline of the image, (b) Fitted outline of the image with intermediate points. Conic curve fitting: (c) Fitted outline of the image, (d) Fitted outline of the image with intermediate points

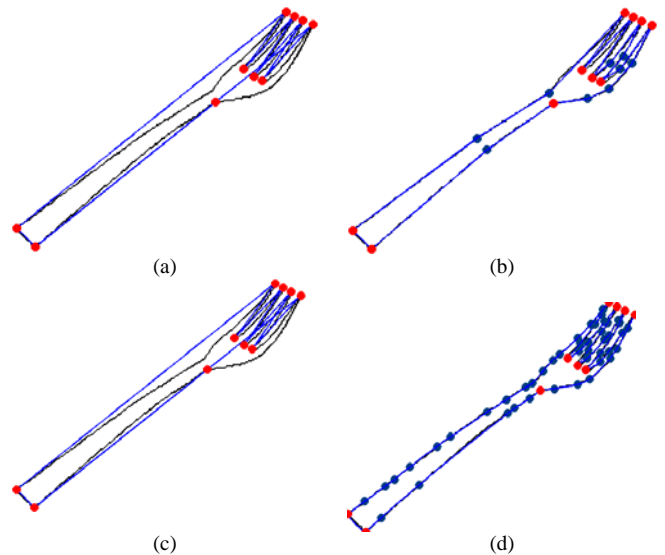


Fig. 7. Results of linear scheme : (a) Fitted outline of the image, (b) Fitted outline of the image with intermediate points. Results of conic scheme: (c) Fitted outline of the image, (d) Fitted outline of the image with intermediate points.

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.

## VII. DEMONSTRATIONS

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.

Fig. 3 shows the implementation results of the two algorithms for the image "Lillah" of Fig. 1(a). Figs. 3(a) and 3(b) are the results for the linear scheme, respectively, without and with insertion of intermediate points. Similarly, Figs. 3(c) and 3(d) are the results for the conic scheme, respectively, without and with insertion of intermediate

points.

Figs. 4 and 5 show the implementation results of a “Tua” image. Figs. 4(a), 4(b), 4(c) are respectively the original image of the Plane, its outline, outline together with the corner points detected. Fig. 5 shows the implementation results of the two algorithms for the “Tua” image in Fig. 4(a). Figs. 5(a) and 5(b) are the results for the linear scheme, respectively, without and with insertion of intermediate points. Similarly, Figs. 5(c) and 5(d) are the results for the conic scheme, respectively, without and with insertion of intermediate points.

Figs. 6 and 7 show the implementation results of a “Fork” image. Figs. 6(a), 6(b), 6(c) are, respectively, the original image of the Fork, its outline, outline together with the corner points detected. Fig. 7 shows the implementation results of the two algorithms for the image in Fig. 6(a). Figs. 7(a) and 7(b) are the results for the linear scheme, respectively, without and with insertion of intermediate points. Similarly, Figs. 7(c) and 7(d) are the results for the conic scheme, respectively, without and with insertion of 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 I: NAMES AND CONTOUR DETAILS OF IMAGES




Image	Name	# of Contours	# of Contour Points	# of Initial Corner Points
	Lillah	2	[1522+161]	14
	Tua	2	[1045+285]	12
	Fork	1	[693]	10

TABLE II: 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
Tua.bmp	19	0.5053	0.8511
Fork.bmp	9	0.2265	0.4029

Tables I, II and III 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

interpolation, and total time taken for conic interpolation.

TABLE III: 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
Tua.bmp	80	67.066	503.657
Fork.bmp	45	32.350	221.885

## VIII. 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 two methods ultimately produce optimal results for the approximate vectorization of the digital contours obtained from the generic shapes. The schemes provide an optimal fit with an efficient computation cost as far as curve fitting is concerned. The proposed algorithms are fully automatic and require no human intervention. The author is also thinking to apply the proposed methodology for another model curve namely cubic. It might improve the approximation process. This work is in progress to be published as a subsequent work.

## ACKNOWLEDGMENT

M. Sarfraz thanks Kuwait University for supporting this work through Project# [WI 05/10].

## REFERENCES

- [1] X. Yang, “Curve Fitting and Fairing using Conic Spines,” *Computer Aided Design*, vol. 6, no. 5, pp. 461-472, 2004.
- [2] X. N. Yang and G. Z. Wang, “Planar Point Set Fairing and Fitting by Arc Splines,” *Computer Aided Design*, vol. 33(1), pp. 35-43, 2001.
- [3] M. Sarfraz, “Designing Objects with a Spline,” *Computer Mathematics*, vol. 87, no. 4, pp. 797-817, 2010.
- [4] M. Sarfraz, “Representing Shapes by Fitting Data using an Evolutionary Approach,” *Computer-Aided Design & Applications*, vol. 1, no. 1-4, pp. 179-186, 2004.
- [5] M. Sarfraz and M. A. Khan, “An Automatic Algorithm for Approximating Boundary of Bitmap Characters,” *Future Generation Computer Systems*, vol. 20, no. 8, pp. 1327-1336, 2004.
- [6] M. Sarfraz, “Some Algorithms for Curve Design and Automatic Outline Capturing of Images,” *Image and Graphics*, vol. 4, no. 2, pp. 301-324, 2004.
- [7] Z. J. Hou and G. W. Wei, “A New Approach to Edge Detection, Pattern Recognition,” vol. 35, no. 7, pp. 1559-1570, 2002.

- [8] M. Sarfraz, "Computer-Aided Reverse Engineering using Simulated Evolution on NURBS," *Virtual & Physical Prototyping*, vol. 1, no. 4, pp. 243 – 257, 2006.
- [9] M. Sarfraz, M. Riyazuddin, and M. H. Baig, "Capturing Planar Shapes by Approximating their Outlines," *Computational and Applied Mathematics*, vol. 189, no. 1-2, pp. 494 – 512, 2006.
- [10] M. Sarfraz, A. Rasheed, "A Randomized Knot Insertion Algorithm for Outline Capture of Planar Images using Cubic Spline," in *Proc. 22<sup>nd</sup> ACM Symposium on Applied Computing (ACM SAC-07)*, Seoul, Korea, pp. 71 – 75, 2007.
- [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*, vol.169, no.1, pp. 96-128, 2005.
- [12] Z. Yang, J. Deng, F. Chen, "Fitting Unorganized Point Clouds with Active Implicit B-Spline Curves," *Visual Computer*, vol. 21, no. 8-10, pp. 831-839 (2005).
- [13] G. Lavoue, F. Dupont, A. Baskurt, "A New Subdivision Based Approach for Piecewise Smooth Approximation of 3D Polygonal Curves, *Pattern Recognition*," vol. 38, pp. 1139-1151, 2005.
- [14] H. Yang, W. Wang, and J. Sun, "Control Point Adjustment for B-Spline Curve Approximation," *Computer Aided Design*, vol. 36, pp. 639-652, 2004.
- [15] J. H. Horng, "An Adaptive Smoothing Approach for Fitting Digital Planar Curves with Line Segments and Circular Arcs," *Pattern Recognition Letters*, vol. 24, no. 1-3, pp. 565-577, 2003.
- [16] B. Sarkar, L. K. Singh, and D. Sarkar, "Approximation of Digital Curves with Line Segments and Circular Arcs using Genetic Algorithms," *Pattern Recognition Letters*, vol. 24, pp. 2585-2595, 2003.
- [17] M. Sarfraz, "Vectorizing Image Outlines using Spline Computing Approach," in *Proc. 4th International Conf. Machine Learning and Computing (ICMLC 2012)*, Hong Kong, China, March 10 – 12, 2012, ASME.
- [18] S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi, "Optimization by Simulated Annealing", *Science*, Vol. 220, no. 4598, pp. 671-680, 1983.
- [19] M. Sarfraz, and M. I. Sarfraz, "Capturing Image Outlines using Spline Computing Approach," in *Proc. 5th International Conf. Signal-Image Technology & Internet-Based Systems (SITIS-2009)*, November 30 – December 03, 2009, Marrakech, Morocco, pp. 126 - 132, 2009, IEEE Computer Society Press.
- [20] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, "Digital Image Processing Using MATLAB," 2<sup>nd</sup> Ed., Gatesmark Publishing, 2009.
- [21] M. S. Nixon, A. S. Aguado, "Feature extraction and image processing, Elsevier, 2008.
- [22] M. Sarfraz, "Outline Capture of Images by Multilevel Coordinate Search on Cubic Splines," *Lecture Notes in Artificial Intelligence*, A. Nicholson and X. Li (Eds.): LNAI 5866, Springer-Verlag Berlin Heidelberg, pp. 636–645, 2009.
- [23] D. Chetrikov and S. Zsabo, "A Simple and Efficient Algorithm for Detection of High Curvature Points in Planar Curves," in *Proc. 23<sup>rd</sup> Workshop of the Australian Pattern Recognition Group*, pp. 1751-2184, 1999.
- [24] A. A. Goshtasby "Grouping and Parameterizing Irregularly Spaced Points for Curve Fitting," *ACM Transactions on Graphics*, vol. 19, no. 3, pp. 185-203, 2000.
- [25] P. Reche, C. Urdiales, A. Bandera, C. Trazegnies, and F. Sandoval, "Corner Detection by Means of Contour Local Vectors," *Electronic Letters*, vol. 38, no. 14, 2002.
- [26] M. Marji and P. Siv, "A New Algorithm for Dominant Points Detection and Polygonization of Digital Curves," *Pattern Recognition*, vol. 36, no. 10, pp. 2239-2251, 2003.
- [27] Wu-Chih Hu, "Multiprimitive Segmentation Based on Meaningful Breakpoints for Fitting Digital Planar Curves with Line Segments and Conic Arcs," *Image and Vision Computing*, vol. 23, no. 9, pp. 783-789, 2005.
- [28] H. Freeman, L. S. Davis, "A corner finding algorithm for chain-coded curves," *IEEE Trans. Computers*, Vol. 26, pp. 297-303, 1977.
- [29] N. Richardand T. Gilbert, "Extraction of Dominant Points by estimation of the contour fluctuations," *Pattern Recognition*, vol. 35, pp. 1447-1462, 2002.
- [30] M. Sarfraz, "Vectorizing Outlines of Generic Shapes by Cubic Spline using Simulated Annealing," *Computer Mathematics*, Vol. 87, no. 8, pp. 1736 – 1751, 2010.



**Muhammad Sarfraz** is a Professor in Kuwait University, Kuwait. He received his Ph. D. from Brunel University, UK, in 1990. His research interests, in general, are Computer Graphics, Pattern Recognition, Computer Vision, Image Processing, and Soft Computing. He is currently working on various projects related to academia and industry.

Prof. Sarfraz has advised/supervised more than 50 students for their MSc and PhD theses. He is the Head of Research Committee in Information Science Department, Kuwait University. He has published more than 200 publications in the form of various Books, Book Chapters, journal papers and conference papers.

Prof. Sarfraz is member of various professional societies including IEEE, ACM, IVS, IACSIT, Pattern Recognition Society, and ISOSS. He is a Chair, member of the International Advisory Committees and Organizing Committees of various international conferences, Symposia and Workshops. He is the reviewer, for many international Journals, Conferences, meetings, and workshops around the world. He is the Editor-in-Chief of three international journals. He is also Editor/Guest Editor of various International Conference Proceedings, Books, and Journals. He has achieved various awards in education, research, and administrative services.