Journal for Geometry and Graphics
Volume 2 (1998), No. 1, 1–6

Free-form Surfaces for Scattered Data by Neural Networks

Miklós Hoffmann, Lajos Várady

Institute of Mathematics and Informatics, Lajos Kossuth University
P.O.Box 12. H-4010 Debrecen, Hungary
email: hofi@math.klte.hu

Abstract. The handling of scattered spatial points is an important question in computer graphics and there are several methods to construct surfaces from these type of data. The aim of our paper is to present a new method which produces standard free-form surfaces from the scattered data. Earlier methods normally construct a triangular control grid from the data however NURBS or Bézier surfaces originally use quadrilateral control grid. In this paper a new approach is presented, where first an artificial neural network is used to order the data and form a grid of control vertices with quadrilateral topology, hence after this step the original, well-known free-form methods can be applied to construct the surface. Another advantage of this method that it can handle arbitrary set of points as well as very few data.

1. Introduction

There are several applications of approximation and interpolaton of scattered data in computer graphics and CAD/CAM, hence there are numerous different methods to solve this problem, see e.g. [1]-[4]. Some of them use free-form surfaces which require ordered input points, hence the basic step in these methods is to prepare a triangular grid from the points, mainly from large amount of data.

In our new method one of the two main purposes was to produce a quadrilateral grid (instead of the triangular one), because the standard B-spline, Bézier or NURBS surfaces use such a grid as input data. On the other hand, our aim was to try the advantage and effectiveness of the artificial neural networks. There are theoretical results of this latter topic [8], while the application of neural networks in free-form modelling is an interesting topic of research projects, see e.g. [9]. Here the Kohonen network is used, which has a strong ability of learning data. This means that during a so called training procedure a topologically invariant grid is moved towards the scattered points. The expression ”training” will be discussed later, since this is the essential point of the neural networks as well as our whole method. One of the most important facility of the Kohonen network is to keep the original topology of the predefined grid, hence if a quadrilateral (or any other type of) grid has been defined, it has the same topology after the training procedure. The final result is a method which can handle arbitrary set of scattered points as well as very few data, and the grid can have arbitrary topology, however here the quadrilateral grid was desired.

Hence our approximation procedure consists of two separated parts:

1. the neural network learns the scattered points and prepares a quadrilateral grid
2. one of the standard surface approximation or interpolation methods (here the NURBS method) is applied to form the surface using the grid as input data.

First of all we give a short introduction to the theory of the Kohonen neural networks. After these paragraphs the application of this method for our special problem will be discussed. The final result is demonstrated by a NURBS surface approximating 3D scattered points.

2. Artificial neural networks

Artficial neural networks are biologically inspired models, based on the functions and structure of biological neurons. A neural network consists of numerous computational elements (neurons or nodes), highly interconnected to each other. A weight is associated to every connection. Normally nodes are arranged into layers. During a training procedure input vectors are presented to the input layer with or without specifying the desired output. According to this difference neural networks can be classified as supervised or unsupervised (self-organizing) neural nets. Networks can also be classified according to the input values (binary or continuous). The learning procedure itself contains three main steps, the presentation of the input sample, the calculation of the output and the modification of the weights by specified training rules. These steps are repeated several times, until the network is said to be trained. For details and survey of artificial neural networks see e.g. [5]-[6].

3. The Kohonen network and the training procedure

The Kohonen net is a two-layered, unsupervised, continuous valued network. The great advantage of this network, which will be used in this problem, is the ability of fast organization of any number of unordered points. The training procedure will result in a topology-preserving grid following the structure of the input points.

Here the number of input neurons is three, since the network will be trained by the coordinates of the 3D input points. The output neurons form a quadrilateral grid, and this topology will be preserved during the whole procedure. All the output nodes are connected to each input node and a weight is associated to every connection. Hence a three dimensional weight vector is assigned to each output node.

Now consider these weights as the spatial coordinates of points of the grid. During the training process the weights will be changing, hence this grid will move slowly in the three dimensional space toward the input points, meanwhile the topology of the grid will remain the same. A short description of the training is the following: one of the scattered input points is selected randomly to be the input vector of the net. A winning unit is determined by the minimum Euclidean distance of this point to the output nodes. The node with the minimum distance is the winning unit. Around this node a neighborhood of output points is determined according to the topology of the grid (this neighborhood decreases in time). Finally the weights of the nodes in this neighborhood are updated, i.e. change slightly toward the value of the input vector. After updating the weights in the neighborhood, a new input vector is presented and the above steps are repeated. For a more detailed description of the Kohonen network see [7].

4. Training of the net by scattered data

In a standard example mentioned also by Kohonen the network is trained by points selected from a uniformly distributed area [7]. A main difference between this kind of applications and our problem is that we normally have finite number of points and they are in the 3D space, while the grid remains two dimensional. Now we give the exact algorithm of the applied Kohonen network, which produces a rectangular grid onto the 3D scattered points.

Let a set of points $p_i(x_{1i}, x_{2i}, x_{3i}) \quad (i = 1, \ldots, n)$ be given. The coordinates of these points will form the input vectors of the net. The net itself contains two layers: the input layer consists of three nodes and the output layer consists of m nodes. The number m depends on the number of input vectors, generally $m = 4 \times n$ is used, where n is the number of input points. However if the number of input points is large, or the input is given by a distribution, then m can be a convenient number independently of the input points. These m output nodes form a grid with arbitrary, but predefined topology, which is quadrilateral in our case. Each node of the output layer connected to each of the nodes of the input layer. Every connection has a weight: w_{ij} denotes the weight between the input node i and the output node j.

- Coordinates of the scattered points: $p_i(x_{1i}, x_{2i}, x_{3i}) \quad (i = 1, \ldots, n)$
- Coordinates of the output points: $q_j(w_{1j}, w_{2j}, w_{3j}) \quad (j = 1, \ldots, m)$
- STEP 1. Initialize the weights $w_{sj}, \quad (s = 1, 2, 3 \quad j = 1, \ldots, m)$ as small random values around the average of the coordinates of the input points. Let the training time $t = 1$
- STEP 2. Present new input values $(x_{1i_0}, x_{2i_0}, x_{3i_0})$, as the coordinates of a randomly selected input point p_{i_0}
- STEP 3. Compute the Euclidean distance of all output nodes to the input point:

$$d_j = \sum_{s=1}^{3} (x_{si_0} - w_{sj})^2$$

- STEP 4. Find the winning unit q_{j_0} as the node which has the minimum distance to the input point, so where j_0 is the value for which $d_{j_0} = min(d_j)$
- STEP 5. Compute the neighborhood $N(t) = (j_0, j_1, \ldots, j_k)$
- STEP 6. Update the weights (i.e.the coordinates) of the nodes in the neighborhood by the following equation:

$$w_{sj}(t+1) = w_{sj}(t) + \eta(t)(x_{si_0} - w_{sj}(t)) \quad \forall j \in N(t)$$

  where $\eta(t)$ is a gain term decreasing in time.
- STEP 7. Let $t = t + 1$. Repeat STEP 2–7 until the network is trained.

If the number of input points is relatively small, then the network is said to be trained if all the input points are on the grid. If we have hundreds of input points, or data given by a distribution, as in some of the scattered data problems, then this requirement would yield long computing time. In this case the network is said to be trained if the changes of the grid is under a certain predefined limit. Hence, contrary to most of the other methods, the proposed procedure can handle a distribution as well as a limited number of spatial points.

The network produces a quadrilateral grid. This particular topology, however, depends only on the original connections of the output nodes. If a triangular or any other kind of grid would be desired, the nodes can be connected accordingly. Now we examine some properties of the network which influence the efficiency of the training procedure.

4.1. The radius and the gain term

The radius of the neighborhood and the gain term are important factors of computing time, see [10]. Both terms have to decrease in time, but in the theory of Kohonen networks there is no general rule to calculate these factors, except they have to be Gaussian [6]. Here the following formulas were used to calculate the gain term and the radius:

$$Gain(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{t}{q}\right)^2} \qquad Radius(t) = \frac{m}{2} e^{-\frac{1}{2}\left(\frac{t}{s}\right)^2}$$

where t denotes the number of iterations, q and s are the scales on the time axis in the two functions, while m denotes the number of input points. The scales q and s can be given as arbitrary small numbers, but to improve the efficiency of the procedure they can be calculated in advance. The crucial point of the process is the number of iterations t_0, when the radius diminishes to zero. After this point the fundamental order of points is determined and only the non-approximated input points attract the nearest output.

4.2. Initializing the weights

At the very beginning of the process the weights of the network have to be initialized. The initialization gives a starting spatial location to the points of the grid. Run results show that different types of input points may need different initializations.

The simplest way is to set the weights to small random values. In this case the output points can be far from the input points. To increase the efficiency of the procedure the weights have to initialize around the centroid of the input points.

In case of several input points or a distribution we should initialize the weights close to the point where the density (or the distribution) of the inputs is the highest. This place can be easily determined by clustering the input points to some clusters and the cluster with the maximum number of points is chosen. The weights are initialized around the centroid of that cluster.

4.3. The presentation of input points

The original algorithm presents the input points randomly, hence the grid spreads out in a smooth way. However at the final iterations only few input points are out of the grid, the rest of the input points are presented unnecessarily. Hence to decrease the number of iterations the input points overtook by the grid are not presented any more. This way improves the performance of the network as can be seen in the table. However this type of input presentation cannot be applied in case of large amount of input points.

	Special input	Original input
Mean	3265	6821
Standard deviation	328	1747
Fail	0	13

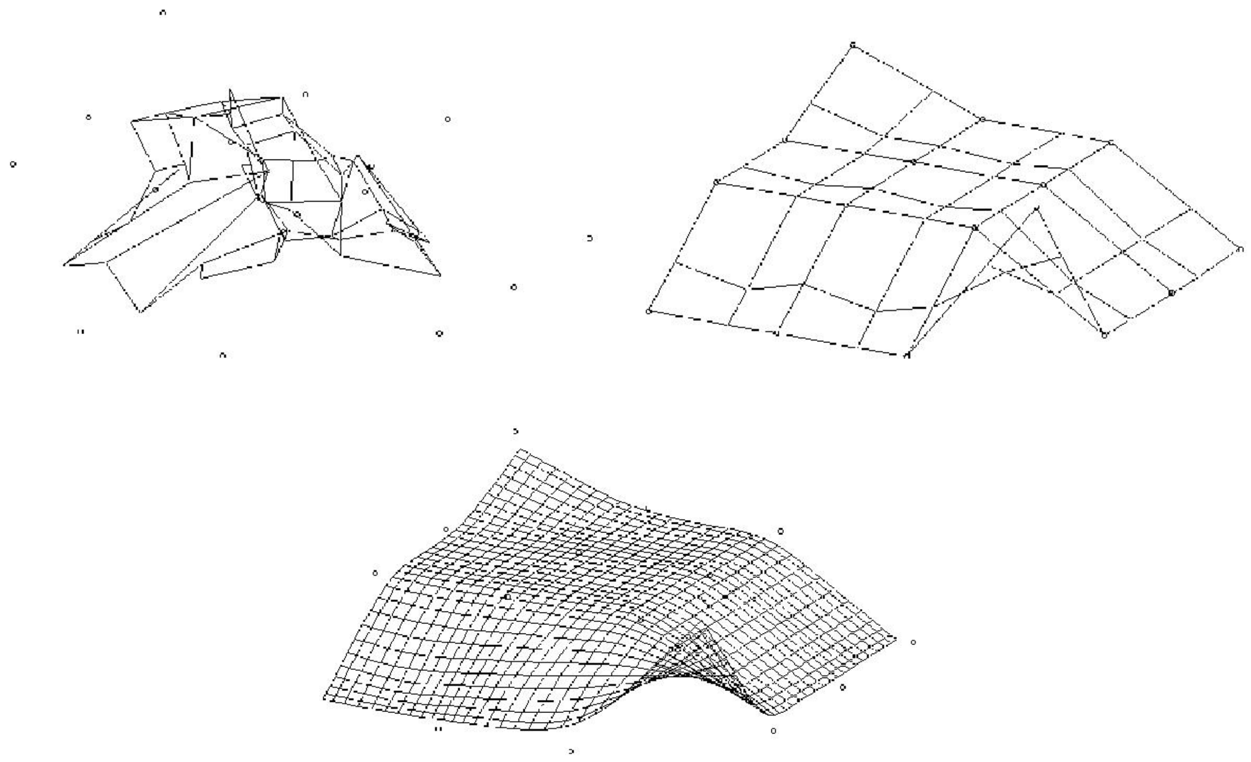

Figure 1: Grid and surface produced by the algorithm

5. The surface

After the training session a grid will be obtained produced by the Kohonen net. This grid can be considered as the control mesh of the future surface. Either interpolation or approximation can be executed by the standard NURBS surface or Bézier-surface method and at this time of the procedure it is irrelevant, that the starting points were scattered so we can consider them as regular, ordered vertices of a control mesh. Figure 1 shows a grid and a surface which has been produced by the algorithm described above.

6. Concluding remarks

Scattered data manipulation is one of the important questions of computer graphics. The existing algorithms are based on the triangularization of the points, however the basic free-form methods use quadrilateral control grids. To solve this problem, and to try the effectiveness of the artificial intelligence method, Kohonen neural network was used to produce the desired grid from the scattered input points. After this step the NURBS method was applied to generate the surface. Comparing with other methods the advantages are twofold: using this algorithm the ordered and scattered points can be interpolated or approximated by the same type of surface, while in term of scattered points our algorithm can be used in a wide range of input conditions.

Acknowledgement

The authors wish to thank the Hungarian National Scientific Fund (OTKA No. F019395) for their financial support of this research project.

References

[1] P. Zsombor-Murray: *General Purpose Contour Plotting Routines*. Proc. 3rd Man-Computer Communications Conf., Nat. Research Council Canada, Ottawa 1973, 35.1–35.21.

[2] W. Boehm, G. Farin, J. Kahmann: *A survey of curve and surface methods in CAGD*. Computer Aided Geometric Design **1**, 1–60 (1984).

[3] T. Foley, H. Hagen: *Advances in scattered data interpolation*. Surv. Math. Ind. **4**, 71–84 (1994).

[4] J. Hoschek, D. Lasser: *Fundamentals of Computer Aided Geometric Design*. AK Peters, Wellesey, Massachusetts 1993, 388–437.

[5] J. Freeman, D. Skapura: *Neural Networks; Algorithms, Applications and Programming Techniques*. Addison-Wesley, 1991.

[6] R. Rojas: *Neural Networks. A Systematic Introduction*. Springer-Verlag, 1996.

[7] T. Kohonen: *Self-organization and associative memory*. 3rd edition, Springer Verlag, 1989.

[8] M. Alder, R. Togneri, E. Lai, Y. Attikiouzel: *Kohonen's algorithm for the numerical parametrisation of manifolds*. Pattern Recognition Letters **11**, 313–319 (1990).

[9] P. Gu, X. Yan: *Neural network approach to the reconstruction of freeform surfaces for reverse engineering*. Computer-Aided Design **27**, no. 1, 59–64 (1995).

[10] M. Hoffmann: *Modified Kohonen Network for Surface Reconstruction*. Publ. Math. Debrecen (to appear).

Received November 26, 1996; final form June 4, 1998