Acta Academiae Paedagogicae Agriensis, Sectio Mathematicae, 24. (1997) pp. 99–104

Free-form curve design by neural networks

MIKLÓS HOFFMANN and LAJOS VÁRADY

Abstract. This paper gives a new approach of the two dimensional scattered data manipulation. The standard approximation and interpolation methods which can only be used for non-scattered data will also be applicable for scattered input with the help of the neural network. The Kohonen network produces an ordering of the scattered input points and here the B-spline curve is used for the approximation and interpolation.

Introduction

The interpolation and the approximation of two dimensional scattered data are interesting problems of computer graphics. By scattered data we mean a set of points without any predefined order. Unfortunately all the standard interpolation and approximation methods—like Hermite interpolation, Bezier curves or B-spline curves—need a sequence of points, hence if we want to apply these methods we have to order the data. A good survey of the scattered data interpolation can be found in [1]. In this paper a completely new approach is given where the self-organizing ability of the neural networks will be used to order the points. The Kohonen network [2,3] can be trained by scattered data, that is the points will form the input of the network, while the weights of the network and their connections give us a polygon, the vertices of which will be the input points. In this way the polygon we obtained can be used as the control polygon of a B-spline curve, so finally a standard approximation or interpolation method can be applied for the scattered data. We begin our discussion with the short definition of the B-spline curve and Kohonen's neural network.

The B-spline curve

The B-spline curve is the most common and widely used free-form representation method which can be used as an approximating and also as an interpolating curve [4]. If we have a sequence of points P_i $(i = 1, \ldots, n)$, then the curve approximating the plane polygon given by the points is defined as

Research supported by the Hungarian National Research Science Foundation, Operating Grant Number OTKA F 019395.

$$S_i(u) = \sum_{r=-1}^{2} P_{i+r} b_r(u) \quad u \in [0,1] \quad i = 2, \dots, n-2$$

where b_r are the well-known B-spline basis functions.

The Kohonen neural network

Neural networks can be divided into two classes, the supervised and the non-supervised learning or self organizing neural networks. Supervised learning neural nets have to be trained with training or test data sets, where the result of the task to be done has to be provided in advance. After training, the net is adapted to the problem by the test set and is able to generalize its behavior. Self organizing networks, however, organize the data during the learning phase where the result of the task is not required. Following the training rules, the network adapts its internal knowledge to the task. The Kohonen neural network is a two-layered non-supervised learning neural network.

Adaptation of the Kohonen net to the problem

Let a set of points P_i $(i = 1, \dots, n)$ (scattered data) be given on the plane. Our purpose is to fit (by interpolation or approximation) a B-spline curve to them. Thus our first task is to determine the order of the points for the interpolating or approximating methods.

The Kohonen net is used to order the points. The first layer of neurons is called input layer and contains the two input neurons which pick up the data. The input neurons are entirely interconnected to a second, competitive layer, which contains m neurons (where $m \geq n$). The weights associated with the connections are adjusted during training. Only one single neuron can be active at a time and this neuron represents the cluster which the input data set belongs to.

Let a set of two dimensional vectors $P_i(x_1, x_2)$ be given. These vectors are called input vectors. The coordinates of these vectors are submitted to the input layer which contains two neurons. When all the input vectors were presented to the input neurons, we restart at the first vector.

Let the output vectors $o_1, \dots, o_m$ be two dimensional vectors with the coordinates $(w_{1j}, w_{2j}), j = 1, \dots, m$, where w_{ij} denotes the weights between the input neuron i and the output neuron j. We use the terms "output vector" and "weights of the output neuron" interchangeably. Let the output map be one dimensional (see Figure 1).

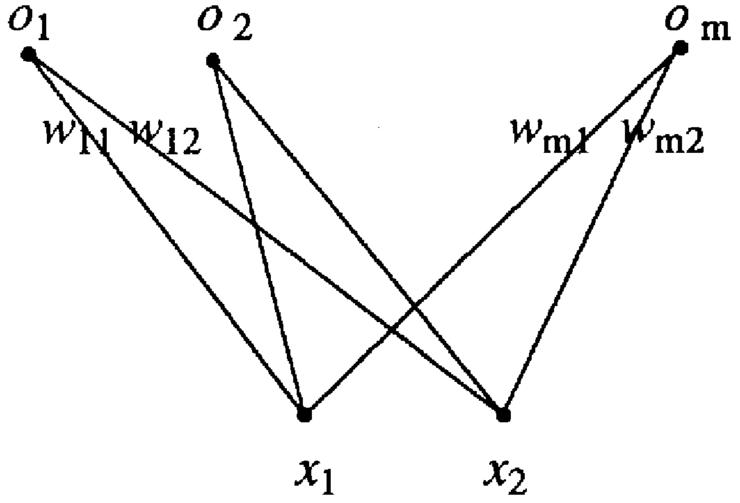

Figure 1.

The training of the network is figured out by presenting data vectors P_i to the input layer of the network whose connection weights w_{ij} are initially chosen as random values. Compute the Euclidean distance between the input point $P_1(x_1, x_2)$ and the output neurons $o_i(w_{i1}, w_{i2})$ with

$$d_i = \sqrt{\sum_{j=1}^{2}(x_j - w_{ij})^2}$$

The neuron c with the minimum distance will be activated, where $d_c = \min\{d_i\}$ $(i = 1, \ldots, m)$. The update of the weights w_{ij} associated to the neurons is only performed within a neighbourhood $N_c(t)$ of c. This neighbourhood is reduced with training time t. The update follows the equation

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}, \quad (i = 1, \ldots, m;\ j = 1, 2)$$

where

$$\Delta w_{ij}^{(t)} = \eta(t)(x_j - w_{ij}^{(t)})$$

and

$$\eta(t) = \eta_0 \left(1 - \frac{t}{T}\right), \quad \text{where} \quad t \in [0, T]$$

Here $\eta(t)$ represents a time-dependent learning rate which is decreasing in time. The term can be chosen as a Gaussian function.

After updating the weights w_{ij} a new input is presented and the next iteration starts. The algorithm determines (using the Euclidean distance) the closest output vector to the presented input vector. The coordinates i.e. the weights of this output vector and those of the vectors that are in a certain neighbourhood of the nearest output vector are updated so that these output vectors get closer to the presented input vector. The degree of the update depends on the gain term and the distance of the output vector and the presented input vector. When the radius (which specifies the neighborhood around an output vector) is large, many output vectors tend towards the presented input. For this reason, initially the output vectors move to places where the density of the input vectors is large, since more input vectors are presented from this areas. The radius (i.e. the size of the neighborhood) and the gain term is decreasing in time. The latter results in that after enough iterations the locations of the output vectors does not change significantly (if the gain term is almost zero then the change in the weights is negligible). The gain term should diminish only when the weights are already close to the input vectors.

A net is said to be convergent if for all the input vectors P_i $(i = 1, \ldots, n)$ there is an output vector o_j such that after a certain time t_0 the Euclidean distance of o_j and P_i is smaller than a predefined limit. A stronger convergence can be obtained if we require that the output vectors which do not converge to an input vector are on the line determined by its two neighbouring output vectors.

In the general case the convergence of the Kohonen net has not been proved yet. Kohonen proved the convergence only in a very simple case when the output is one dimensional and the inputs are the elements of an interval (see[2]).

The radius, the gain term and the number of the outputs can be adjusted so that the output vectors satisfy the stronger convergence mentioned above. This stronger convergence is important especially in term of the smoothness of the future curve.For the detailed description and evaluation of this problem see [5,6]. Let two converging outputs be o_i and o_{i+k} while the outputs which are between the converging outputs be $o_{i+1}, \ldots, o_{i+k-1}$. These outputs are in the neighborhoods of the outputs o_i and o_{i+k} (depending on the radius and k). Since these converged output vectors are close to some input vectors, the outputs $o_{i+1}, \ldots, o_{i+k-1}$ will move towards these outputs (and the input vectors). Since they will move to the common line of the ceonverged output vectors.

The Kohonen net retains the topological ordering of its output vectors. The weights of two output vectors will be close to each other if the vectors are close on the map. The same is true for the approximated input vectors.

Results and further possibilities

The following figures show the ordering of the input vectors and the approximating B-spline curve. There are 20 input vectors and 80 output nodes.

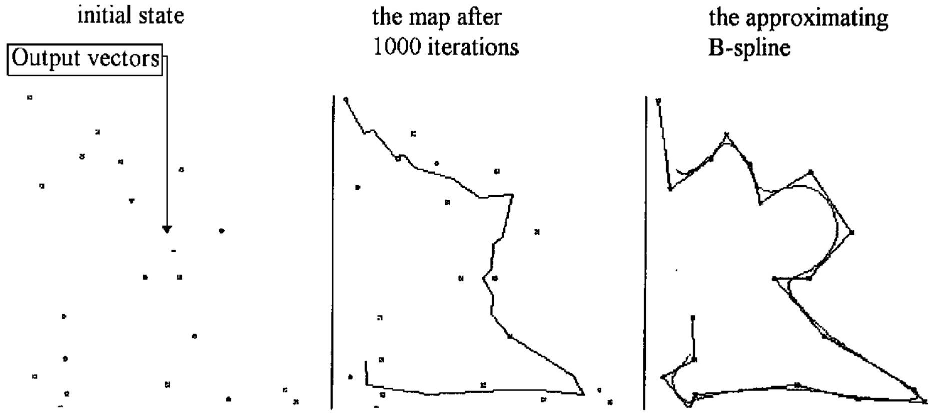

Figure 2.

We plan to generalize the method to three dimensional input points using the Kohonen net. In this case the output map is two dimensional and the input vectors and the weights are three dimensional. When the net converges, the grid approximates the input points and an interpolating or approximating surface can be fitted to the input points.

References

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

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

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

[4] L. D. Faux and M. J. Pratt, Computational Geometry for Design and Manufacture, Wiley & Sons, NY, 1979.

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

[6] M. Hoffmann, Modified Kohonen Neural Network for Surface Reconstruction, *Publ. Math. Debrecen*, 1997 (to appear)

Miklós Hoffmann
Eszterházy Károly Teachers' Training College
Department of Mathematics
Leányka u. 4.
3301 Eger, Pf. 43.
Hungary
E-mail: hofi@gemini.ektf.hu

Lajos Várady
Institute of Mathematics and Informatics
Lajos Kossuth University
H–4010 Debrecen, P.O.Box 12
Hungary
E-mail: lvarady@pc123.math.klte.hu