Curve Reconstruction In Different Cubic Functions Using Differential Evolution

Zabidi Abu Hasan1,*, Zainor Ridzuan Yahya1, Nur Afifah Rusdi1 and Nurshazneem Roslan1
1Institute of Engineering Mathematics, University Malaysia Perlis, UniMAP, 02600 Arau, Perlis Malaysia

Abstract. This paper discusses the comparison on using two types of curves for curve reconstruction. Differential Evolution (DE) is used to optimize the parameter in the related spline function. DE minimized the Sum Square Error (SSE) to find the best curve that fit the data. The two curves namely cubic Bézier and cubic Ball is used for comparison purposes. For the curve reconstruction, the cubic Ball consumes less calculation time compare to cubic Bézier and gives better curve approximation based on the errors result. Visualization and numerical comparison shall be given.

1 Introduction

Curve reconstruction is a process of reconstructing a curve or a mathematical function, which has the best fit to a set of data points, possibly subject to certain constraints. It becomes one of the main issue highlighted by recent researchers especially in image processing as well as computer graphics pattern recognition and computer aided geometric design (CAGD) in solving reverse engineering problems [1]. According to [2], the purpose of this curve fitting is to find the set of data points that can accurately represent the given curve. Variety splines have been discussed in the literatures to represent the curve like cubic Bézier curve [1], [2], [3], [4] quartic Bézier curve [5] and B-splines [6], [7]. Other than that, [8] used ball cubic curve to represent the curve.

Soft computing is a recent keyword in information technologies. Soft computing based optimization methods has been used as an attempt to find the best solution in solving various types of problem especially in complex engineering problems. This is mainly due to inefficiency of classical optimization algorithms in solving complex problems. These methods do not guarantee optimal solution but require a relatively short computational time. There are several techniques in soft computing that grabs the attention of researchers such as Genetic Algorithm (GA) [7], [9], Differential Evolution (DE) [5], [10], [11], [8] and Artificial Bee Colony algorithm (ABC) [12], [13], [14].

In this study, the comparison of two different curves which are cubic Bézier and cubic Ball with Differential Evolution has been proposed in order to find the best curve in solving curve fitting problem. Differential Evolution (DE) has been chosen as a propose technique in solving curve fitting problem because this algorithm offers great convergence characteristics and require few control parameters. According to [15], the use of mutation operation, crossover operation and greedy selection makes DE a fast converging evolutionary algorithm. Not only that, the minimum number of control parameters of DE makes it easy to be tuned in order to find best solution [16] and [17]. Besides that, DE able to find the acceptable solution with reasonable time demand, but efficiency of the search is sensitive to the setting of control parameters [18].

2 Curve

2.1. Bézier

As been mention in the introduction, Bézier curve has been proposed by many researchers in solving curve fitting problems because of their being computationally efficient and having high level of accuracy to approximate character shapes [19], [20]. Bézier curve is defined as a parametric curve $R(t)$ that use the Bernstein polynomials as a basis. In general, the equation of Bézier curve is given by:

$$R(t) = \sum_{i=0}^{n} B^n_i(t) P_i$$  \hspace{1cm} (1)

where $B^n_i(t)$ is a basis function for Bézier curve and $P_i$ refers to the control points of the curve. The basis function of the curve can be defined as the following:

$$B^n_i(t) = \frac{n!}{(n-i)!i!} (1-t)^{n-i} t^i$$  \hspace{1cm} (2)

The curve can be represented in any degree $n$ with
For this study, cubic Bézier curve with has been chosen as a proposed curve. Note that, for cubic Bézier curve, it consists of four control points which are $P_0$, $P_1$, $P_2$, and $P_3$. This curve interpolates the two end control points $P_0$ and $P_3$ and approximate the two intermediate points $P_1$ and $P_2$. Figure 1 shows the example of Cubic Bézier curve of degree three.

$$B_i(t) = (1-t)^2, B_i(t) = 2(1-t)t, B_i(t) = 2(1-t)^2, B_i(t) = t^2$$ (3)

where $0 \leq t \leq 1$.

The equation for cubic Ball function similar to cubic Bézier in terms of the number of points needed. However, this function gives a great potential when recently a few research demonstrated the advantages of this function. The most often mentioned from researchers is the ability to generate curves in a quicker time. In other hand, the property of end point interpolation makes this function as a suitable interpolant to connect point by point. Therefore, in curve reconstruction, we need a function that can interpolate every point. Hence, an appropriate interpolant can created by using cubic Ball basis.

Let $F_i$ and $F_{i+1}$ as a two-point edge while $P_{1,i}$ and $P_{2,i}$ are two control points on the cubic Ball. Then, by using the basic functions in (3) Ball cubic parametric curve defined by:

$$P_i(t) = F_i(1-t)^2 + 2P_{1,i}(1-t)t + 2P_{2,i}(1-t)^2 + F_{i+1}t^2$$ (4)

Figure 2 shows the example of cubic Ball curve.

### 3 Differential evolution

Differential Evolution (DE) was proposed by Rainer Storn [23]. DE is an extremely powerful yet simple evolutionary algorithm that improves a population of individuals over several generations through the operators of mutation, crossover and selection for global optimization. It has successfully been applied to diverse domains of science and engineering, such as mechanical engineering design, signal processing, chemical engineering, machine intelligence, and pattern recognition [24]. Besides that, DE also has been used to solve optimal power flow problems [16], [17] and [25]. The process involved in DE algorithm are Initialization, Mutation, Recombination and Selection. These process are described as below and being summarized in Figure 3.

**Fig. 3.** Process on DE algorithm.

#### 3.1 Initialization

The algorithm starts by creating an initial population of by using the following formula:

$$X_{i,j} = X_j^L + rand(0,1) \cdot (X_j^U - X_j^L)$$ (5)

where $rand(0,1)$ is a uniformly distributed random number lying between 0 and 1. Whereby $X_j^U$ and $X_j^L$ is an upper boundary and lower boundary of the population respectively. The population size is an algorithm control parameter selected by the user which remain unchanged throughout the optimization process.

#### 3.2 Mutation

DE process continued with the process of mutation strategy. In this process, each generation (or one iteration
of the algorithm) will change the population member known as target vector, $X_{i,G}$, by creating a donor vector $V_{i,G}$. The mutation is based on the difference between two individuals randomly chosen from the current population [26]. There are various methods of creating this donor vector that differentiates one DE scheme from another. In this paper, we only focus to the scheme of DE/rand/1 in producing the donor vectors. Several variance of DE mutation strategies are described in the following, [23],[27],[28]. In these scheme, for each vector $X_{i,G}$, a donor vector $V_{i,G}$ is generated according to the following:

$$V_{i,G} = X_{i,G} + F(X_{r2,G} - X_{r3,G})$$

(6)

The notation DE/rand/1 specifies that the vector to be perturbed is randomly chosen, and that the perturbation consists of one weighted difference vector. In this process, vector $X_{j1,G}, X_{j2,G}, X_{j3,G}$ are selected randomly such that they are mutually different indices from the current generation. The mutation factor, $F$ is normally used is $0.5 \leq F \leq 1$, [23] and also chosen $0 < F \leq 2$, [16] and [17].

### 3.3 Recombination

The recombination process incorporates successful solution from previous generation where the trial vector $U_{j,j,G}$ is produced from the target vector $X_{i,G}$ and elements of the donor vector $V_{i,G}$ with the probability of $CR$. This process is illustrated as follows:

$$U_{j,j,G} = \begin{cases} V_{j,i,G} & \text{if } \text{rand}_{j1}(0,1) \leq CR, \\ X_{j,j,G} & \text{or } j = j_{\text{rand}} \\ X_{j,i,G} & \text{else} \end{cases}$$

(7)

From equation (7), $CR$ become a control parameter of DE just like $F$. The value of the $CR$ is between the range of $[0,1]$.

### 3.4 Selection

The next step is called selection. This step will determine whether the target or the trial vector survives to the next generation. The selection operation is described as

$$X_{i,G+1} = \begin{cases} U_{i,G} , & \text{if } f(U_{i,G}) \leq f(X_{i,G}) \\ X_{i,G} , & \text{else} \end{cases}$$

(8)

where $f(X)$ is the function to be minimized. So, the best (lowest) fitness value will be included in the next generation. Mutation process, recombination, and the selection process will be repeated until some stopping criterion is reached. Commonly, stopping criteria are satisfied when a predefined small number of fitness value or a fixed number of generation or iteration have been reached.

### 4 Boundary extraction and corner detection

The proposed method starts with finding the boundary of the generic shape and then using the output to detect the corner points of the image. The image of the generic shape can be acquired by scanning or by some other method. The purpose of the boundary extraction is to produce an object's shape in graphical representation in a number of boundary points. There are a lots of algorithm that has been used in order to extract the boundary of the generic shape such as [29], [30], and Chain Codes [19]. But in this paper, we used the MATLAB function that called Boundaries in order to extract the boundary of the font images.

After finding out the boundary points, the next step is to find the corner detection of the generic shape. The corner points can also be described as the point that separates the outline into various segments. Some of the possible methods to detect the corner points is Davis algorithm. This algorithm has been applied in the research of [19] and [30]. For the detection of corners, in this paper, we used SAM06 algorithm [31].

According to [31], SAM06 algorithm is robust, simple to implement, efficient and performs well on noisy shapes as well. Detected corners by SAM06 were precise and without detecting any wrong ones. Besides that, we also used Douglas-Peucker algorithm [32] in order to reduce the data in a particular curve. The purpose of this algorithm is to create same curve by using a small number of data. We hope that this algorithm can reduce the computational time.

We have decided to use different types of image such as the Japanese (Kanji) character, and image of heart and fork. Details for the boundary extraction and corner detection of the images is shown in Table 1.

### 5 Chord length parameterization

In this paper, chord length parameterization is used to estimate the parametric value $t$ associated with each point. Chord length parameterization also has been used by [1], [5], [6], [10], [14], [21], and [27]. Given parametric curve $Q(t)$ over a certain domain $t \in [0,1]$, its chord length at a given point $Q(t)$ is define as:

...
\[
I_i = \begin{cases} 
0 & \text{if } i = 1 \\
|p_1p_2| + |p_2p_3| + \ldots + |p_ip_{i+1}| & \text{if } i < n \\
|p_1p_2| + |p_2p_3| + \ldots + |p_{n-1}p_n| & \text{if } i = n 
\end{cases}
\]

(9)

Table 1. Details of the boundary extraction and corner detection

| Original image | Name           | Boundary extraction and corner detection |
|----------------|----------------|------------------------------------------|
| @             | At(.getTag)    |                                          |
| 友            | Friend(Yuur)   |                                          |
| Fork          |                |                                          |
| 力            | Strength(Chikara) |                                      |

6 Curve reconstruction

In this section, we apply DE to solve curve reconstruction. The process of the forming curve requires several steps related as we can see in the Algorithm 1 for curve reconstruction. Steps 1, 2 and 3 already described in section 4, while step 4 we will use the DE to solve curve reconstruction by using three different curves as mention in section 2.

However, to fulfill the different result, the method of sum square error (SSE) used in order to calculate the different or errors between the parametric points of generated curve (P(t_i)) and given points (F_{i,j}). Let \( F_{i,j} = (x_{i,j}, y_{i,j}) \) the data points are given or extracted where \( i = 1, 2, \ldots, n \) and \( j = 1, 2, \ldots, m \). Hence, by using the formula of SSE, we define

\[
S_i = \sum_{j=1}^{m} \left( P(t_i) - F_{i,j} \right)^2, \quad i = 1, 2, \ldots, n
\]

where \( I_i \) refer to parameterization as explained in section 5.

Algorithm 1: Curve reconstruction:

**Input:** Data of image \( F_{i,j} = (x_{i,j}, y_{i,j}) \)

**Output:** Curve fitting, \( P(t_i) \)

Step 1: Get appropriate image to reconstruct
Step 2: Boundary extraction from the image to acquire data points.
Step 3: Corner points detected
Step 4: The DE for curve reconstruction using:
   a) Cubic Ball curve.
   b) Cubic Bezier
Step 5: Calculate the number of errors and total time taken.

7 Result and discussions

Generally, DE could be used for curve fitting. In this paper, two type of splines explained. Before using the splines, a few pre-processing steps needed in order to interpolate the data. However, to test which curve have a good result, the curve fitting generated needed to compare with original image to measure the error or known as SSE. As the result demonstrated, the reconstruction have similar image visually, while the error as shown in Table 2 give different result where the cubic Ball lead all from the four examples. Table 3 reflect the result in terms of time where the cubic Ball give impressive result compare to cubic Bezier curve. This is because of the degree in cubic Ball less cubic Bezier. Figure 4 illustrate the curve reconstruction for the image selected.

Table 2. Average error obtained by each image.

| Image  | Cubic Ball | Cubic Bezier |
|--------|------------|--------------|
| AT     | 199.9212   | 199.9878     |
| Fork   | 8.9260     | 9.2857       |
| Friends| 384.2103   | 411.1150     |
| Strength| 572.9575   | 573.6694     |
Table 3. Average time taken for each image

| Image | Cubic Ball | Cubic Bezier |
|-------|------------|--------------|
| AT    | 11.7240    | 12.8224      |
| Fork  | 5.1706     | 5.2744       |
| Friends | 11.3632   | 12.1389      |
| Strength | 7.5557    | 8.1168       |

Acknowledgement

The authors highly acknowledge the referee's valuable comments on the improvement of this paper. The authors also grateful for the financial support provided by UniMAP, under Short Term Grants-(STG) (9001-00531).

References

1. N.A. Rusdi and Z.R. Yahya, Reconstruction of generic shape with cubic Bézier using least square method, International Conference on Mathematics, Engineering & Industrial Applications 2014 (ICoMEIA 2014), AIP Publishing 1660, 050004 (2015)
2. M. Sarfraz and A. Masood, Capturing outlines of planar images using Bézier cubics, Computers & Graphics 31, 719-729(2007)
3. M. Sarfraz and M.A. Khan, An automatic algorithm for approximating boundary of bitmap characters, Future Generation Computer Systems 20, 1327-1336 (2004)
4. L. Shao and H. Zhou, Curve fitting with Bezier cubics, Graphical models and image processing 58, 719-729 (2007)
5. N.A. Rusdi and Z.R. Yahya, Reconstruction of Arabic Font with Quartic Bézier Curve, Sains Malaysiana 44, 1209-1216 (2015)
6. N. Roslan and Z.R. Yahya, Reconstruction of egg shape using B-spline, International Conference on Mathematics, Engineering & Industrial Applications 2014 (ICoMEIA 2014), AIP Publishing 1660, 050016 (2015)
7. M. Sarfraz, M. Irshad and M.Z. Hussain, Reverse Engineering of Planar Objects Using Gas, Sains Malaysiana 42, 1167-1179 (2013)
8. Z.A. Hasan, Z.R. Yahya, A.R.M. Piah, Differential Evolution for Reverse Engineering Using Cubic Ball Curve, EDUCATUM Journal of Science, Mathematics and Technology 2, 58-68 (2015)
9. K. Barkeshli, M. Mokhtari and M. Amiri, Image reconstruction of impenetrable cylinders using cubic B-splines and genetic algorithms, In Antennas and Propagation Society International Symposium, 2001. IEEE 2, 686-689 (2001)
10. N. Roslan and Z.R. Yahya, Reconstruction of Font with Cubic Bézier Using Differential Evolution Sains Malaysiana 44, 1203-1208 (2015)
11. P. Pandunata and S.M.H. Shamsuddin, Differential evolution optimization for Bézier curve fitting, In Computer Graphics, Imaging and Visualization (CGIV), 2010 Seventh International Conference on IEEE 68-72 (2010, August)
12. A. Singh, An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem, Applied Soft Computing 9, 625–631 (2009)
13. G. Zhu and S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function
14. N.A Rusdi and Z.R. Yahya, Reconstruction of Arabic Font using Artificial Bee Colony Algorithm, Acceptance for Malaysian Technical Universities Conference on Engineering and Technology 2015 (MUCET 2015), Johor Bharu, 11-13 October 2015.

15. D. Karaboga, and B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Applied soft computing 8, 687-697 (2008)

16. N. Sinsuphun, U. Leeton, U. Kwannetr, D. Uthitsunthorn and T. Kulworawanichpong, Loss minimization using optimal power flow based on swarm intelligences, Transactions on Electrical Engineering, Electronics, and Communications 9, 212-222 (2011)

17. S.P. Karthikeyan, K. Palanisamy, L.J. Varghese, L.J. Raglend and D.P. Kothari, Comparison of intelligent techniques to solve economic load dispatch problem with line flow constraints, In Advance Computing Conference, 2009. IACC 2009. IEEE International 446-452 (2009, March)

18. J. Tverdik, Adaptive differential evolution and exponential crossover. Proceedings of IMCSIT 2008 927-931 (2008)

19. M. Sarfraz and M.A. Khan, Automatic outline capture of Arabic fonts, Information Sciences 140, 269-281 (2002)

20. M. Sarfraz and M.F.A. Razzak, An algorithm for automatic capturing of the font outlines, Computers and Graphics (Pergamon) 26, 795-804 (2002)

21. N. Roslan and Z.R. Yahya, Different Mutation Strategies For Reconstruction of Japanese Character, Acceptance For Malaysian Technical Universities Conference On Engineering And Technology 2015 (MUCET 2015), Johor Bharu, 11-13 October 2015.

22. A. A. Ball. CONSURF. Part one: Introduction of the conic lofting tile 6, 243-249 (1974)

23. R. Storn, On the usage of differential evolution for function optimization, In Fuzzy Information Processing Society, 1996. NAFIPS., 1996 Biennial Conference of the North American. IEEE 519-523 (1996, June)

24. S. Das, A. Abraham, U.K. Chakraborty and A. Konar, Differential evolution using a neighborhood-based mutation operator, Evolutionary Computation, IEEE Transactions on 13, 526-553 (2009)

25. M. Varadarajan and K.S. Swarup, Solving multi-objective optimal power flow using differential evolution, Generation, Transmission & Distribution, IET 2, 720-730 (2008)

26. R.S. Prado, R.C. Silva, F.G. Guimaraes and O.M. Neto, Using differential evolution for combinatorial optimization: A general approach, In Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on. IEEE 11-18 (2010, October)

27. N. Roslan, and Z.R. Yahya, Different Mutation Strategies For Reconstruction Of Japanese Character, Acceptance For Malaysian Technical Universities Conference On Engineering And Technology 2015 (MUCET 2015), Johor Bharu, 11-13 October 2015.

28. S. Das, A. Abraham and A. Konar, Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives, In Advances of Computational Intelligence in Industrial Systems. Springer Berlin Heidelberg 1-38 (2008)

29. G. Avrahami and V. Pratt, Sub-pixel edge detection in character digitization, Raster Imaging and Digital Typography II 54-64 (1991)

30. K. Ith and Y. Ohno, A curve fitting algorithm for character fonts, Electronic publishing 6, 195-205 (1993)

31. M. Sarfraz, Corner Detection for Curve Segmentation, Interactive Curve Modeling: With Applications to Computer Graphics, Vision and Image Processing 209-240 (2008)

32. The MathWorks Inc. (MATLAB), Ramer-Douglas Peucker algorithm demo by Ligong Han, http://www.mathworks.com/matlabcentral/fileexchange/41986-ramer-douglas-peucker-algorithm (2014)