International Journal of Database Theory and Application
Vol.7, No.4 (2014), pp. 209-216
http://dx.doi.org/10.14257/ijdta.2014.7.4.16

A Novel Self-Organized Fuzzy Neural Network Surface Reconstruction Algorithm for Point Clouds Without Normal

Liu Yan-ju[1], Liu Yan-zhong[2*], Tao Bai-rui[1], Jiang Jin-gang[3] and Zhang Hong-lie[2]

[1]*Computer Center, Qiqihar University, Qiqihar, China*
[2]*College of Computer and Control Engineering, Qiqihar University, Qiqihar, China*
[3]*College of Mechanical & Power Engineering, Harbin University of Science and Technology, Harbin, China*
**15146692464@163.com*

Abstract

This paper presents a self-organized fuzzy neural network (SOFNN) surface reconstruction algorithm suitable for point clouds without normal. It overcomes the defect of traditional Delaunay triangulation which is difficult to reconstruct point clouds with noises and implicit function which is limited to the number of point clouds and point clouds are required very strict. The SOFNN is based on the fuzzy clustering method optimizing training data before learning fuzzy rules, in order to remove noise data and resolve conflicts in data. The approach not only reduce computational burden of neural network, but also make it easy to fit the surface for point clouds without normal and suitable for mass point clouds. The feature of the SOFNN has dynamic self-organized structure, fast learning speed and flexibility in learning. The experiment results show that is very fine.

Keywords: *surface reconstruction; self-organized fuzzy neural network; normal of point clouds; implicit function*

1. Introduction

Surface reconstruction for point clouds has been applied widely in CAD, medical imaging, archaeology and so on. Many existed algorithms can get shape of the object perfectly, but point clouds are required very strict, especially normal is essential during the reconstruction process. Actually, normal of point clouds is not often accuracy or even is absent.

While surface reconstruction continues to be an important application of computer vision, the estimation of oriented normal field is fundamental to reconstruction as oriented normal provide the first-order approximation and identify outside/inside of the underlying sharp. At present, few literatures have been devoted to reconstruct from point clouds without normal. One possible reason is that normal can be obtained from the laser scanner. Actually, the noises during scanning and discontinuities make normal unreliable or absent for rendering and reconstruction. How to reconstruct the surface of model for point clouds without normal is always a hot problem.

In this paper, we shall proposed Self-Organized Fuzzy Neural Network (SOFNN) to reconstruct the surface for point clouds without normal. These include: off-line extract fuzzy if-then rules from normal of point clouds which is clustered from mass original point clouds, online self-organized structure and parameter learning for unknown information of point clouds.

ISSN: 2005-4270 IJDTA

This paper is organized as follows. We firstly review the related work of surface reconstruction and normal estimation in Sections 2. In Section 3, we give the structure of SOFNN, off-line extract fuzzy if-then rules from typical datum which is clustered from mass original point clouds, online self-organized structure and parameter learning. The Section 4 calculates the normal vector by SOFNN and the procedure of surface reconstruction from point clouds in detail. The results are illustrated feasibility of the proposed approach in Section 5. Conclusions are given in the following.

2. Related Work

Three kinds of surface reconstruction are mainly MC, Delaunay triangulation, and implicit surface. The interested readers are referred to [1] for a survey. Allegre et al. [2] reconstructed the simplified mesh surfaces from large unstructured point clouds to work on dynamic surface that dynamically refined or coarsened the reconstructed surface. Julie et al. [3] developed a scale space strategy for orienting and meshing exactly a raw point. The scale space was based on mean curvature motion that could smooth scale of denture. However, these methods are sensitive to noise. Moreover, inserting thousands of points into triangulation is computationally expensive.

Implicit ones can deal with models of arbitrary topology, blend and perform Boolean operations on surface primitives, and fill holes automatically. This kind of approach has two parts to realize. One is to compute a signed distance function [4]. The other is to represent the reconstructed implicit surface by iso-contour [5] of implicit function. At last, the reconstructed surface can be obtained by solving the linear equations. The gain in vision depends on the normal accuracy of point clouds largely. Radial Basis Function (RBF) is a popular technique of implicit. Samozino [6] used compactly-supported RBFs to achieve local control and reduce computational costs by solving a sparse linear system. But the robustness of method was not good for non-uniform point clouds. Surface reconstruction from raw point clouds could be cast as a spatial Poisson problem in [7]. The Poisson formulation considered all the points without resorting to heuristic spatial [8] partitioning or blending and was therefore highly resilient to data noise. But Poisson needed to solve global spare linear equations and requested the accuracy of normal of point clouds.

The normal at the sample point is the furthest Voronoi vertex in the point's Voronoi cell, named Voronoi Pole. Dey *et al.*, [9] presented a provable surface reconstruction, which extend the idea of Voronoi Pole. However, Delaunay method is more sensitive to noise than other method. Voronoi Pole and PCA are consolidated and can achieve better effects for noise. Because it must compute for the same data twice, the approach costs CPU much time. The implicit surface is suitable to describe reconstructed surface during the process of volume visualization. When the representation equation of implicit surface is determined, the common method is radical basis function (RBF) [10] which has higher precision and stability to solve interpolating scatter data for tasks. Many methods for deriving normal information from unorganized points take the strategy of estimating a normal field and trying to propagate the normal of a seed point that must be given in advance over the model to align the individual normal vectors [11]. Jalba *et al.*, [12] proposed a method that it attracted the evolving surface towards the data points in a velocity field generated by Coulomb potentials. But surface features which are smaller than the smallest grid cells are not accurately reconstructed.

Recently, hybrid control laws containing Neural Networks (NNs) have attracted more and more attention. NNs were used to adjust and optimize parameters of fuzzy conductor through offline or online learning and control performance was demonstrated

to be good. Takatoshi *et al.*, proposed a model for the design of Fuzzy Inference Neural Network (FINN)[13]. In [14], Meng Joo Er *et al.*, presented a D-FNNs controller suitable for industrial applications. In [15], nonlinear systems controller design is proposed. All of these methods proposed new adding, pruning techniques and a recursive learning algorithm, respectively. However, the methods mentioned above are limited in optimal structure and parameters of NNs and need too computational burden to real-time control in practice proceedings.

3. SOFNN Structure and Learning Algorithm

The architecture of SOFNN is shown in Figure 1 with six layers. This structure is similar to Gang's[15] except for the clustering data in the second layer.

The layer operation of the SOFNN is described briefly as follows. Each neuron $x_k = (x_{k1}, x_{k2}, y_k^*)$, k=1, 2,…,n is an input variable. For given n training data x_k, using FCM [16] process, we can obtain c(<n) cluster centers as $V_i = (v_{i1}, v_{i2}, y_i^*), i = 1,2,..., c$. V_i can be regarded as the typical data extracted from the original training data and is inputted into the EBF layer. Each neuron of the EBF layer is a T-norm of Gaussian fuzzy membership functions, which is employed in [15].

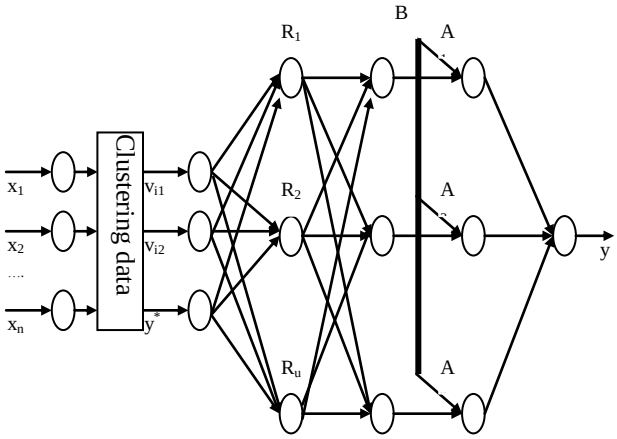


Figure 1. SOFNN Structure

The learning process of the SOFNN includes off-line learning phase and on-line learning phase. In the following we explain each learning phase.

3.1. Off-line Learning

Consider the Fuzzy C-Mean (FCM) clustering algorithm with following functions. First, it resolves conflicts in the original point clouds due to the reasonable distribution of the typical data. Second, it removes the noise of data from the original data, because there is not be a cluster centre around any redundant datum. Third, it improves the computation size by reducing the number of the training data. Now we summarized FCM as follows [16].

It minimizes the following objective function with respect to fuzzy memberships $\mu_{ki}(x_k)$ and cluster centers v_i:

$$J_s(U, v_1, v_2, ..., v_c) = \sum_{k=1}^{n} \sum_{i=1}^{c} (\mu_{ki})^s \left\| x_k - v_i \right\|^2 \quad (1)$$

where $\left\| x_k - v_i \right\|_a^2 = (x_k - v_i) A (x_k - v_i)^T$, A is a positive definite matrix, $s(1 < s < \infty)$ means a smoothing weight.

(a) Initialize parameters, $c(2 \le c < n)$ and $s(1 < s < \infty)$ and $U(0)$ of fuzzy partitions randomly.

(b) Compute the cluster center v_i, for i=1,…,c, by using U(t)

$$v_i = \sum_{k=1}^{n} (\mu_{ki})^s x_k \Big/ \sum_{k=1}^{n} (\mu_{ki})^s \quad (2)$$

(c) Update $U(t)$ in the following: Calculate the sets I_k and I_k^* (k=1,2,…,n) as

$$I_k = \{ i \mid 1 \le i \le c, d_{ki} = \left\| x_k - v_i \right\| = 0 \} \quad (3)$$

$$I_k^* = \{1, 2, ..., c\} - I_k \quad (4)$$

if $I_k = \phi$ then $\mu_{ki} = 1 \Big/ \sum_{j=1}^{c} (d_{ki} / d_{kj})^{2/(s-1)}$ else

$$\mu_{ki} = 0, \forall i \in I_k^*, \quad \sum_{i \in I_k} \mu_{ki} = 1 \quad (5)$$

(d) Repeat (b) and (c) until $||U(t)-U(t-1)|| < \varepsilon$. ε is a positive and small enough real number, which is a terminal criterion. The matrix norm is taken as [9]

$$\left\| U(t) - U(t+1) \right\| = \max_{k,i} \{ \left| \mu_{ki}(t) - \mu_{ki}(t+1) \right| \} \quad (6)$$

Using FCM process, we can obtain c(<n) cluster centers. Then let c clusters be practical training data for the neuro-fuzzy learning process and the fuzzy rule base consists of a collection of fuzzy IF-THEN rules:

R^j: if x_{knn} is A_1^j and x_{cur} is A_2^j, then y is B_m

where $x = [x_{knn}, x_{cur}]^T$ and y are input vectors and output value of FNN, respectively. A_i^j (i=1,2,..,n) and B_k (k=1,2,…,m) are linguistic variables of fuzzy sets in subspace U_i and V_k described by their membership functions $\mu_{A_i^j}(x_i)$ and $\mu_{B_k^j}(y_k)$, j=1,2,…,M. M is total number of the fuzzy rules. By using product inference, singleton-fuzzifier, and center-average defuzzifier strategies, output of FNN can be expressed as

$$y(x) = \hat{\Omega}(x \mid \hat{\Theta}) = \frac{\sum_{j=1}^{M} \bar{y}^j (\prod_{i=1}^{n} \mu_{A_i^j}(x_i))}{\sum_{j=1}^{M} (\prod_{i=1}^{n} \mu_{A_i^j}(x_i))} = \hat{\Theta} \xi(x) \quad (7)$$

where $\mu_{A_i^j}(x_i)$ is membership function value of fuzzy variable x_i, $\bar{y}^j$ is the point at which $\mu_{B_k^j}(y_k)$ achieves its maximum value, and it is assumed that $\mu_{B_k^j}(\bar{y}^j) = 1$.

$$\hat{\Theta} = \begin{pmatrix} \bar{y}_1^1 & \bar{y}_1^2 & \cdots & \bar{y}_1^M \\ \bar{y}_2^1 & \bar{y}_2^2 & \cdots & \bar{y}_2^M \\ \vdots & \vdots & \ddots & \vdots \\ \bar{y}_m^1 & \bar{y}_m^2 & \cdots & \bar{y}_m^M \end{pmatrix} \tag{8}$$

Eq. (7) is adjustable parameter matrix and $\xi(x) = [\xi^1(x), \xi^2(x),..., \xi^M(x)]^T$ is fuzzy basis function vector, in which $\xi^j(x)$ is defined as

$$\xi^j(x) = \frac{\prod_{i=1}^{n} \mu_{A_i^j}(x_i)}{\sum_{j=1}^{M} \prod_{i=1}^{n} \mu_{A_i^j}(x_i)} (j = 1,2,..., M) \tag{9}$$

Generation of these rules is the same as on-line learning algorithm.

3.2. On-line learning

The on-line learning process of the SOFNN includes the structure learning and the parameter learning.

The structure learning is consisted of generation and combination of neurons. There are two criteria to judge whether to generate an EBF neuron or not, that is system error and accommodation boundary. Consider the *n*th observation (x_n, y_n^*), where x_n is the input vector and y_n^* is the desired output, the output of the network with the current structure is y_n. The system error *E* is the Euclidean distance between the vectors y_n and y_n^*. If *E* is larger than predefined error tolerance, a new EBF neuron should be considered. The accommodation boundary is an EBF neuron of representations over a region defined in the input space. For the *n*th observation (x_n, y_n^*), calculate the Euclidean distance *d(j)* between the observed x_n and the center vector of the *j*th EBF c_{ij}. Find the minimum distance $d_{\min}$ among the *d(j)*. If it is larger than a predefined value, a new neural unit should be considered. If (*j+1*)th EBF neuron is generated, its parameters is allocated with

$$c_{i(j+1)} = x_n, \sigma_{i(j+1)} = k \times d_{\min} \tag{10}$$

where $k \in [1.06, 1.2]$ is a positive constant weight Φ_{j+1} can be determined by

$$\phi_{j+1} = \exp[-\sum_{i=1}^{j+1} \frac{(x_{ij+1} - c_{ij+1})^2}{2\sigma_{ij+1}^2}], \ j = 1,2,..u. \tag{11}$$

In the stage of combination neuron, if MFs are similar to each other, they can be combined into one new MF as [8]. In the stage of adjustment parameters, we apply to a gradient descent method [7]. In this way, we can reserve points in the larger curvature of area and express more complete detail. Therefore the least points be used, the most effective expression can be obtained. The detail is given in the algorithm 1.

Algorithm 1: Normal of point clouds estimation by SOFNN

Step 1. Input sample point clouds;

Step 2.Select θ , divide 3D space into grid,

Establish the topology relationships between sample points and grid;

Step 3. If the numbers of points in real grid>=4

To fit plane and quadric using points in grid and estimate normal vectors $\vec{n}$ and curvatures;
Else goto Step 5;
Step 4. Calculate centroid coordinates of points in real grid and the distance $|p_1 p_2|$;
If $|p_1 p_2| < \lambda$
Remain the point closest to the *p*'s centroid coordinates;
Else goto Step 7;
Step 5. Add up the number of the points in the 20 neighbor grids;
If the totals of point<10
Remove these point as noise;
Else fit plane and quadric, estimate normal vectors $\vec{n}$ and curvatures,
Calculate centroid coordinates of points
Remain the point closest to the *p*'s centroid coordinates;
Step 6. If there are real grid can not be sampled goto Step 3;
Else goto Step 8;
Step 7. Grid was divided by Octree subdivision, goto Step 3;
Step 8. End.

4. Implicit Surface Reconstruction

The simple greedy algorithm is proposed to progressive approximating local implicit surface f_i according to the expected precision ε . It improves multi-level partition of unit.

At the beginning, there is only one point rough set C_0 which is regarded as the center of RBF to compute f_i. By estimating f_i, the rest point is computed in the rough set, which is also the condition of loop. When the number of rest point is less than precision ε , the loop stops. Otherwise, the rest point clouds are appended to the new queue C_i as the center of RBF. The detail of algorithm is described as follows.

Restaurant

Algorithm 2: Reconstruct Local Progressive Surface(RLPS)

Step 1. If f_i is not existing
{
Initial set C_0 and compute f_i
Estimate the rest r_j in point set C_i
}
Step 2. While ((max$|r_i|$)>ε &&size(C_i)<δ)
{
Generate C_{i+1}, by appending the max rest points to new queue
Recompute f_i by equation(3)
Update the rest r_i
}
Step 3. If max($|r_i|$)>ε
For each unit point data, reconstruct RLPS(C_i, ε , δ)

This algorithm is similar to unit partition in which there exist partition space of input point clouds loop and approximate local RBF surface. The difference is the procedure of refinement in step 3.

5. Experimental Results

The proposed framework is tested and validated by reconstructing two models. Experimental results presented in this section are generated on a PC equipped with an Intel Core 2 processor at 2.93GHz and 2GB main memory. The input data set including cutting and molar is from 3D laser scanner. Original point clouds are inputted the SOFNN. Then the normal of point clouds is extracted by SOFNN and compare with traditional algorithm, respectively. In Figure 2, we can see the number of offset points used traditional method increases twice than that of raw point clouds of this algorithm. Point clouds in this algorithm increase obviously slowly because the number of offset point has relation with the number of unit rather than the number of raw point clouds in this algorithm.

Next, two kinds of point clouds are reconstructed by variational implicit surfaces and progressive one and refined until desired accuracy 6×10^{-4} is achieved in Figure 3, respectively. From the models it can be illustrated that our algorithm is as similar as the traditional one. Actually, this algorithm saves much time to reconstruct than traditional one in Table 1. The number of offset points in this algorithm is decreased greatly.

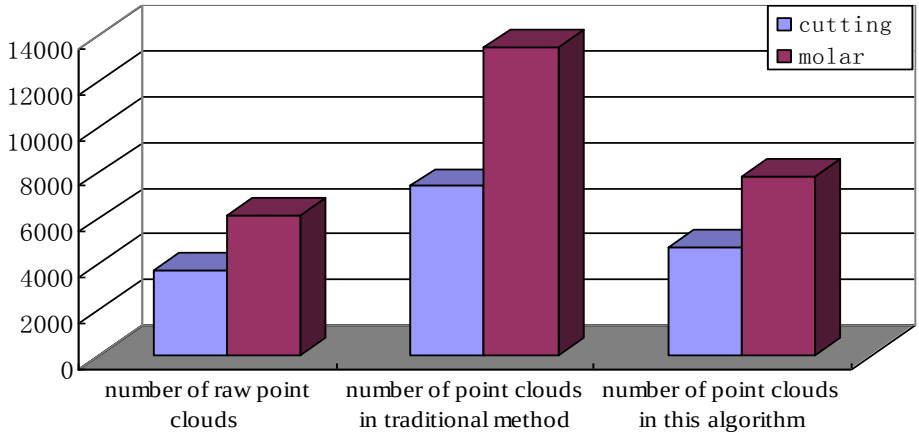
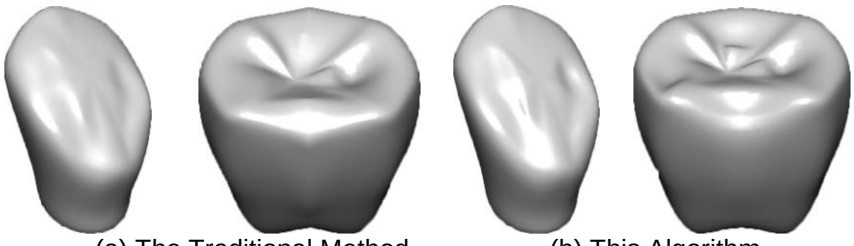


Figure 2. The Number of Points Adding Offset Points by Two Methods



(a) The Traditional Method (b) This Algorithm

Figure 3. Surface Reconstruction for Cutting and Molar

Table 1. Comparison of Running Time (s) between the Traditional and this Algorithm on the Two Kinds of Point Clouds of Denture

Model	Number size	Traditional method	This algorithm
Cutting	3714	0.7869	0.3073
Molar	6179	0.8973	04875

6. Conclusions

The SOFNN surface reconstruction from mass point clouds is proposed. A key advantage of the scheme is the structure of SOFNN for estimation the normal of point clouds. By off-line learning phase and on-line learning phase, the normal can be estimated accuracy. The global point clouds are divided into several smaller sub-domains and the problem of reconstruction is simplified. The sub-surface is fitted by progressive implicit function and the basis function is selected in each sub-domain. The overall surface is formed after repeatedly inferring normal along the sub-surfaces. In the future, this algorithm should be improved further in the selection of offset point in order to apply any arbitrary models.

Acknowledgment

This work was supported by National Natural Science Foundation of China No. 51205093 and Natural Science Foundation of Heilongjiang Province of China No. F201336 and the Education Department of Heilongjiang Province of China, No. 12531765.

References

[1] Y. Zhang, Y. Liu and X. Dai, "Categories and New Developments of Three-dimensional Reconstruction Using Triangulation Mesh [J]", ICIC Express Letters, vol. 5, no. 1, (**2011**), pp. 95-100.
[2] R. Allegre and R. Chaine, "A dynamic surface reconstruction framework for large unstructured point sets [J]", IEEE/Eurographics Symposium on Point-Based Graphics, (**2006**), pp. 17–26.
[3] J. L. Digne, J. M. More, C. M. Souzani and C. Lartigue, *et al."* Scale space meshing of raw data point sets [J]. Comput. Graph. Forum, vol. 30, no. 6, (**2011**), pp. 1630–1642.
[4] H. Avron, A. Sharf, C. Greif and D. Cohen-Or. "ℓ $_1$-sparse reconstruction of sharp point set surfaces [J]", ACM Trans. Graph, vol. 29, no. 5, (**2010**), pp. 1–12.
[5] H. Huang, D. Li, H. Zhang, U. Ascher and D. Cohen-Or. "Consolidation of unorganized point clouds for surface reconstruction [J]", ACM Trans. Graph. (TOG), vol. 28, no. 5, (**2009**), pp. 1–7.
[6] M. Samozino, M. Alexa, P. Alliez and M. Yvinec, "Reconstruction with Voronoi centered Radial Basis Functions [C]", Eurographics Symposium on Geometry Processing, (**2006**), pp. 1-10.
[7] F. X. Dupe and J. M. Fadili, "A proximal iteration for deconvolving poison noisy images using sparse representations [J]", IEEE Trans. Image Processing, vol. 18, no. 2, (**2009**), pp. 310–321.
[8] M. A. T. Figueiredo and I. S. Tecnico, "Restoration of poissonian images using alternating direction optimization [J]", Image Processing, IEEE Transactions on, vol. 9, no. 2, (**2011**), pp. 3133-3145.
[9] T. K. Dey, R. Dyer and L. Wang, "Localized Cocone Surface Reconstruction [J]", Computer and Graphics, vol. 35, (**2011**), pp. 483-491.
[10] I. Tobor, P. Reuter and C. Schlick, "Efficient Reconstruction of Large Scattered Geometric Datasets Using the Partition of Unity and Radial Basis Functions [C]", In Proceedings of WSCG, vol. 12, no. 3, (**2004**), pp. 467-474.
[11] G. Guennebaud and M. Gross, "Algebraic Point Set Surfaces [J]", ACM Transactions on Graphics (SIGGRAPH 2007 Proceedings), vol. 26, no. 3, (**2007**), pp. 23.1-23.9.
[12] A. C. Jalba and J. B. T. M. Roerdink, "Efficient Surface Reconstruction Using Generalized Coulomb Potentials [J], IEEE Transactions on Visualization and Computer Graphics, (**2007**), vol. 13, no. 6, pp. 1512-1519.
[13] T. Nishina and M. Hagiwara, "Fuzzy inference neural network [J]", Neurocomputing, (**1997**), vol. 14, pp. 223 - 239.
[14] M. J. Er, C. B. Low and K. H. Nah, "Real-time implementation of a dynamic fuzzy neural networks controller for an ACARA [J]", Microprocessors and Microsystems, (**2002**), vol. 26, pp. 449 –461.
[15] G. Leng, G. Prasad, T. M. McGinnity, "An on-line algorithm for creating self-organizing fuzzy neural networks [J]", Neural networks, vol. 17, (**2004**), pp. 1477 - 1493.
[16] L. Bai, J. Liang, C. Dang and F. Cao, "A novel fuzzy clustering algorithm with between-cluster information for categorical data [J]", Fuzzy Sets and Systems, vol. 215, no. 16, (**2013**), pp. 55-73.

