Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Design Automation Conference
IDETC/DAC 2011
August 28-31, 2011, Washington, DC, USA

DETC2011-48324

CONCEPTUAL DESIGN OF FREEFORM SURFACES FROM UNSTRUCTURED POINT SETS USING NEURAL NETWORK REGRESSION

Mehmet Ersin Yumer
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
Email: meyumer@cmu.edu

Levent Burak Kara
Department of Mechanical Engineering
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
Email: lkara@cmu.edu

ABSTRACT

This paper presents a new point set surfacing method that employs neural networks for regression. Our technique takes as input unstructured and possibly noisy point sets representing two-manifolds in $\mathbb{R}^3$. To facilitate parametrization, the set is first embedded in $\mathbb{R}^2$ using neighborhood preserving locally linear embedding. A neural network is then constructed and trained that learns a mapping between the embedded 2D parametric coordinates and the corresponding 3D space coordinates. The trained network is then used to generate a tessellation that spans the parametric space, thereby producing a surface in the original space. This approach enables the surfacing of noisy and non-uniformly distributed point sets, and can be applied to open or closed surfaces. We show the utility of the proposed method on a number of test models, as well as its application to freeform surface creation in virtual reality environments.

INTRODUCTION

Current computer-aided design (CAD) tools are not well suited for use in early conceptual design phases where rapid idea generation and exploration from unrefined, incomplete user input is paramount [1, 2]. This is a major concern in shape design specifically, as current tools are tailored for detailed design stages where the designer must negotiate a large number of parameters involving complex mathematical descriptors and relationships. Tools for ideation and early assessment, however, should provide the user with rapid and intuitive means for shape creation, thereby eliminating premature commitments to well developed but unpromising concepts [2, 3].

In this paper, we present a new surface design method that can take as input 3D point sets, and can generate freeform open or closed surfaces through a neural network based regression algorithm. In this work, point sets of interest can be sparse, unstructured, and unevenly distributed, and devoid of normal vector information. Such point sets frequently arise with the use of new generation input devices such as 3D optical or magnetic trackers in VR environments (Fig. 1) where the points are sampled from trackers attached to the users' hands or any part of their bodies. Such point sets are considerably different in nature than the widely studied class of range data, where dense point sets are sampled directly from the surface they represent. In surface design from point tracking, however, one rarely obtains a full and dense coverage of the intended surface. Moreover, point sampling may exhibit significant non-uniformity based on the users' motion speed and their focus on particular regions of the design. The long term goal of the proposed work is thus to provide industrial surface design algorithms that can operate on tracking data to produce surfaces with controllable aesthetic qualities and associated mechanisms enabling further detailed refinement on the initial data.

As one step toward this goal, we present a neural network based surface regression method that takes as input open or closed point sets in $\mathbb{R}^3$, and generates free form surfaces through a parametric embedding and tessellation in $\mathbb{R}^2$. The parametric embedding is achieved through a local neighborhood preserving

Copyright © 2011 by ASME

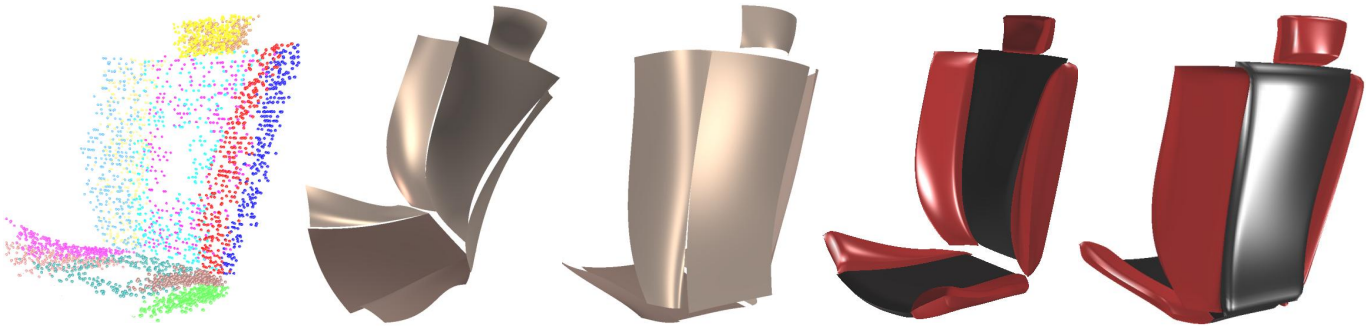

FIGURE 1. Car seat conceptual design. Point clouds, neural network regressed surfaces, trimmed design.

method. Once a parametrization of the input point set is computed, a mapping between the parametric coordinates of input points in $\mathbb{R}^2$ and their corresponding 3D design space coordinates is trained on a multi-layer, feed-forward, back-propagation neural network. A tessellation created in the parametric domain is then fed to the trained network which results in the synthesis of a two-manifold surface in the design space. A key advance in the proposed work is that the surface complexity is dictated by the network topology that iteratively minimizes the under-fit and over-fit to the available data. This approach is in contrast to methods that require the designer to inspect the underlying point set to decide the degree or functional form of the fitted surfaces. We demonstrate that the proposed approach can be used for creating free-form surfaces from arbitrary point sets, as well as from point sets specifically arising from tracking data.

RELATED WORK

In this section we review the previous work in surface creation, fitting, and approximation of point sets based on the surface representations used; parametric, mesh based and implicit, followed by a review of the use of neural networks in this field.

Parametric Surfaces: Parametric surfaces are one of the most widely used representations as they enable compact descriptions, straightforward tessellations with arbitrary resolutions. Gregorski *et al.* [4] introduced a B-spline surface reconstruction method for point sets. Their approach utilizes a quadtree like data structure to decompose the point set into multiple smaller point sets. Least squares quadratic fitting of each subpoint set is then followed by the degree elevation to B-spline surfaces and blending. Bae *et al.* [5], focusing primarily on laser range scanned data, introduced orthogonal coordinate transformations for NURBS surface fitting. The point set is first transformed into an orthogonal coordinate system, followed by B-spline fitting which is finally converted to NURBS surfaces. Adaptive fitting techniques introduced by Pottmann *et al.* [6, 7] utilize an active contour model which gradually approximates the targeted model shape. This iterative approximation minimizes a quadratic functional composed of an internal surface energy for smoothness and an approximation error for fitting. Lin *et al.* [8] introduce an iterative NURBS curve and surface fitting methodology to a given point set which is able to interpolate the point set. The major restriction of their approach is that the point set should be pre-ordered. Following a similar approach boundary condition satisfying NURBS surface fitting is also achieved [9]. The neural network in our method is similar to parametric surface definitions in the sense that it enables arbitrary resolution tessellation straightforwardly and has a compact definition. However, the proposed method differs from parametric fitting in that the functional form of the surface is dictated by the optimized network topology rather than requiring the user to decide the parameters of the fit. As shown in the following examples, however, the proposed method can be easily modified to fit a prescribed functional form such as a parametric surface when desired.

Mesh-based Surfaces: Mesh-based or polygonal surfaces enable a fastest rendition of many different surface representations while exhibiting large space requirements. In an early work, Hoppe *et al.* [10] address the problem by using local linear approximations of the point set to create a mesh-based surface that approximates the point set. The first provably correct mesh-based surface fitting algorithm is presented by Amenta *et al.* [11, 12]. Given a sufficiently dense point sampling from the original surface, the approach guarantees the resulting surface to be topologically correct while interpolating the input samples. Gopi *et al.* [13] introduced a sampling criteria such that the fitted surface is guaranteed to be topologically correct and also provided algorithms that create mesh-based representations of such point sets [14]. Based on Delaunay tetrahedralization of a given point set, Attene *et al.* [15] introduced a method for closed genus-n triangulation fitting provided that the points are sampled from a real object. In 2005, Kuo *et al.* [16] approached the surface fitting problem with a region growing algorithm that gradually adds new triangles to an initial triangulation starting from a seed region of the point set. Dey *et al.* [17] presented a mesh-based surface fitting method applicable to noisy point sets as long as the noise level is within a specified threshold. Many mesh-based

Copyright © 2011 by ASME

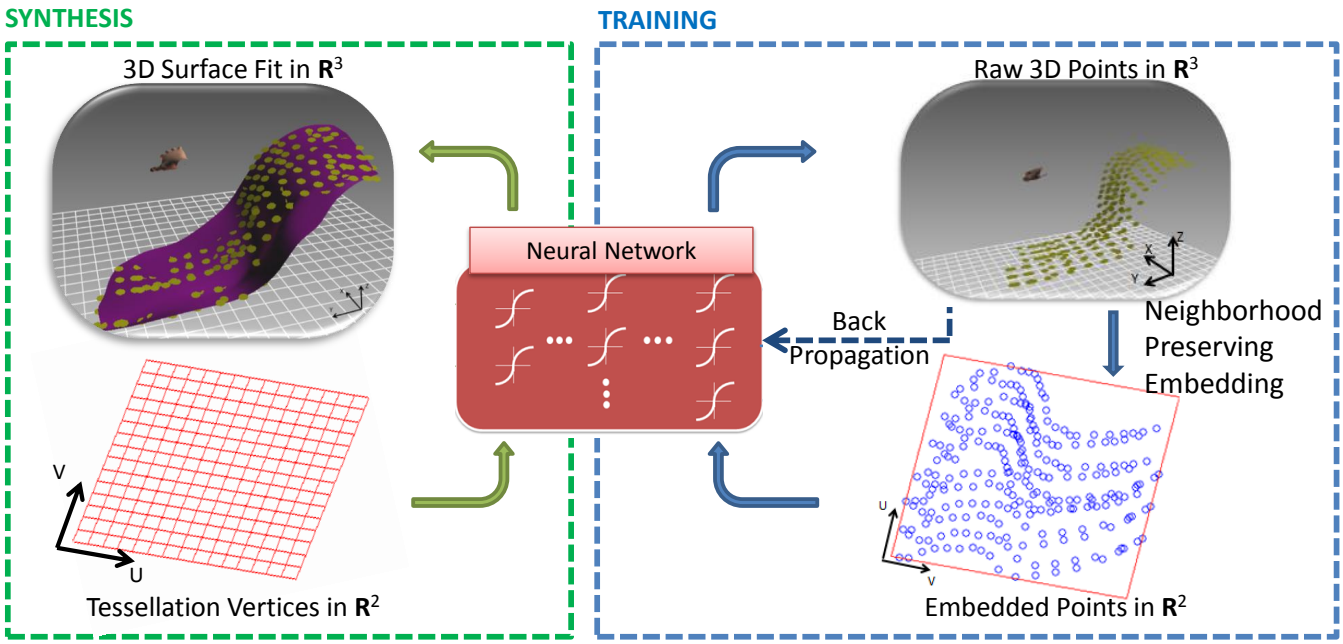

FIGURE 2. Neural network surface regression of unstructured point sets. First, a neighborhood preserving embedding is used for parametrization followed by neural network training. Then the final surface is synthesized by the trained network.

surface fitting algorithms typically require a smoothness or fairness criterion to be minimized which may require considerable post processing after the initial surface fit [18].

Implicit Surfaces: Implicit representations enable compact mathematical descriptions and rapid set operations. However, the tessellation and rendering of such representations is a significant obstacle requiring specialized algorithms for visualization. Juttler *et al.* [19] introduced an approach which results in implicit least squares reconstruction of spline surfaces tailored for reverse engineering. A widely used family of implicit surfaces are the radial basis functions (RBF). In 2004, Kojekine *et al.* [20] employed an octree structure to reduce the computational time associated with RBF spline based volume reconstruction. Ohtake *et al.* [21] used implicit surfaces as a way to facilitate intersection checks on mesh-based geometries. They also employed a similar approach together with compactly supported RBFs for range scanner point cloud surface fitting. Wu *et al.* [22] introduced a combined approach where they use multiple RBFs where individual RBFs construct seed regions that are coalesced into larger regions through a partition of unity functionals. A key drawback of the implicit approaches is the need for specialized visualization mechanisms. Nonetheless, we believe the proposed approach is conceptually similar to RBFs in the way it takes a purely data-driven approach to surface synthesis. The main advantage of the proposed work is in its ability to generate a tessellation directly within the learned mapping function.

Use of Neural Networks: Barhak *et al.* [23] utilized neural network self organizing maps for 2D grid parametrization for surface reconstruction from 3D points sets. The result of the neural network is used to create a 3D surface iteratively with the help of a gradient descent algorithm. Similarly, Galvez *et al.* [24] and He *et al.* [25] utilized neural networks for parametrization and point ordering, rather then surface creation. Khan *et al.* [26] introduced an approach for constructing surfaces from boundary curves that are required to be planar. Their approach addresses the boundary-to-surface learning problem rather than the point-to-surface learning problem. Krause *et al.* [27] implemented a neural gas neural network [28] for approximating a point set with disconnected triangles. These triangles do not necessarily span the whole surface and additional post-processing is required to ensure connectivity and water-tightness of the final surface. The method presented in this paper differs from this approach in that network output in our work is the structured and connected manifold. Additionally, rather than deforming initialized triangles to a given point set similar to mesh-based approaches, the network presented in this paper serves as a direct tessellation engine from which a surface can be synthesized with arbitrary resolution. Other related work [23–26] introduced above do not utilize neural networks for direct surface creation from arbitrary point sets as we do, but rather use the network for point ordering or for surfacing planar boundary curves.

Copyright © 2011 by ASME

OVERVIEW

In this paper, we present a neural network regression based surface creation method from unstructured point sets. Unstructured point sets of interest are sparse, unevenly distributed, can be partially detailed and do not necessarily lie on the manifold they represent. Our methodology consists of two main blocks; training and synthesis (Figure 2). Training consists of both the unique embedding of the original points in the 3D space to the parametric space, and the learning of the mapping between the parametric coordinates and the 3D coordinates. After the mapping is learned, it is used for synthesis in which a fully connected two-manifold surface is created in the 3D space from a tessellation in the parametric space. For this purpose, the tessellation vertices in the parametric space are created with a prescribed density, and are passed through the neural network resulting in the geometry of the 3D surface.

PARAMETRIZATION

Our parametrization approach is invariant to translations and rotations, and can be used for surfaces that fold onto themselves. This capability is achieved using a local neighborhood preserving nonlinear transformation from the 3D space to the 2D parametric space. Two different parametrization methods are used for open and closed manifolds respectively; both of which are detailed in the following paragraphs.

Open Two-Manifold Parametrization

For parameterizing open-manifolds we utilize the locally linear embedding method [29] between $\mathbb{R}^3$ and $\mathbb{R}^2$. A comprehensive review of the method can be found in [29, 30]. The parametrization tailored to our purposes can be summarized as follows:

1. A neighborhood N_i for a fixed number of neighbors is calculated for every $\vec{D}_i$ in $\mathbb{R}^3$ based on the Euclidian distances. $\vec{D}_i$ represents the position vector of point i (Fig. 3).
2. A sparse neighborhood weight matrix W is computed by minimizing Eqn. 1, subject to two constraints; rows of (W) sum to 1, and W_{ij} corresponding to $\vec{D}_j$ that is not in N_i is equal to zero.

$$\theta(W) = \sum_i |\vec{D}_i - \sum_j W_{ij}\vec{D}_j|^2. \quad (1)$$

3. Using W calculated in step 2, an eigenanalysis minimizes Eqn. 2 resulting in the 2D parametric embedding $\vec{P}_i$ of the point $\vec{D}_i$ for all points in the original 3D space.

$$\phi(W) = \sum_i |\vec{P}_i - \sum_j W_{ij}\vec{P}_j|^2. \quad (2)$$

Closed Two-Manifold Parametrization

For parameterizing closed two-manifold surfaces, we utilize the tangential Laplacian minimization introduced by Zwicker *et al.* [31]. This approach aims to uniquely map a closed manifold onto a sphere such that undesired polygon folding is prevented by sliding vertices along local tangent planes. Our closed manifold parametrization process can be summarized as follows:

1. A neighborhood N_i for a fixed number of neighbors is calculated for every $\vec{D}_i$ in $\mathbb{R}^3$ based on the Euclidian distances where $\vec{D}_i$'s are the position vector of input points.
2. An iterative 3D embedding represented by $\vec{P}$ is initialized by setting $\vec{P}_i = \vec{D}_i$ for all i.
3. Adaptive weights are calculated by Eqn. 3, to be used in the Laplacian approximation (Eqn. 4) where $\vec{P}_i$ are the points in the current embedding for all j in N_i.

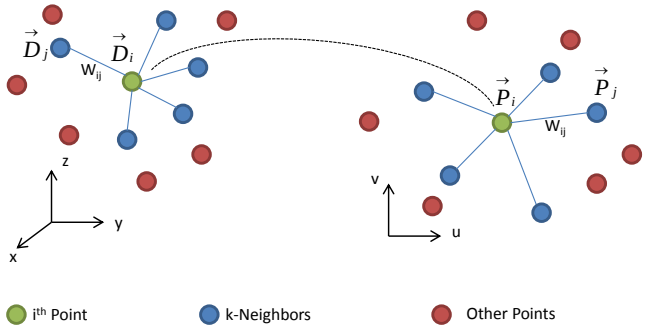

FIGURE 3. Locally linear embedding for open surface parametrization.

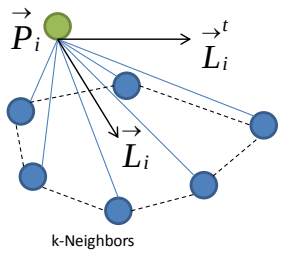

FIGURE 4. Laplacian and tangential Laplacian over k-neighbors

Copyright © 2011 by ASME

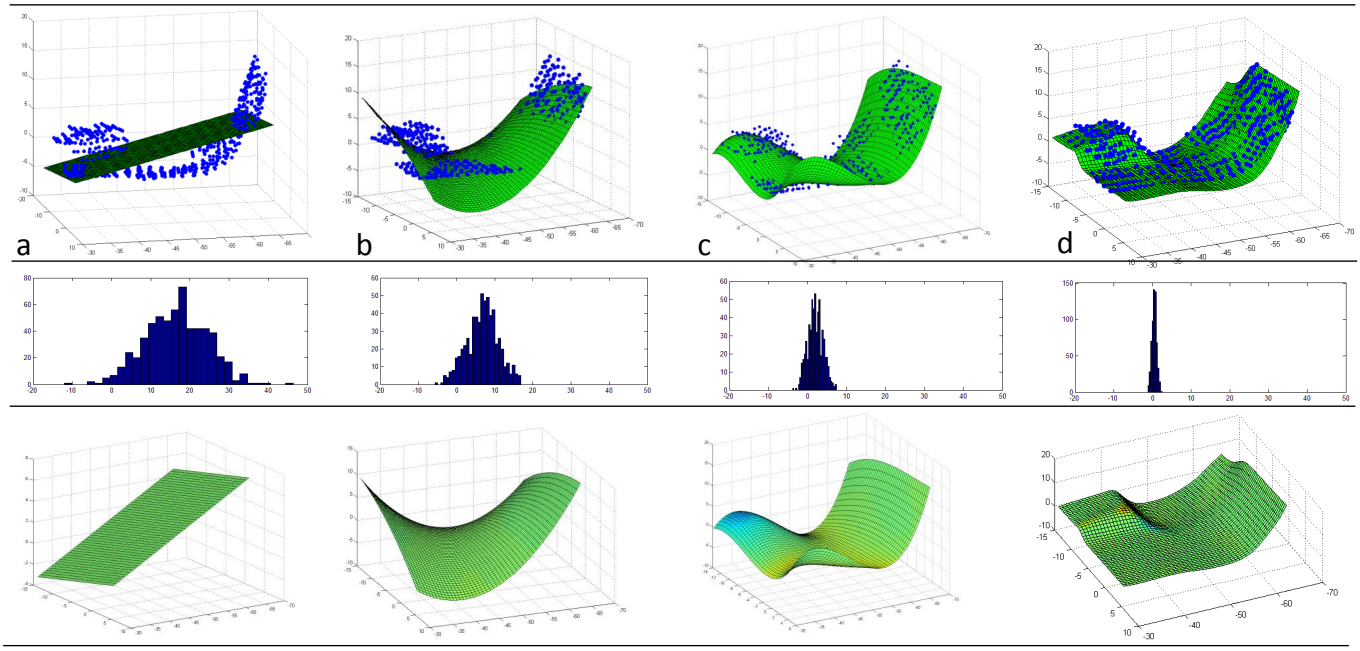

FIGURE 5. Linear(a), quadratic(b), cubic bézier(c) and 3-layer×2-hidden neuron network(d) fits (top), corresponding error distributions (middle) and gaussian curvatures (bottom).

$$W_{ij} = ||\vec{P}_i - \vec{P}_j||/||\vec{D}_i - \vec{D}_j||. \quad (3)$$

4. A discrete Laplacian approximation ($\vec{L}_i$) is calculated by Eqn. 4, where weights (W_{ij}) are given by Eqn. 3 for all j in N_i and zero for the rest.

$$\vec{L}_i = \frac{1}{k}\sum_j W_{ij}(\vec{P}_j - \vec{P}_i). \quad (4)$$

where k is the number of neighbors of $\vec{P}_i$.

5. A normal $\vec{n}_i$ associated with $\vec{P}_i$ is calculated by a principal component analysis (PCA) on N_i. This analysis aims to fit a local plane to the neighbors of $\vec{P}_i$. The smallest eigenvector resulting from the PCA represents the normal to the plane, and is chosen as an approximation to $\vec{n}_i$.
6. An approximate tangential component of the Laplacian, $\vec{L}_i^t$ is calculated by Eqn. 5.(Fig. 4)

$$\vec{L}_i^t = \vec{L}_i - (\vec{L}_i.\vec{n}_i)\vec{n}_i \quad (5)$$

$\vec{L}_i^t$ represents the component of $\vec{L}_i$ that lies parallel to the PCA plane computed in Step 5.

7. $\vec{P}_i$ is updated by $\vec{P}_i \leftarrow \vec{P}_i + \lambda\vec{L}_i^t$ where λ is the damping constant emprically set to 0.5 for all i.
8. Steps 3-8 are iteratively applied until $||L_i{}^t|| < \varepsilon$ for all i.
9. The resulting embedding is projected onto the unit sphere by normalizing all $\vec{P}_i$'s by their magnitudes. The resulting embedding is then conveniently parameterized in $\mathbb{R}^2$ using the inclination and azimuth angles of the spherical coordinate system.

SURFACE REGRESSION

In this section, we present the details of our neural network regression method for freeform surface creation from point sets. Neural networks can be treated as functions that map an input space to an output space (Fig. 6). This general idea can be exploited in a number of ways. For instance, if we use a single neuron in the neural network topology given in Fig. 6, with a linear activation function, mapping from the X,Y input space to the Z output space will result in the planar surface fit to the given points (Fig. 5a). In this scenario, the mapping is from $\mathbb{R}^2$ to $\Re^1$. Similarly, the network can be designed to regress any input space to any output space using a prescribed polynomial or parametric form. Fig. 5 shows various such examples.

Copyright © 2011 by ASME

The Learning Problem

In our approach, the key to constructing a surface on the point set lies in the ability to learn a mapping from the 2D embedded space to the original 3D space. Fig. 6 illustrates the idea.

Here, $\vec{P}$ is the 2D parametric coordinate of the 3D input point $\vec{D}$ as computed using the techniques described in the previous section. Hence, the surface constructing problem boils down to a learning problem from $\mathbb{R}^2$ to $\mathbb{R}^3$.

In this study, we use fully connected, multi-layer, feed-forward neural networks with back-propagation training as the mapping structure. For a single hidden layer with n neurons, with each neuron using a sigmoid activation function, this neural network will map $\vec{P}$ to $\vec{D}$ with Eqn. 6.

$$\{D\}_k = \sigma\left(\sum_{j=1}^{n} w_{kj}\sigma\left(\sum_{i=1}^{2} w_{ji}\{P\}_i + w_{j0}\right) + w_{k0}\right) \quad (6)$$

where

$$\sigma(a) = \frac{1}{1+exp(-a)}. \quad (7)$$

is the sigmoid function.

Neural Network Training

In our approach the numbers of inputs and outputs are fixed at two and three respectively as dictated by the learning problem. The number of hidden layers and the number of neurons in each layer, however, can be chosen as desired. Increasing the number of neurons or the number of hidden layers will result an increased degrees of freedom and non-linearity, which may result in an undesirable over fitting to the data [32]. On the contrary, an insufficient number of neurons and/or layers will produce a stiff map, which may result in under fitting [32]. Therefore, the number of neurons and layers must be chosen judiciously. To this end, we employ an iterative procedure for selecting these parameters as follows:

1. Decompose the available input to training (85%), validation (10%) and test sets (5%).
2. Initialize a network with a single hidden layer ($n_L = 1$) and single neuron ($n_N = 1$) in its hidden layers.
3. Train the network until the validation set performance (P_V) converges.
4. Record the test set performance (P_T) for the current network configuration.
5. Increase the number of hidden neurons by 1 ($n_N \leftarrow n_N + 1$).

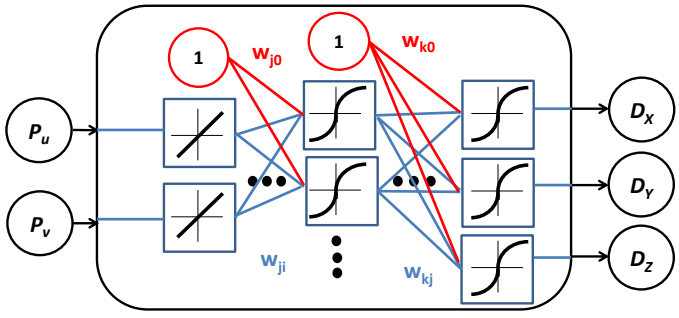

FIGURE 6. Learning problem and the multi-layer feed-forward network.

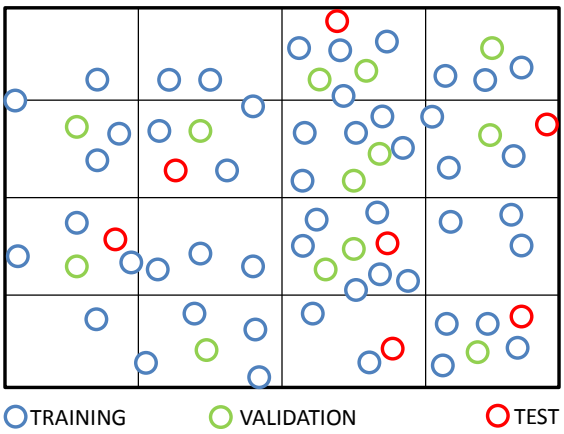

FIGURE 7. Example sampling prior to learning.

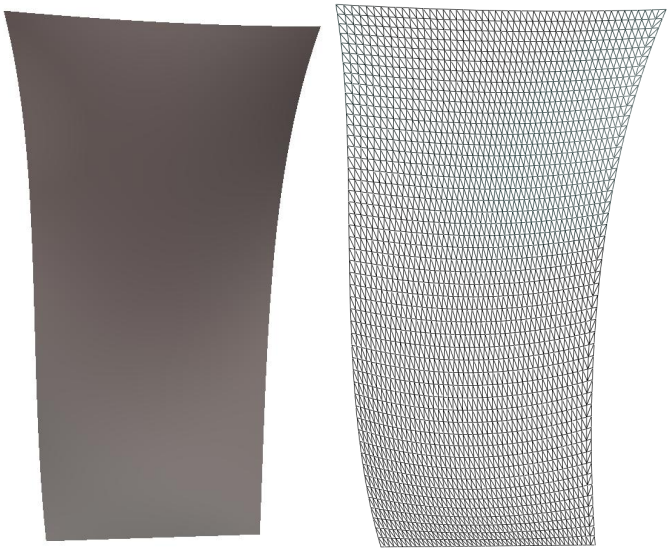

FIGURE 8. Tessellation of the seat back surface.

Copyright © 2011 by ASME

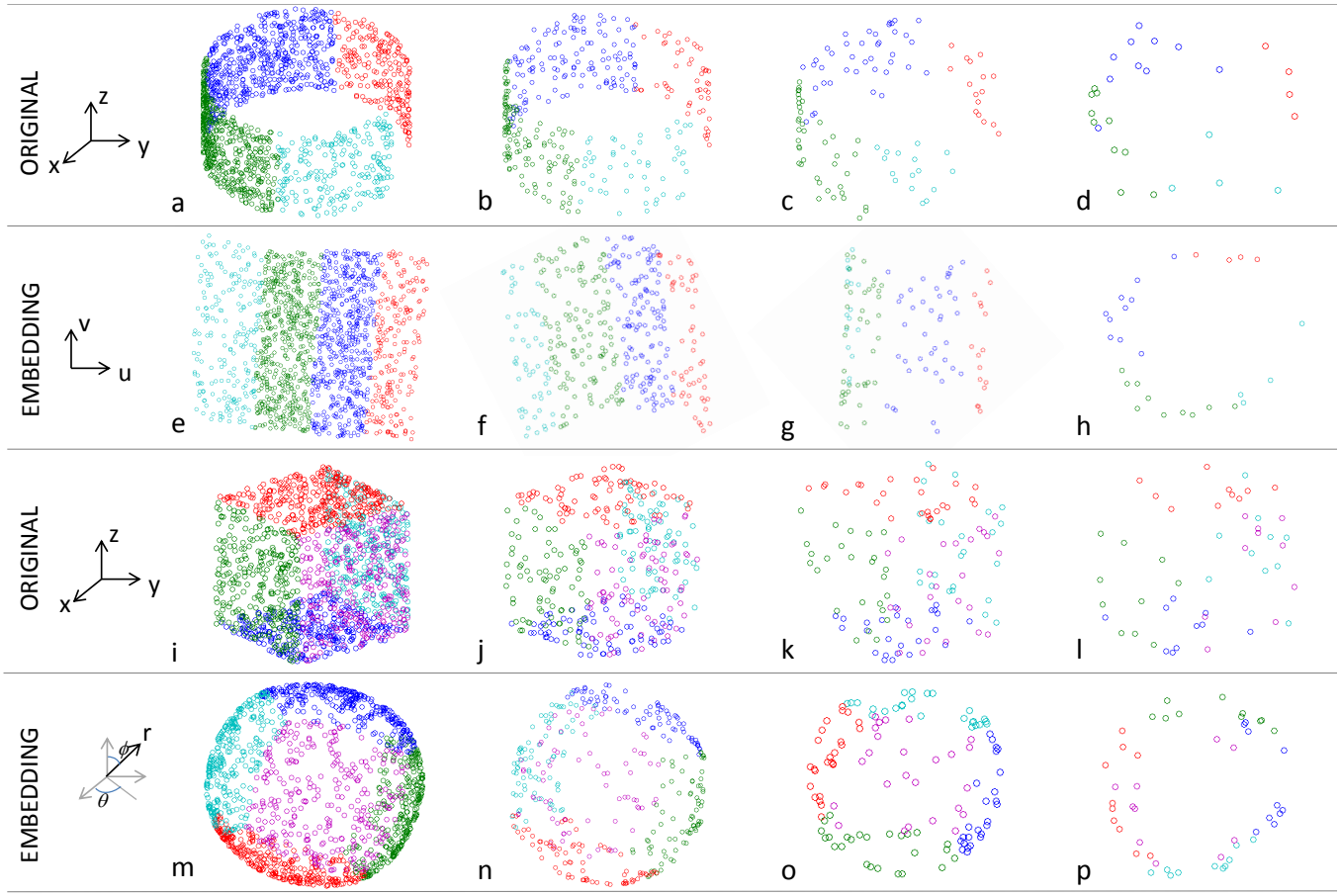

FIGURE 9. Parametrization of an open and closed surface with uniformly sampled random points. Open cylinder data points (a-d) and corresponding parametrization (e-h) for 2000, 500, 150, 38 points respectively. Closed cube data points (i-l) and corresponding spherical parametrization (m-p) for 1800, 450, 150, 60 points respectively. *Note that back side of the cube is not shown for visualization purposes but included in the calculations.*

Iterate steps 3-5 until P_T converges.

6. Record n_N and P_T for current n_L.
7. If $n_L < n_{Lmax}$, increase the number of hidden layers by 1 ($n_L \leftarrow n_L + 1$). Iterate steps 3-7 until $n_L = n_{Lmax}$.
8. Report the network configuration (n_L and n_N) with the best performance P_T on the test set.

Previous works have shown that this iterative search for the network architecture prevents over-fitting and under-fitting [32, 33].

For the above training scheme, we need to sample the validation data set as well as the training data set from the original input points. To ensure an unbiased coverage of the input space, we divide the parametric space into 16 subregions and sample points randomly from a uniform distribution from each subregion. An example is shown in Fig. 7.

Surface Tessellation

After the network is trained on the input data, the parametric space is divided into a uniform grid of user chosen resolution. The vertices in this grid are then fed through the trained network producing the coordinates of the surface in $\mathbb{R}^3$, while sharing the same tessellation topology established in the parametric space. An example surface tessellation is shown in Fig. 8.

RESULTS

Parametric Embedding

As described in the previous sections, we use locally linear embedding for parameterizing open surfaces and tangential Laplacian minimization for parameterizing closed surfaces. In this section we demonstrate the capabilities and limits of these parametrization methods. For this purpose, an open cylinder and a closed cube is utilized for open and closed surface parametrization respectively. Fig. 9 shows the embedding results on these

Copyright © 2011 by ASME

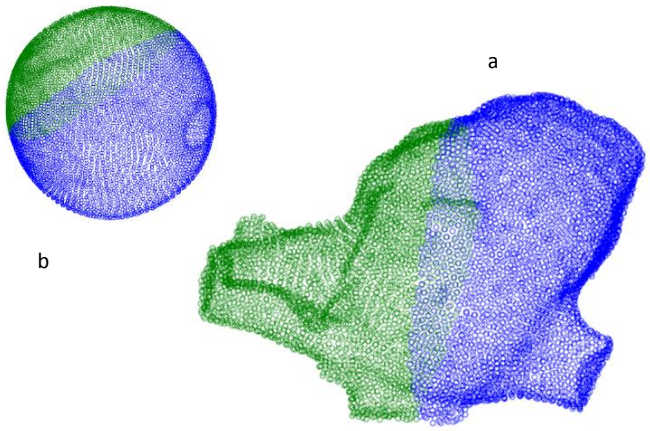

FIGURE 10. Heart point set (a) and corresponding spherical embedding (b). *Colored regions in the original point set match the same in the embedding.*

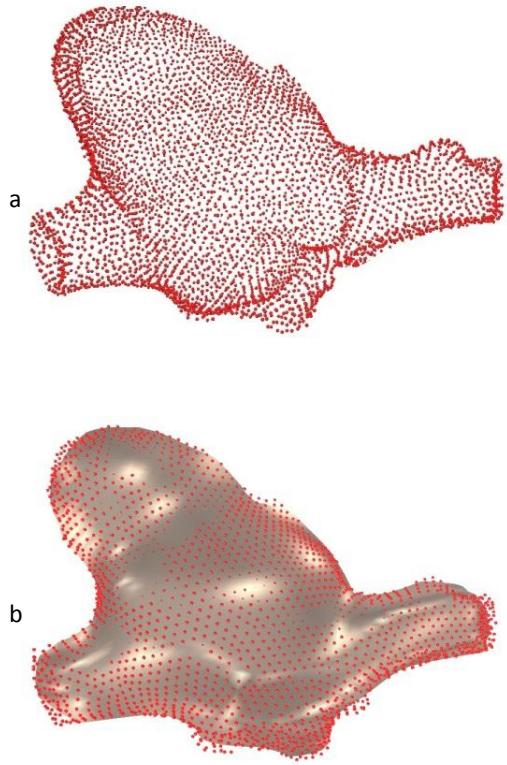

FIGURE 11. Points sampled from heart model set (a), neural network regressed heart surface (b).

two examples. In both cases, the sampling in the original space becomes increasingly sparse to the point that the underlying geometry is no longer discernable (Fig. 9d and l). As shown, the parametric embedding of both examples faithfully capture the intended geometry in the embedded space when the sampling rate is sufficiently high. This capability diminishes with increasing sparsity.

Surfacing of Synthetically Sampled Closed Point Sets

This section demonstrates the surface generation algorithm on a point set that is sampled from an existing heart model. The original point set sampled from the model is shown in Fig. 10a.

The spherical embedding of this point set (Fig. 10b.), shows that the local connectivity is preserved. The final neural network regressed surface is shown in Fig. 11. The topology of the neural network used for regressing this surface has 3 layers with 5 neurons in each layer.

Conceptual Design in Virtual Reality Environment

We have deployed the proposed method to a Virtual Reality (VR) design environment. In this system, a magnetic tracker is attached to the user's hand producing point samples as the user gestures in space. Tracking the user's hand enables the direct dictation of the point set that, in turn, represents the desired surface geometry. The user may sketch multiple strokes in space in any desirable direction to indicate the surface shape. The resulting point sets are used as the input to our regression method, which leads to the final freeform surface geometry.

Fig. 12a shows the complete point cloud drawn by the designer representing the multiple surfaces constituting a car seat. Individual surfaces are demarcated by different colors, and are separated by the user with a keypress during construction. Fig. 12c shows the four point sets forming the front and left sides of the back rest and the seat sections of the seat. Note that they are all open surfaces. Fig. 12d is the corresponding 2D locally linear embedding of these point sets. Fig. 12e are the resulting constructed surfaces. Table 1 shows the number of network layers and neurons as computed from the iterative network topology optimizer.

Fig. 1 shows the neural network regression of the front and back surfaces of the seat. Note that a neural network is trained

TABLE 1. Network degree of freedom.

SURFACE	LAYERS	NEURONS
Front of the Back Rest	2	3
Seat Section	2	2
Side of the Seat Section	3	3
Side of the Back Rest	3	3

Copyright © 2011 by ASME

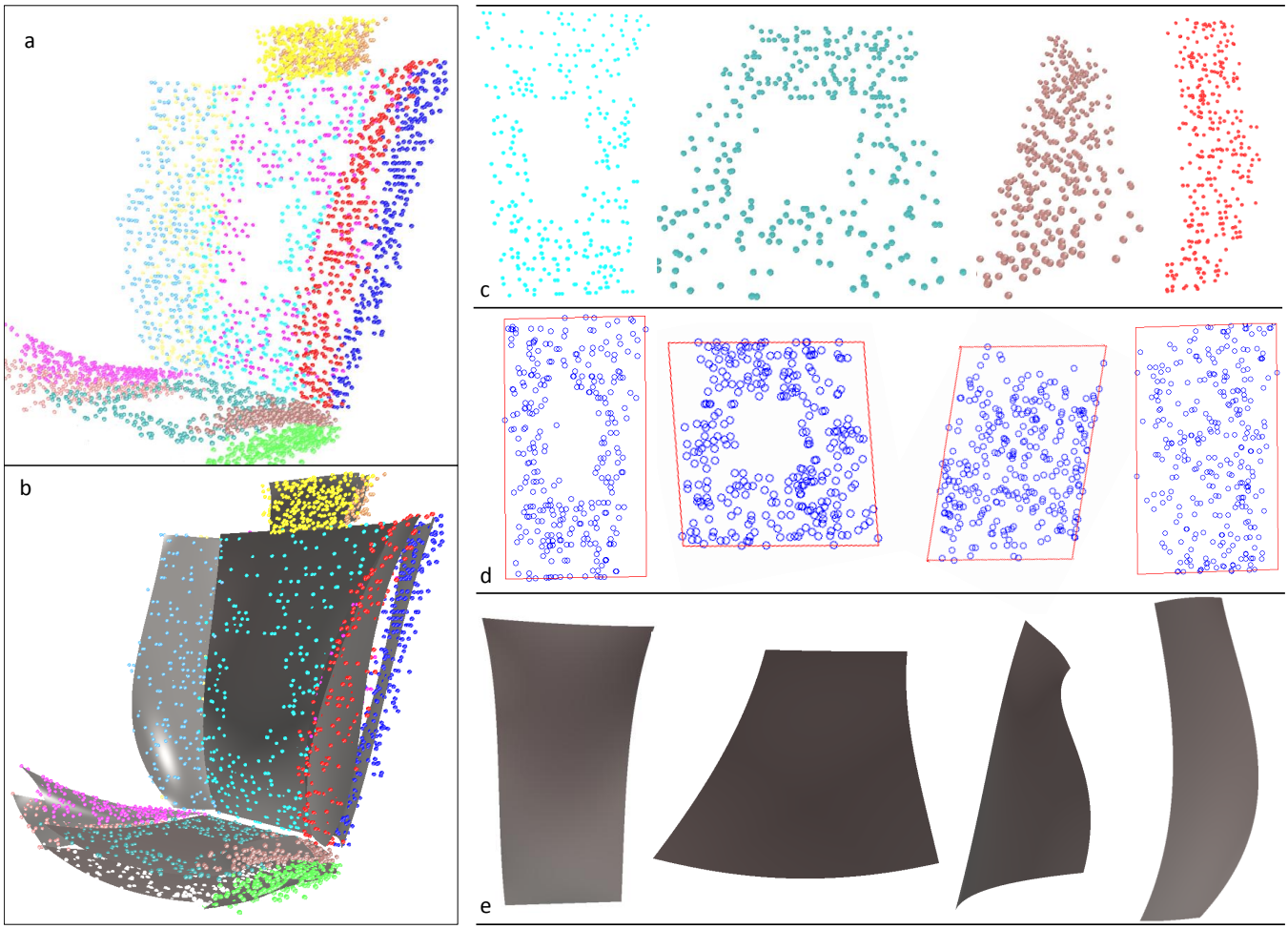

FIGURE 12. Conceptual design of a car seat. Complete point set (a) created by the user in a virtual reality environment where the position and orientation of the user's hand is tracked. Neural network regressed surfaces (b). Point sets for the front and left sides of the back rest and the seat section (c). Corresponding parameterizations (d). Corresponding regressions (e). *For final surfaces after trimming see Figure 1.*

and regressed for each individual point set. Table 2 shows the total network training time for different point sets. In all cases, once the network is trained, the synthesis is near instantaneous.

Since point sets are treated independently, the synthesized surface patches do not form shared boundaries. Currently, we blend and trim these surfaces using an off-the-shelf software.

TABLE 2. Time for parametric embedding and topology optimization. (*On a Intel-i7 1.6 GHz, 4 GB machine.*)

Point Set Size	Parametric Embedding	Topology Opt. & Learning
100	0.3 *sec.*	0.3 *sec.*
1000	2 *sec.*	2 *sec.*
10000	12 *sec.*	12 *sec.*

DISCUSSIONS AND CONCLUSION

We presented a neural network regression method for freeform surface creation on point sets. Point sets of interest are primarily sparse, unstructured, unevenly distributed and can be partially detailed. Our surface regression procedure is composed of three steps; parametrization, neural network training, and synthesis.

On Point Set Parametrization. Two different algorithms are used for parameterizing a given point set. We use locally linear embedding for open two-manifold parametrization, and the tangential Laplacian embedding method for closed surfaces followed by an angular decomposition in spherical coordinates. In Fig. 9, it can be observed that the parametrization is accurate for

Copyright © 2011 by ASME

the point sets in which the underlying geometry is discernable to the human eye, whereas the parametrization deteriorates as the sampling rate becomes prohibitively low. Point sets that are unevenly distributed and/or partially detailed in nature (Figure 12) are also successfully parameterized. These parameterizations are independent of the global position and orientation of the point set and are also able to process surfaces that fold onto themselves.

On Neural Network Training. Once the point set is parameterized, a neural network that takes the parametrization and 3D space coordinates as input-output pairs is trained for surface regression. This training process is controlled by continuously monitoring the validation and test sets over different network structures in order to concurrently minimize both under-fitting and over-fitting. Under-fitting results in a loss of information in which the regressed surface will exhibit fewer details compared to the intended one. Over-fitting, on the other hand, will result in a network structure that has excess degrees of freedom, which typically results in undulations in the synthesized surfaces. Our approach aims to curtail such phenomena through an integrated network topology optimization and learning algorithm. In practice, however, further surface quality control and edits may be necessary that involve direct methods for surface fairing and smoothing. We believe such tools must be an integral component of any geometric design system.

On Surface Tessellation. The trained network forms a bridge between the parametrization domain and the 3D space, thereby effectively serving as a tessellation mechanism. Through an automatic or guided grid generation process in the parametric space, a surface tessellation in 3D can be obtained with a controllable resolution. One such tessellation is shown in Fig. 8. Currently, the proposed algorithm does not facilitate an explicit dictation of positional or derivative constraints which are functionalities necessary for defining explicit surface boundaries or for developing manufacturing procedures. We are currently developing techniques to incorporate such constraints into the design environment through the use of meta-networks that satisfy prescribed constraints through a minimization of energy functionals.

REFERENCES

[1] Yamaguchi, Y., Nakamura, H., and Kimura, F., 1992. "Probabilistic solid modeling: a new approach for handling uncertain shapes". In Selected and Expanded Papers from the IFIP TC5/WG5.2 Working Conference on Geometric Modeling for Product Realization, North-Holland Publishing Co., pp. 95–108.

[2] Horváth, I., 2005. "On some crucial issues of computer support of conceptual design". In *Product Engineering*. Springer Netherlands, pp. 123 – 142.

[3] Horváth, I., Rusák, Z., Vergeest, J., and Kuczogi, G., 2000. "Vague modeling for conceptual design". In TMCE 2000 Conference, pp. 131 – 144.

[4] Gregorski, B., Hamann, B., and Joy, K., 2000. "Reconstruction of B-spline surfaces from scattered data points". In Proceedings Computer Graphics International 2000, IEEE Comput. Soc, pp. 163–170.

[5] Bae, S., 2002. "NURBS surface fitting using orthogonal coordinate transform for rapid product development". *Computer-Aided Design,* ***34***(10), Sept., pp. 683–690.

[6] Pottmann, H., Leopoldseder, S., and Hofer, M., 2002. "Approximation with active B-spline curves and surfaces". In 10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings., IEEE Comput. Soc, pp. 8–25.

[7] Pottmann, H., 2003. "A concept for parametric surface fitting which avoids the parametrization problem". *Computer Aided Geometric Design,* ***20***(6), Sept., pp. 343–362.

[8] Lin, H., 2004. "Constructing iterative non-uniform B-spline curve and surface to fit data points". *Science in China Series F,* ***47***(3), p. 315.

[9] Yin, Z., 2004. "Reverse engineering of a NURBS surface from digitized points subject to boundary conditions". *Computers & Graphics,* ***28***(2), Apr., pp. 207–212.

[10] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W., 1992. "Surface reconstruction from unorganized points". *ACM SIGGRAPH Computer Graphics,* ***26***(2), July, pp. 71–78.

[11] Amenta, N., Bern, M., and Kamvysselis, M., 1998. "A new Voronoi-based surface reconstruction algorithm". In Proceedings of the 25th annual conference on Computer graphics and interactive techniques, ACM, pp. 415–421.

[12] Amenta, N., and Bern, M., 1999. "Surface reconstruction by Voronoi filtering". *Discrete and Computational Geometry,* ***22***(4), pp. 481–504.

[13] Gopi, M., Krishnan, S., and Silva, C., 2000. "Surface Reconstruction based on Lower Dimensional Localized Delaunay Triangulation". *Computer Graphics Forum,* ***19***(3), Sept., pp. 467–478.

[14] Gopi, M., and Krishnan, S., 2000. "A fast and efficient projection-based approach for surface reconstruction". *Proceedings. XV Brazilian Symposium on Computer Graphics and Image Processing,* ***1***(1), pp. 179–186.

[15] Attene, M., and Spagnuolo, M., 2000. "Automatic Surface Reconstruction from Point Sets in Space". *Computer Graphics Forum,* ***19***(3), Sept., pp. 457–465.

[16] Kuo, C., and Yau, H., 2005. "A Delaunay-based region-growing approach to surface reconstruction from unorganized points". *Computer-Aided Design,* ***37***(8), July, pp. 825–835.

[17] Dey, T., and Goswami, S., 2006. "Provable Surface Reconstruction from Noisy Samples". *Computational Geometry,* ***35***(1-2), Aug., pp. 124–141.

[18] Siqueira, M., Xu, D., Gallier, J., Nonato, L. G., Morera,

Copyright © 2011 by ASME

D. M., and Velho, L., 2009. "A new construction of smooth surfaces from triangle meshes using parametric pseudo-manifolds". *Computers & Graphics,* ***33***(3), June, pp. 331–340.

[19] Juttler, B., and Felis, A., 2002. "Least-squares fitting of algebraic spline surfaces". *Advances in Computational Mathematics,* ***17***(1), pp. 135–152.

[20] Kojekine, N., Savchenko, V., and Hagiwara, I., 2004. *Surface reconstruction based on compactly supported radial basis functions*. Citeseer, pp. 218–231.

[21] Ohtake, Y., Belyaev, A., and Seidel, H.-P., 2004. "Ridge-valley lines on meshes via implicit surface fitting". *ACM SIGGRAPH 2004,* ***1***(212), p. 609.

[22] Wu, X., Yu, M., and Xia, W., 2005. "Implicit fitting and smoothing using radial basis functions with partition of unity". In Ninth International Conference on Computer Aided Design and Computer Graphics, IEEE Computer Society, pp. 139–148.

[23] Barhak, J., and Fischer, a., 2001. "Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques". *IEEE Transactions on Visualization and Computer Graphics,* ***7***(1), pp. 1–16.

[24] Gálvez, A., Iglesias, A., Cobo, A., Puig-Pey, J., and Espinola, J., 2007. "Bézier curve and surface fitting of 3D point clouds through genetic algorithms, functional networks and least-squares approximation". In Proceedings of the 2007 international conference on Computational science and Its applications-Volume Part II, Springer-Verlag, pp. 680–693.

[25] He, X., Li, C., Hu, Y., Zhang, R., Yang, S. X., and Mittal, G. S., 2009. "Automatic sequence of 3D point data for surface fitting using neural networks". *Computers & Industrial Engineering,* ***57***(1), Aug., pp. 408–418.

[26] Khan, U., Terchi, A., Lim, S., Wright, D., and Qin, S., 2006. "3D freeform surfaces from planar sketches using neural networks". In Neural Information Processing, Springer, pp. 651–660.

[27] Krause, F., Fischer, a., Gross, N., and Barhak, J., 2003. "Reconstruction of Freeform Objects with Arbitrary Topology Using Neural Networks and Subdivision Techniques". *CIRP Annals - Manufacturing Technology,* ***52***(1), pp. 125–128.

[28] Martinetz, T., and Schulten, K., 1994. "Topology representing networks". *Neural Networks,* ***7***(3), pp. 507–522.

[29] Roweis, S. T., and Saul, L. K., 2000. "Nonlinear dimensionality reduction by locally linear embedding.". *Science (New York, N.Y.),* ***290***(5500), Dec., pp. 2323–6.

[30] Zhang, Z., 2007. "MLLE: Modified locally linear embedding using multiple weights". *Advances in Neural Information Processing Systems*.

[31] Zwicker, M., 2004. "Meshing point clouds using spherical parameterization". In Proc. Eurographics Symp. Point-Based Graphics.

[32] Bishop, C., 2006. *Pattern Recognition and Machine Learning*. Springer.

[33] Heaton, J., 2005. *Introduction to Neural Netowrks with Java*. Heaton Research.

Copyright © 2011 by ASME