FITTING OF BEZIER SURFACES USING THE FIREWORKS ALGORITHM

K. Srinivas Reddy[1], Amitava Mandal[2], Krishna Kumar Verma[3] and G. Rajamohan[4]
[1]PG Student, [2]Professor, [3]Associate Professor and [4]Assistant Professor
Department of Manufacturing Engineering
National Institute of Foundry and Forge Technology, Ranchi, India

ABSTRACT

This paper presents an extension of our work entitled "Fitting of Bezier curves using the fireworks algorithm" to solve the problem of parameterization of given set of data points for least squares fitting with Bezier surfaces. Performance of the proposed method is validated through several example surfaces of varying complexities.

KEYWORDS: *Bezier surfaces, Computer-aided design, Fireworks algorithm, Parameterization methods, Least squares method, Swarm intelligence.*

I. INTRODUCTION

Many industrial and technological areas, such as computer animation, computer graphics, computer-aided design and manufacturing (CAD/CAM), gaming, medical imaging, reverse engineering, virtual reality, etc. make use of parametric surface fitting to point clouds. Reverse engineering is performed to transform real-life engineering objects into their CAD models and concepts to improve their design, manufacture and analysis. It finds applications where the original drawings of parts are not available. Figure 1 shows the various phases of a generic reverse engineering process [1]. Interested reader may consult [2] for detailed information on these phases. The data points are acquired in *Cartesian space* using the appropriate coordinate measuring equipment (*first phase*), pre-processed (*second phase*) and then segmented (*third phase*). As CAD models are generally created in *parametric space*, the *fourth phase* (known as *parameterization*) assigns an unique parameter value to each point in the point cloud for use in fitting of surfaces (*fifth phase*) that finally lead to CAD models (*sixth phase*). The accuracy of fitted surface therefore depends on adequate parameterization even though the other phases are also important. Among the several parameterization methods available, the *uniform*, *chord length* and *centripetal* methods are commonly used. These parameterization methods, which generally work well with data points arranged as organized grids (Figure 2), fail miserably with unorganized data points.

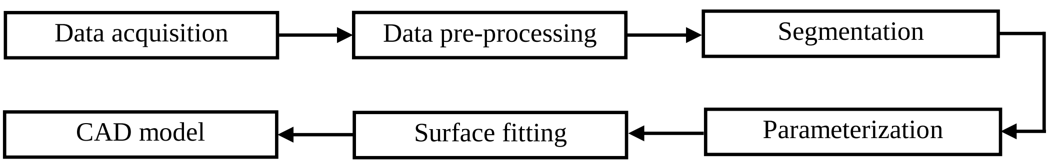


Figure 1. Phases of reverse engineering

Fitting of parametric surfaces is generally based on the *least squares method (LSM)*. Fitting process starts with the estimation of parameters and knot vectors, followed by the determination of control points of fitted surface by minimizing the sum of squares of distances between the data points and the fitted surface. An attempt to fit B-spline surfaces for randomly measured points using LSM fitting has been reported [3]. This method uses an *initial base surface* and improvises it to obtain better surfaces. Azariadis introduced the concept of *dynamic base surfaces* (DBS), which dynamically adapt to the three-dimensional shape implied by the clouds of points, for parameterization of scattered data points. The authors assumes the existence of a boundary defined by a closed path of four curves in the point cloud [1].

It may be noted that the number of unknown variables to be solved increases with large numbers of data points. In such cases, solving the system of equations using conventional optimization techniques may not be feasible or may result in inferior surfaces. This motivated the researchers to explore the possibility of solving the fitting problem using *neural networks* [4-8] and *evolutionary algorithms* such as *genetic algorithms* [10-11], *artificial immune system* [12], *swarm intelligence algorithms* [14-16], etc. Some of the research efforts related to use of evolutionary algorithms to curve fitting may be found in [9], while those of surface fitting are presented below.

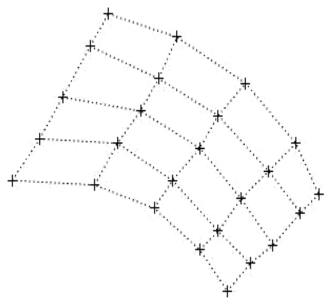


Figure 2. Grid of selected surface points

Genetic algorithms are widely used evolutionary algorithms for solving more complex optimization problems. Galvez, et al. developed two Artificial Intelligence (AI) techniques for fitting Bezier curves and surfaces using the least-squares approximation method, viz. a genetic algorithms based approach for curve and surface parameterization, and a functional networks scheme for handling the additional functional constraints used in fitting [10]. Galvez, et al. applied the genetic algorithms to iteratively fit a given cloud of noisy 3D data points by using strictly polynomial B-spline surfaces. Their method used genetic algorithms in two steps, viz. the first one determines the parametric values of data points and the later computes the surface knot vectors. The fitted surface is then calculated by least-squares through either SVD (singular value decomposition) or LU decomposition methods [11].

Galvez, et al. developed an approach for effectively applying the *clonal selection algorithm* (CSA) for accurate fitting of 3D noisy data points, obtained through either laser scanning or other digitizing methods, with the Bezier surfaces [12]. The CSA is an artificial immune system (AIS) algorithm. The *swarm intelligence* algorithms are being used in recent times to solve the parametric surface fitting problems. Swarm intelligence is the study of computational systems inspired by *collective intelligence* [13]. Galvez, et al. used the *particle swarm optimization* algorithm for obtaining suitable parameters for Bezier surface reconstruction [14]. Galvez and Iglesias applied the *firefly algorithm*, a powerful metaheuristic algorithm inspired from the social flashing behaviour of *fireflies* in nature, to solve the parameterization of polynomial Bezier surfaces [15]. Iglesias, et al. proposed a *bat algorithm* based method for optimal parameterization of polynomial Bezier surfaces for least-squares fitting. Their approach has been shown to yield better numerical and visual results even for point clouds of strong irregular patterns [16].

The evolutionary algorithm-based fitting methods involve tuning of several parameters that form the basic framework. Their values usually vary from problem to problem and also significantly affect the performance of the particular method. The user will have to perform a large number of trials before setting the values of these parameters for every new problem. This necessitates to use an algorithm involving parameters that can be easily set. One such algorithm is the *fireworks algorithm* [17]. The fireworks algorithm involves only four parameters that are almost same for every parameterization problem. Some of the basic details of this algorithm are given in Section III. In the present work, a fireworks algorithm based parameterization approach is proposed for the fitting of polynomial Bezier surfaces and its performance has been tested with several example cases.

Remainder of the paper is organized as follows. Section II presents some theoretical topics relevant to this paper and Section III describes about the proposed method. Section IV presents and discusses the results obtained using the proposed method and finally the conclusions are presented in Section V.

II. LEAST SQUARES FITTING OF BEZIER SURFACES

The *Bezier surface* can be mathematically defined as [18]:

$$p(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} p_{i,j} B_{i,m}(u) B_{j,n}(v); \quad 0 \le u \le 1;\ 0 \le v \le 1 \tag{1}$$

where, the vectors $p_{i,j}$ represent the $((m + 1) \times (n + 1))$ *vertices* of a *characteristic or control polygon net* and $B_{i,m}(u)$ is the i^{th} Bernstein basis function of degree m along parameter u and $B_{j,n}(v)$ is the j^{th} Bernstein basis function of degree n along parameter v. The vertices are known as *control points* as they control the shape of the Bezier surface. Eq. (1) can be used to compute the points on a Bezier surface. The basis function values can be computed using Eq. (2).

$$B_{i,m}(u) = \binom{m}{i} u^{i}\ (1-u)^{m-i}; \quad B_{j,n}(v) = \binom{n}{j} v^{j}\ (1-v)^{n-j} \tag{2}$$

Let us consider the scenario wherein the points or *data points* are known and the control points are to be estimated, as in the case of reverse engineering. Assume that there are N data points (D_k) in a *3-dimensional space*. The control points of the Bezier surface can be estimated using the *least squares* (*LS*) fitting method, i.e. by *minimizing the sum of squared errors* (E) over the set of data points. The LS fitting can be represented as follows:

$$E = \sum_{k=1}^{N} [D_k - p(u,v)]^2 \tag{3}$$

The system of equations to be solved by LS fitting can be represented using Eq. (4).

$$D_k = \sum_{i=0}^{m}\sum_{j=0}^{n} Q_{i,j} B_{i,m}(u) B_{j,n}(v); \quad 0 \le u \le 1;\ 0 \le v \le 1;\ k = 0,\ldots,(N-1) \tag{4}$$

Eq. (4) can be written in matrix form as shown below.

$$[D] = [B][Q] \tag{5}$$

where, $[D]$ represents the given data points in vector form, $[Q]$ is the control points in matrix form and the $[B]$ is the matrix with blending function values as its elements. The control points $[Q]$ may be obtained using Eq. (6). Solving Eq. (5) amounts to LS fitting shown in Eq. (3).

$$[Q] = [B]^{-1}[D] \tag{6}$$

III. PROPOSED METHOD

To solve the system of equations in Eq. (4), suitable values of parameters u and v are to be found first. Given the nonlinear nature of the problem, parameters are to be estimated through optimization only. The *fireworks algorithm* is used here for this purpose. The flowchart of the proposed method is shown in Figure 3. The *preparatory steps* needed for the fireworks algorithm are as follows:

1) The parameter values are considered as fireworks and are encoded as real coded vectors of length equal to number of given data points (N), i.e. $u = [u_1, u_2, \ldots, u_N]$ and $v = [v_1, v_2, \ldots, v_N]$.
2) The objective function is the function E given in Eq. (3).

3) The degrees of the underlying surface and number of control points are not known apriori and are dependent on the complexity of the surface. The proposed method therefore starts with a minimum number of control points, and increases it until the error reduces below a certain threshold value.
4) Setup the control parameters, namely the *population size* (n_{pop}), the *number of iterations* (n_{ite}), *total number of sparks* (n_{spr}) and the *maximum explosion amplitude* (A).

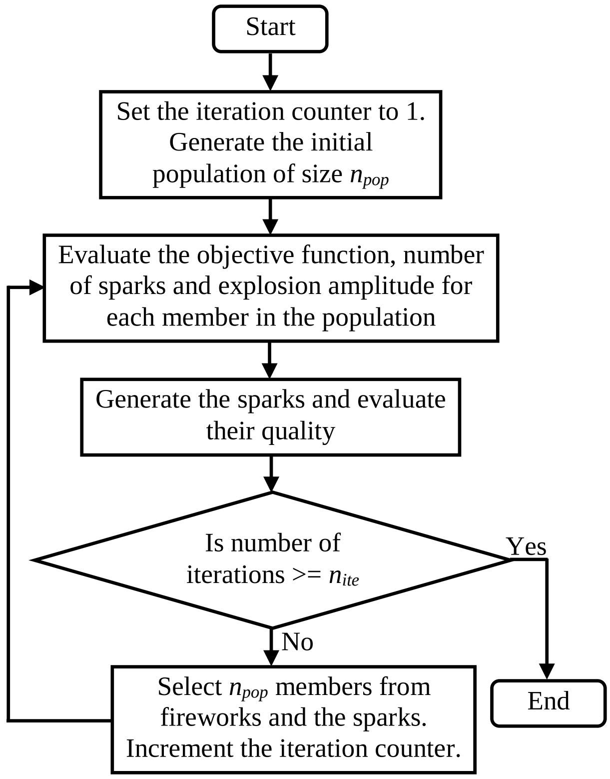


Figure 3. Flowchart of the proposed method

3.1 Number of sparks

The number of sparks to be generated for each member in the population can be found using Eq. (7).

$$s_i = n_{spr} \frac{E_{\max} - E_i + \zeta}{\sum_{i=1}^{n_{pop}} (E_{\max} - E_i) + \zeta} \tag{7}$$

where, ζ is a small constant introduced to avoid the zero division error.

3.2 Amplitude of explosion

The amplitude of explosion of every member in the population can be estimated using Eq. (8).

$$A_i = A \frac{E_i - E_{\min} + \zeta}{\sum_{i=1}^{n_{pop}} (E_i - E_{\min}) + \zeta} \tag{8}$$

where, ζ is a small constant introduced to avoid the zero division error.

3.3 Generation of sparks

Sparks are generated by mimicking the explosion process. Considering the convergence and diversity of explosion, two spark generation methods are generally employed. One method generates the sparks towards the best among the population and the other generates at random. Both the methods are given in Eq. (9) and Eq. (10), where the superscripts *s* and *f* stand for sparks and fireworks respectively.

$$
\begin{aligned}
& u_i^s = u_i^f + A_i \ rand(0,1) \ sign(u_{ibest}^f - u_i^f) \\
& u_i^s = u_i^f + (-1)^{randi(2,1,1)} \ A_i \ rand(0,1) \\
& u_i \in u; \ i = 1,2,\ldots,N;
\end{aligned} \tag{9}
$$

$$
\begin{aligned}
& v_i^s = v_i^f + A_i \ rand(0,1) \ sign(v_{ibest}^f - v_i^f) \\
& v_i^s = v_i^f + (-1)^{randi(2,1,1)} \ A_i \ rand(0,1) \\
& v_i \in v; \ i = 1,2,\ldots,N;
\end{aligned} \tag{10}
$$

3.4 Selection of sparks or locations

At the end of every iteration, the current best member, for which the objective function is optimal among population is kept for the next explosion generation. After that, (n_{pop} - 1) members are selected based on their distance to other members so as to keep diversity.

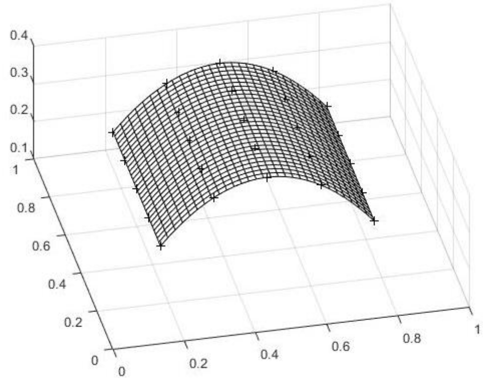
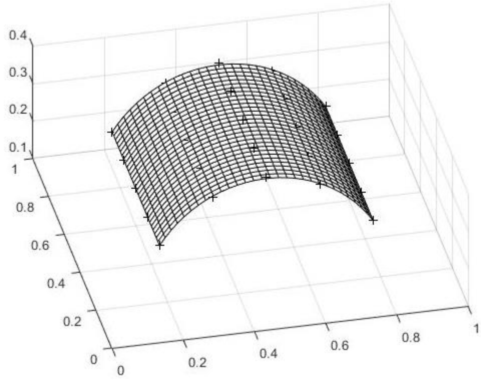
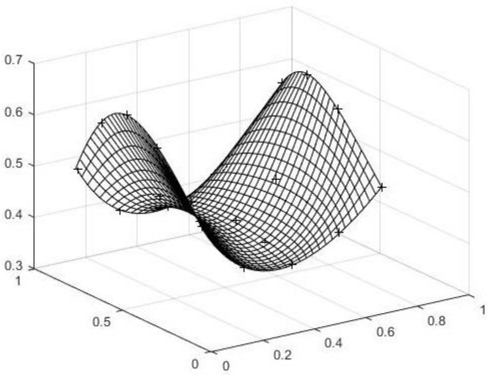
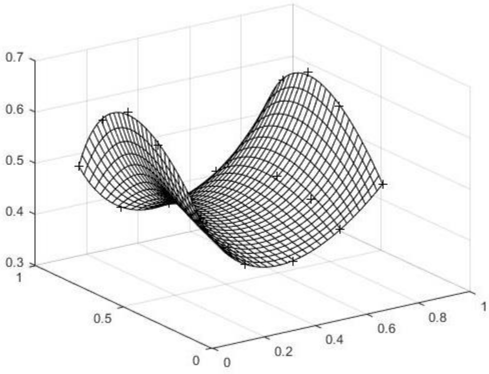
IV. RESULTS AND DISCUSSION

The proposed parameterization method has been implemented using MATLAB language. The chord length parameterization method has also been implemented for organized point clouds only as it cannot be used with unorganized point clouds. Several example surfaces of varying complexities have been used to compare these methods. The results obtained are presented and discussed below.

4.1 Organized point clouds

Three examples have been considered in this case. The Bezier surfaces approximated by chord length and proposed parameterization methods are shown in Table 1 and corresponding error-of-fit values are shown in Table 2. The *plus marks* in images in Table 1 are the cloud points while the mesh shows the approximated surface. The axes are marked in normalized coordinates. First example is relatively a very simple surface and the last one is the most complicated of the three.

Table 1. Organized point clouds approximated using chord length and proposed parameterization methods

Example No.	Parameterization method	
	Chord length parameterization	Proposed parameterization
O1		
O2		

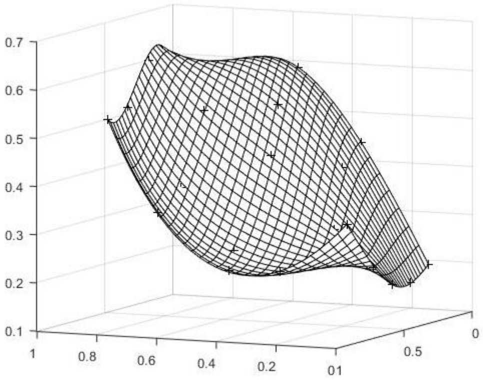
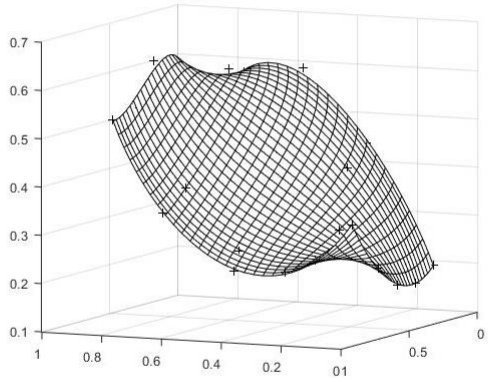
O3		

Table 2. Error-of-fit values for organized point clouds

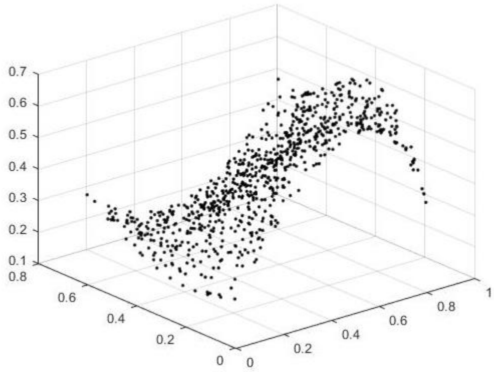
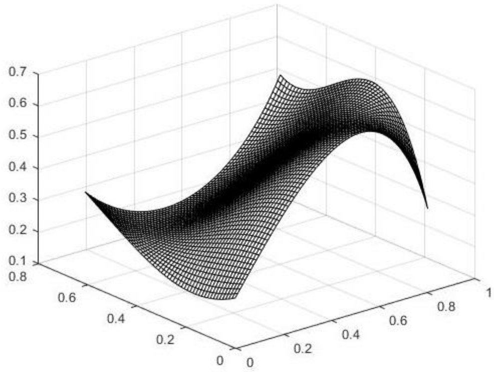
Example No.	Chord length parameterization			Proposed parameterization		
	E_x	E_y	E_z	E_x	E_y	E_z
O1	1.05E-30	6.53E-31	2.04E-04	4.36E-31	7.89E-31	8.17E-32
O2	3.10E-04	2.32E-05	4.44E-04	7.17E-05	1.99E-04	1.05E-04
O3	1.20E-03	7.05E-04	1.30E-03	2.17E-04	9.47E-04	3.83E-04

In all the cases, it may be seen that the proposed method is able to approximate the points very well, both in terms of visual pleasing Bezier surfaces (Table 1) and smaller error-of-fit values (Table 2). The chord length method also performs close to the proposed method, a fact that may be attributed to the uniform spacing of points in the point cloud. Higher difference in error-of-fit values between the two parameterization methods can be observed as the complexity of the underlying surface increases.

4.2 Unorganized point clouds

Fitting of unorganized points clouds to parametric surfaces is a challenging task. As stated earlier, the existing parameterization methods cannot be applied to unorganized point clouds, given their nature of requirement discussed in [9]. However, the proposed parameterization method can handle such unorganized point clouds. To illustrate this, several example cases have been considered. Five such example point clouds and the corresponding approximated Bezier surfaces using the proposed method are shown in Table 3 and their error-of-fit values are shown in Table 4. Axes of approximated surface are marked in the normalized coordinates. The number of control points and degrees along parameters *u* and *v* are taken at the lowest values initially and are increased till desired accuracy level is reached. In other words, the initial control net of size 4 × 4 and degrees of 3 along both *u* and *v* are taken as the starting point.

Table 3. Unorganized point clouds and corresponding approximated Bezier surfaces

Example No.	Point cloud	Approximated surface
U1		

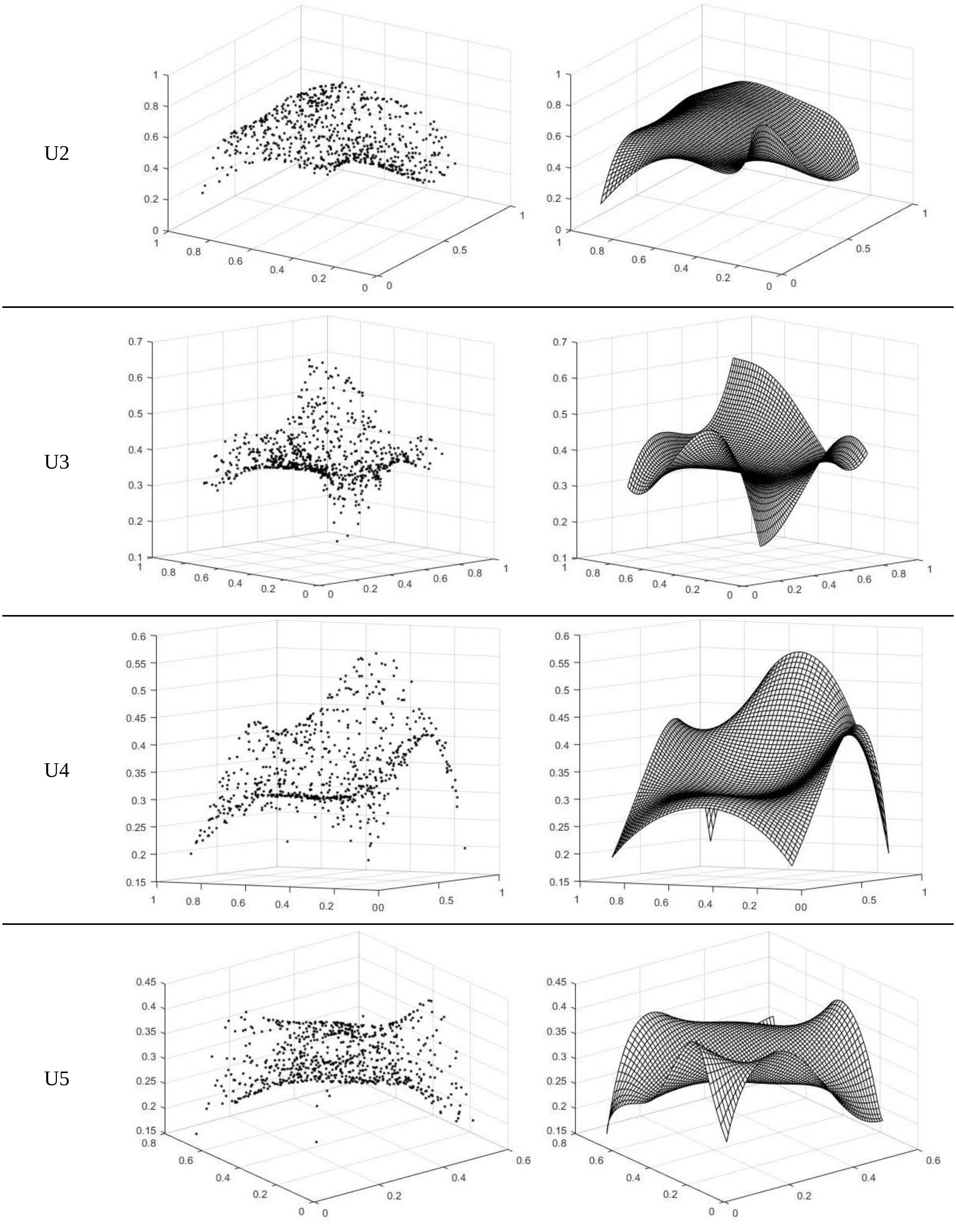


Table 4. Error-of-fit values for example cases of surface fitting to unorganized points

Example	E_x	E_y	E_z	Polygon Net
U1	6.19E-04	5.09E-04	7.13E-04	4 × 4
U2	4.81E-04	5.38E-04	6.11E-04	5 × 5
U3	3.78E-04	4.33E-04	3.86E-04	4 × 5
U4	5.42E-04	5.59E-04	3.47E-04	5 × 5
U5	4.63E-04	5.40E-04	2.27E-04	6 × 7

In all the cases, it may be seen that the proposed parameterization method is able to approximate the given point cloud very well in terms of visual appearance as seen in Table 3. The error-of-fit values (E_x, E_y and E_z) in Table 4 are very low. Apart from this, it can also be seen that the size of control net is also small. As Bezier surfaces have their degrees tightly coupled with the number of control points, smaller number of control points mean lower surface degrees (along *u* and *v*). It can be concluded that the proposed method is able to handle the parameterization problems associated with the unorganized points very effectively.

V. CONCLUSIONS AND FUTURE SCOPE

In LSM fitting of parametric Bezier surfaces to point clouds, the parameterization method plays a key role in generating aesthetically pleasing surfaces. A new parameterization method has been proposed in the present work to address the Bezier surface fitting problem. The proposed method is based on the fireworks algorithm. The advantage is this algorithm is that it requires only very few parameters to be set. Test cases involving organized and unorganized point clouds have been taken to verify the efficacy of the proposed method. It has been found to outperform the chord length parameterization method in case of organized point clouds. A comparison of this method with proposed method in case of unorganized points is not possible for obvious reasons. In both cases (of organized and unorganized point clouds), the proposed algorithm has been found to yield visually more appealing surfaces with very low values of errors of fit.

An extension of the proposed approach to approximate given sets of organized or unorganized points using B-splines and NURBS may form the future work.

REFERENCES

[1]. P. N. Azariadis (2004) "*Parameterization of clouds of unorganized points using dynamic base surfaces*", Computer Aided Design, Vol. 36, pp. 607-623.
[2]. T. Varady, R. R. Martin & J. Cox (1997) "*Reverse engineering of geometric models*", Computer Aided Design, Vol. 29, pp. 255-268.
[3]. W. Ma & J. P. Kruth (1995) "*Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces*", Computer Aided Design, Vol. 27, pp. 663-675.
[4]. P. Gu & X. Yan (1995) "*Neural network approach to the reconstruction of freeform surfaces for reverse engineering*", Computer Aided Design, Vol. 27, No. 1, pp. 59-64.
[5]. J. Barhak & A. Fisher (2001) "*Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques*", IEEE Transactions on Visualization and Computer Graphics, Vol. 7, Issue 1, pp. 1-16.
[6]. F. L. Krause, A. Fischer, N. Gross and J. Barhak (2003) "*Reconstruction of freeform objects with arbitrary topology using neural networks and subdivision techniques*", Annals of the CIRP, Vol. 52, No. 1, pp. 125-128.
[7]. M. Hoffman (2005) "*Numerical control of Kohonen neural network for scattered data approximation*", Numerical Algorithms, Vol. 39, pp. 175-186.
[8]. M. E. Yumer & L. B. Kara (2012) "*Surface creation on unstructured point sets using neural networks*", Computer Aided Design, Vol. 44, pp. 644-656.
[9]. K. Srinivas Reddy, Amitava Mandal, Krishna Kumar Verma & G. Rajamohan (2016) "*Fitting of Bezier curves using the fireworks algorithm*", International Journal of Advances in Engineering & Technology (Communicated)
[10]. A. Galvez, A. Iglesias, A. Cobo, J. P. Pey & J. Espinola (2007) "*Bezier curve and surface fitting of 3D point clouds through genetic algorithms, functional networks and least squares approximation*", Proc. of International Conference on Computational Science and its Applications, Vol. 4706, pp. 680-693.
[11]. A. Galvez, A. Iglesias & J. P. Pey (2012) "*Iterative two step genetic algorithm based method for efficient polynomial B-Spline surface reconstruction*", Information Sciences, Vol. 182, pp. 56-76.
[12]. A. Galvez, A. Iglesias & A. Avila (2013) "*Immunological based approach for accurate fitting of 3D noisy data points with Bezier surfaces*", Procedia Computer Science, Vol. 18, pp. 50-59.
[13]. J. Brownlee (2012) *Clever Algorithms*, www.cleveralgorithms.com, accessed on 26th May 2016.
[14]. A. Galvez, A. Cobo, J. P. Pey & A. Iglesias (2008) "*Particle swarm optimization for Bezier surface reconstruction*", Proc. of the 8th International Conference on Computational Science (Part II).

[15]. A. Galvez & A. Iglesias (2013) *Firefly algorithm for polynomial Bezier surface parameterization*, Journal of Applied Mathematics, Vol. 2013, http://dx.doi.org/10.1155/2013/237984.
[16]. A. Iglesias, A. Galvez & M Collantes (2015) "*A bat algorithm for polynomial Bezier surface parameterization from clouds of irregularly sampled data points*", Proc. of International Conference on Natural Computation, pp. 1034-1039.
[17]. Y. Tan & Y. Zhu (2010) "*Fireworks algorithm for optimization*", Advances in Swarm Intelligence, Vol. 6145, pp. 355-364.
[18]. M. E. Mortenson (2006) *Geometric Modelling*, 3rd Edition, Industrial Press Inc.

Authors

K. Srinivas Reddy received his B. Tech. (Mechanical Engineering) degree from JNTU, Kakinada. He is currently pursuing his M. Tech. (Manufacturing Engineering) degree at National Institute of Foundry & Forge Technology, Ranchi. His research interests include computer aided design and other aspects of manufacturing engineering.

Amitava Mandal received his B. Tech. (Mechanical Engineering), M. Tech. (Mechanical Engineering) and Ph. D. (Mechanical Engineering) degrees from Indian Institute of Technology Delhi. His teaching and research interests include the finite element analysis, non-conventional machining, welding and other aspects of manufacturing engineering.

Krishna Kumar Verma received his M. Sc. (Mathematics) degree from Bhagalpur University, Bhagalpur and M. Phil. (Applied Mathematics) degree from Indian School of Mines, Dhanbad. His teaching and research interests include computer aided design, numerical methods and operations research.

G. Rajamohan received his B. E. (Mechanical Engineering) degree from College of Engineering, Anna University, Chennai, M. E. (Manufacturing Technology) degree from National Institute of Technology, Tiruchirappalli and Ph. D. (Mechanical Engineering) degree from Indian Institute of Technology Madras. His teaching and research interests include machining, metrology and computer aided inspection, image processing and applications of computers in design and manufacturing.