Acta Acad. Paed. Agriensis, Sectio Mathematicae **27** *(2000) 69–74*

CURVE RECONSTRUCTION FROM SCATTERED DATA BY KOHONEN NETWORK

Emőd Kovács (Eger, Hungary)

Abstract. The aim of this paper is to extend the method of modelling scattered data by free-from surfaces presented in [7]. In that paper the Kohonen neural network was used for ordering the data from the scattered points. After the ordering process the B-spline curve or surface approximation and interpolation methods can be applicable also for scattered input points. In this paper we extend this method to solve this problem in case of infinite set of data getting from a cloud of points. A given B–spline curve can also be reconstructed by our method with the help of the Kohonen network.

AMS Classification Number: 68U05

1. Introduction

The manipulation of scattered data is interesting problem, and we can find several application of approximation and interpolation of scattered data in computer graphics and CAD/CAM. In paper [8] mentioned above we developed a new technique for modelling scattered data by free-form surfaces. The main advantage of this method is a preprocessing step which produces a topologically quadrilateral grid from scattered data, hence the basic free-form methods can be applied without any kind of restrictions or modifications. The standard free-form methods like Bézier-surface or NURBS could be applied to approximate or interpolate the data. On the other hand, the new method apply the advantage and effectiveness of the artificial neural networks. The improvement of the network yields a much faster and more reliable algorithm. Obviously the method is also applicable in the case of curves in two or three dimension, when we would like to receive a curve from a cloud of infinitely many points.

For the mentioned preprocessing step an artificial neural network, the Kohonen network, developed by T. Kohonen [2] has been used. The Kohonen network has a self-organizing ability of learning data, which allows a predefined grid to keep its topology during the training procedure, when this grid moves towards the scattered points and follows their spatial structure. The expression "training" will be describe

This research was sponsored by Hungarian National Scientific Research Foundation (OTKA) No. F032679

later, since this is the basic point of the neural network as well as our entire method. A good survey of this method can be found in [8] and [9].

In this paper we try to apply this method in the case of reverse engineering. The final result is a method, which can handle arbitrary set of scattered points. During the reverse engineering process we can capture point data from the curve or surface and applying Kohonen network to produce a CAD model of the object. Hence the user can edit the model to produce an improved or modified object, and perform other computation, leading to whole range of possibilities. We extend the method in training phase, which means that the Kohonen network is trained by point data captured from the object. But these data are not the same in the training cycles, the only common property is that all data are captured from the same object. We can call these type of data stream as infinite data. The number of various data only depends on the measure method. We will prove that the obtained polygon is also applicable for example the B-spline interpolation. Moreover we will examine the reliability and the efficiency of the Kohonen network adapted to the problem mentionde above.

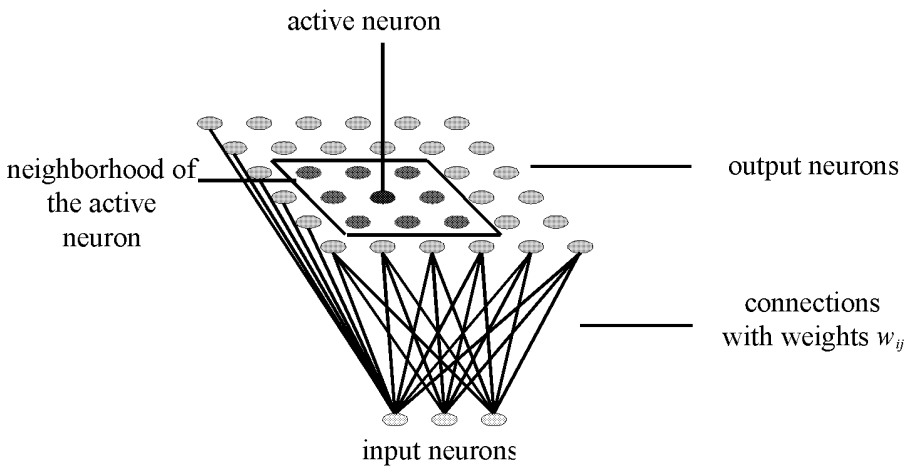
2. Interpolation and Kohonen network

First of all we describe the theory of the Kohonen neural networks and our extension. After this step we will discuss the application of this method for our special problem. The final result is demonstrated by a B-spline interpolation of scattered points. Since the theory of the artificial neural networks is well-known, for details and survey of them we refer e.g. Freeman [4] and Rojas [5]

In this paper we only give a short description of the algorithm of the applied Kohonen network, which produces a rectangular grid onto the 3D scattered points. The exact algorithm can be found in full details in Hoffmann, Várady [8]. 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 a topology preserving grid following the structure of the input points. 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. In the case of 2D input points the output neurons form a line grid. All the output nodes are connected to each input node and a weight is associated to every connection and adjusted during the training. Hence a three or two 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.

The original situation is the following. Let a set of points $p_i(x_{1i}, x_{2i}, x_{3i}) \quad (i = 1, ..., n)$ be given. In our case the number of the points are also given, but every

training period is not required the same input points. During the algorithm we choose n points randomly from the clouds of points. The coordinates of these points will form the input vectors of the net, but every training period this input vector is different from the previous one. 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. We have to remark, that the large number of input points yield difficult problem, when we want to use e.g. B-spline interpolation. 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. The following figure shows the topology of the Kohonen map.



3. Effiency of the algorithm

Since the original method has been described in [7], now we care only for the application of the network and our modification of the algorithm and the results.

If the number of input points in one training cycle 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.

At the 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 type 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. The weights are initialized around the centroid of that cluster.

In two cases the method can give fail result. One of them can occur when the network does not converge, the other one can occur during the Gaussian elimination (see later), because the large number of input points may produce a singular matrix and it cannot be solved. The table below shows the changes of the number of sufficient iterations and interpolations (average after several runs). In our case the weights are initialized by the clustering method.

input points	iterations	neurons	fail
10	3250	40	3%
100	6120	400	2%

4. B-spline interpolation of the output points

When we get the output points, we want to determine a polygon that generates a B-spline curve for a set of known output points we considered. If data points $Q_k, k = 0, ..., n$ lies on the pth-degree nonrational B-spline curve, then it must satisfy the $(n+1) \times (n+1)$ system of linear equations

$$Q_k = C(\overline{u}_k) = \sum_{i=0}^{n} B_i N_{i,p}(\overline{u}_k) \tag{1}$$

where we assign a parameter value, $(\overline{u}_k)$ to each Q_k, and select an appropriate knot vector $U = u_0, \ldots, u_m$.

The control points, B_i, are the $n+1$ unknowns. Eq.(1) has one coefficient matrix, with r solution sets for the r coordinates of the B_i, if r is the number of coordinates in the Q_k (typically 2,3, or 4).

The problem of choosing the $\overline{u}_k$ and U remains, and their choice affects the shape parameterization of the curve. We assume that the parameter lies in the

range $u \in [0,1]$. A lot of common methods are known, equally spaced, chord length, centripetal method. (For more details please see e.g. [1]). In this paper we use a centripetal method: Let

$$d = \sum_{k=1}^{n} \sqrt{|Q_k - Q_{k-1}|}$$

Then $\overline{u}_0 = 0$ and $\overline{u}_n = 1$,

$$\overline{u}_k = \overline{u}_{k-1} + \frac{\sqrt{|Q_k - Q_{k-1}|}}{d} \qquad k = 1, \ldots, n-1.$$

This method gives better result than the chord length method when the data takes very sharp turns. Combining the centripetal method with the averaging technique is recommended. *Averaging,*

$$u_0 = \cdots = u_p = 0 \qquad u_{m-p} = \cdots = u_m = 1$$

$$u_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} \overline{u}_i \qquad j = 1, \ldots, n-p$$

This combination leads to a system of equations Eq.(1) and provides the next advantage, $N_{i,p}(\overline{u}_k) = 0$ if $|i-k| \geq p$. Hence, it can be solved by Gaussian elimination. We used a LU Decomposition to decompose the coefficient matrix into *Lower* and *Upper* triangular components. A good survey of this method can be found in [10].

Although the continuity of the resulting curve is everywhere C^{p-2}, it may not be smooth, or sweet, or fair. The fitted curve may develop unwanted wiggles or undulations. A fairer or smoother curve is obtained by specifying defining polygon points than data points, i.e., $2 \leq p \leq n+1 < k$. The Eq.(1) is more compactly written in matrix form as

$$[Q] = [N]\,[B]$$

$$[N]^T\,[Q] = [N]^T\,[N]\,[B]$$

$$[B] = \left[[N]^T\,[N]\right]^{-1}\,[N]^T\,[Q]$$

Here, $[N]$ is no longer square, the problem is overspecified and can only be solved in a mean sense.

5. Concluding Remarks

Scattered data manipulation is one of the important questions of computer graphics. Comparing with other methods the advantages are twofold: using this

algorithm the ordered and scattered points can be interpolated by the same type of curve, while in term of scattered points our algorithm can be used in a wide range of input conditions. We plan to extend our method to the interpolation of the surface which surface is given by scattered points.

References

[1] Piegl, L., Tiller, W., The NURBS Book, Springer Verlag, 1997.

[2] Kohonen, T., Self-organization and associative memory, Springer Verlag, 1984.

[3] Farin, G., Curves and Surfaces for Computer Aided Geometric Design A Practical Guide, Academic Press, 1996.

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

[5] Rojas, R., Neural Networks. A Systematic Introduction, Springer-Verlag, 1996.

[6] Várady, L., Analysis of the Dynamic Kohonen Network Used for Approximating Scattered Data, Proceedings of the 7th ICECGDG, Cracow (1996), 433–436.

[7] Hoffmann, M., Várady, L., Free-form curve design by neural networks, *Acta. Acad. Paed. Agriensis,* , **Tom. XXIV.,** (1997), 99–104.

[8] Hoffmann, M., Várady, L., Free-form Surfaces for Scattered Data by Neural Networks, *Journal for Geometry and Graphics*, **Vol. 2, No. 1** (1998), 1–6.

[9] Várady, L., Hoffmann, M., Kovács, E., Improved Free-form Modelling of Scattered Data by Dynamic Neural Networks, *Journal for Geometry and Graphics*, **Vol. 3, No. 2,** (1999), 177–181.

[10] Press, W. H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T., Numerical Recipes in Pascal, The Art of Scientific Computing, Cambridge University Press, 1989

Emőd Kovács
Institute of Mathematics and Informatics
Eszterházy Károly College
Leányka str. 4–6.
H-3300 Eger, Hungary
e-mail: emod@ektf.hu