Bull. Nov. Comp. Center, Comp. Science, 27 (2008), 93–103

How to make self-organizing maps produce smooth adaptive meshes

Olga Nechaeva

Abstract. The problem of smoothness of adaptive meshes produced by the Self-Organizing Maps is considered. It is shown that to improve the mesh smoothness, it is necessary to increase the learning radius. This leads, in turn, to the border effect. The main goal of this paper is to develop a technique allowing us to use a large learning radius for obtaining the sufficiently smooth adaptive meshes without border effect. The technique does not require changing the structure of the SOM neuron layer, and affects only the way of the SOM learning. In addition, an inherent parallelism of the SOM neural network is preserved in the proposed learning algorithm and the algorithm is simple to implement.

1. Introduction

The Self-Organizing Maps (SOM) are neural networks used in a wide variety of areas. In some applications of the SOM, such as image processing, data compression, surface reconstruction [1], the smoothness of maps produced by the SOM is important for obtaining qualitative results. Special attention to map smoothness has to be paid when using the SOM for constructing adaptive meshes intended for numerical simulation problems [2], because the accuracy of a solution algorithm and the quality of adaptive meshes essentially depend on the mesh smoothness. The focus of this paper is on how to obtain sufficiently smooth adaptive meshes by the SOM-based method. We believe that the proposed smoothing technique can be applied not only in the field of mesh construction but in other areas as well.

Within the scope of all kinds of numerical adaptive meshes, there is a class of meshes in which a mesh is an image under an appropriate mapping of a fixed mesh. For the first time, some ideas of applying the SOM to the construction of adaptive meshes from this class are discussed in [3]. But the most reliable SOM-based method of the adaptive mesh construction is proposed in [4]. This method is performed by the composite algorithm, which is based on the composition of the SOM algorithms of different dimensionalities interacting in a special way during self-organization and typically being responsible for the border or the interior of a physical domain.

Our experiments have shown that this method allows us to obtain a good approximation of the border of a physical domain and a mesh density function even if this domain is non-convex. However, these good results can be obtained if a learning radius is sufficiently small at the end of the learning process. It has been noted that a small learning radius results in unsmooth adaptive meshes. But when increasing the learning radius in a composite algorithm, a notorious border effect of the SOM neural network appears, thus making resulting adaptive meshes inappropriate for numerical simulations.

The main objective of this paper is to develop a technique that would allow us to use a large learning radius for obtaining sufficiently smooth adaptive meshes without the border effect. Counteracting the border effect, which enormously rises with increasing the learning radius, was the most challenging task of this investigation.

Some approaches have been proposed to overcome the border effect such as the heuristic weighting rule method by Kohonen [1] and implementation of the SOM on a spherical lattice by Ritter [5]. However, the first approach does not take into account the size of a learning radius, which directly influences the border effect and is a critical parameter for mesh smoothness; and the second one is not applicable, because in the field of adaptive mesh construction it is impossible to change the structure of a given fixed mesh. The technique proposed in this paper does not require changing the structure of a fixed mesh, and influences only the way of the SOM learning. In addition, an inherent parallelism of the SOM neural network is preserved in the proposed learning algorithm and the algorithm is simple to implement.

The paper is organized as follows. Section 2 describes how to apply the SOM for the adaptive mesh construction and contains the learning parameters definition. In Section 3, a measure of mesh smoothness is proposed, and the general scheme of the adaptive mesh construction with a smoothing technique is presented. Section 4 contains the border effect estimation and the detailed description of its elimination. The algorithm of the smoothing technique and examples of smoothed adaptive meshes are proposed in Section 5. Section 6 concludes the paper.

2. Adaptive mesh construction based on SOM

Let G be a physical domain in the Euclidean space with the physical coordinates $x = (x^1, x^2)$, on which an adaptive mesh $G_N = \{x_1, \dots, x_N\}$ is to be constructed, where $x_i = (x_i^1, x_i^2) \in G$, $i = 1, \dots, N$ are the mesh nodes. Let Q be a computational domain in a 2D Euclidean space with coordinates $q = (q^1, q^2)$ with a fixed mesh $Q_N = \{q_1, \dots, q_N\}$, where $q_i = (q_i^1, q_i^2) \in Q$, $i = 1, \dots, N$. Let us consider the case when Q_N is a rectangular uniform mesh. This means that each node of Q_N has four neighbors, and the distances between the neighboring nodes are equal to d_Q. Also, the mesh density function $w : G \to R^+$ is given. The density of a desired adaptive mesh is to be proportional to the values of w.

To construct an adaptive mesh, it is necessary to find a mapping of Q onto G, which transforms the mesh Q_N into the adaptive one G_N with a given mesh density. The method of the mapping determination is required to assure that the boundary nodes of Q_N are automatically transformed into the nodes distributed along the border of G. Let N_{b} be the number of boundary nodes, and N_{int} be the number of the interior ones.

The SOM algorithm can be applied for mesh construction in such a way that neurons are put into correspondence with mesh nodes [6]. This means that there are N neurons in a SOM neural network. In the SOM neuron layer, all the neurons are geometrically located in the same way as in the computational domain. For each pair of neurons q_i and q_j, $i \neq j$, there is a lateral connection between them with strength being a decreasing function of a distance between q_i and q_j.

The weights of the ith neuron are the coordinates $x_i = (x_i^1, x_i^2)$ of the ith mesh node in G. Random points from G serve as input data for the SOM. The density distribution of the resulting mesh is controlled by the probability distribution used for random point generation. The probability distribution can be given by the normalized density function $w(x)$:

$$p(x) = \frac{w(x)}{\int_G w(z)\,dz}. \tag{1}$$

At each iteration t of the SOM learning algorithm, a random point y is generated from G; the winning neuron is selected, which has the weight vector $x_m(t)$ being closest to y; and all the neurons adjust their weights according to the following rule:

$$x_i(t+1) = x_i(t) + \delta(t)\,\eta_{q_m}(t, q_i) \cdot (y - x_i(t)), \tag{2}$$

where $\delta(t) \in [0,1]$ is responsible for a learning step and $\eta_{q_m}(t, q_i) \in [0,1]$ is a function that defines the strength of a lateral connection between the neurons q_m and q_i. These two functions control the magnitude of nodes displacements in G, while the nodes move towards the point y, and essentially influence the quality of resulting meshes and the speed of construction process.

According to [4], a learning step is selected as $\delta(t) = t^{-0.2}\chi(t)$, where $\chi(t) = 1 - \exp[5(t-T)/T]$ and T is a maximum number of iterations that are fixed beforehand depending on N. Let us denote by $B_\gamma(q)$ the neighborhood of a point q in the computational domain, where γ is a radius of the neighborhood, i.e., $B_\gamma(q) = \{p \in Q \mid d(p,q) < \gamma\}$, where $d(\cdot,\cdot)$ is the Euclidean distance. The function for lateral connections is proposed here in the following form: $\eta_{q_m}(t, q_i) = s^{(d(q_m,q_i)/r(t))^2}$, where $s \in (0,1)$ is fixed to be close to zero, e.g., $s = 10^{-5}$, and $r(t)$ is a learning radius, which indicates that all the neurons $q_i \in B_{r(t)}(q_m)$ are connected with q_m by a lateral connection of the strength being greater than s, i.e. $\forall q_i \in B_{r(t)}(q_m) \Rightarrow \eta_{q_m}(t, q_i) > s$. The function $\eta_{q_m}(t, q_i)$ satisfies the following conditions:

- if $d(q_m, q_i) = r(t)$, then $\eta_{q_m}(t, q_i) = s$;
- $\eta_{q_m}(t, q_i) = \eta_{q_i}(t, q_m)$, i.e., a lateral connection between q_m and q_i is symmetric; and
- $\eta_{q_m}(t, q_m) = 1$.

Therefore, at each iteration, the winner receives a maximum displacement, while for other nodes the greater the distance between them, the less their weights change. The learning radius is changing during the iteration process as $r(t) = r(T) + \chi(t)\big(r(1)0.05^{t/T} - r(T)\big)t^{-0.25}$. Here $r(1)$ and $r(T)$ are the initial and the final radii, $r(1) > r(T)$.

Since adaptive meshes are intended for numerical simulations, it is very important to ensure that the boundary nodes properly approximate the border of the physical domain. When using the SOM in a pure form, it is impossible to gain a good border approximation because of the border effect and the topology preservation failures. Much better results can be obtained by using the composite algorithm proposed in [4].

The composite algorithm is based on a special composition of the SOM algorithms of different dimensionalities interacting in a special way during self-organization, typically applied to the border and the interior of a physical domain. As a result, boundary nodes are distributed along the border of the physical domain. At the same time, the composite algorithm essentially improves the topology preservation of mapping and the quality of meshes constructed over non-convex domains and, also, overcomes the border effect for small values of a learning radius. At the end of the iteration process of mesh construction by the composite algorithm, it is necessary to use a small learning radius, e.g., $r(T)$ can be such as only the nearest neighbors, in Q, are located in the neighborhood, because it makes the mesh be sensitive to fine details of the border of G and the mesh density function.

On the other hand, a small radius leads to unsmooth adaptive meshes, and this usually causes a decrease in accuracy of numerical simulations on these meshes. The aim of this paper is to develop the technique allowing us to use a large learning radius for obtaining sufficiently smooth adaptive meshes without the border effect.

3. Measure of mesh smoothness

To measure the smoothness of a quadrilateral adaptive mesh, let us consider the polygonal lines, which are images under the SOM mapping of vertical and horizontal lines of the fixed mesh Q_N. Segments of these polygonal lines connect the corresponding nodes of G_N. The smoothness of a polygonal line

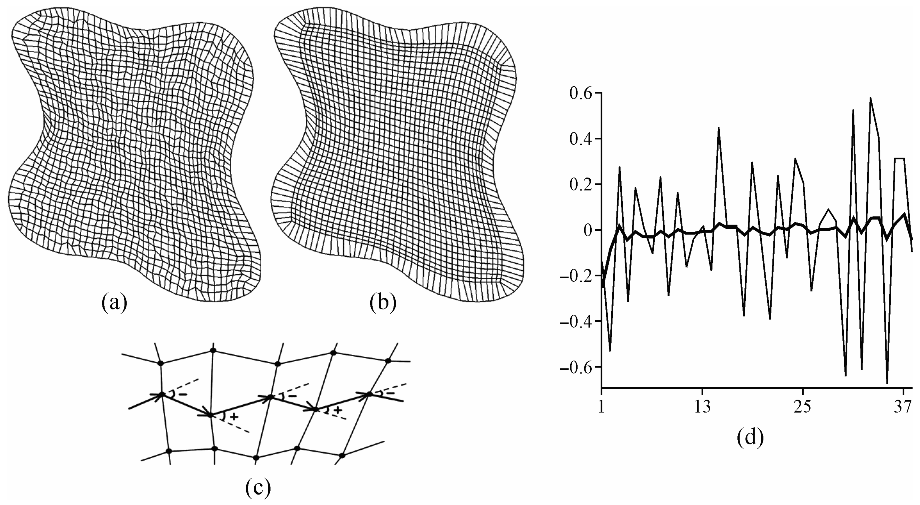

Figure 1. The measure of mesh smoothness: (a) the mesh obtained by the composite algorithm with artificially small final radius $r(T) = 1$; (b) the result of application of the SOM procedure to the mesh (a) with constant learning parameters $r = 15$ and $\delta = 0.005$; (c) illustration of how to measure the mesh smoothness; (d) diagram of sine values for the meshes (a) (thin line) and (b) (thick line)

can be measured by sine values of the angles between segments in a sense that the less the quantity of sign inversions and the amplitude of these values, the smoother the line. Figure 1(c) illustrates this smoothness criterion.

Our experiments have shown that the mesh smoothness depends on the relation between a learning step and a radius. Given a fixed learning step, the larger the radius, the smoother the mesh. This can be clearly seen from the example below. In Figure 1(a), the mesh constructed by the composite algorithm with an artificially small final radius $r(T)$ is shown. This mesh is unsmooth even visually, and a grey line in the diagram of Figure 1(d) indicates large values with many sign inversions for a middle polygonal line of the mesh.

For comparison, the SOM procedure with a large learning radius was performed during certain amount of iterations starting with the mesh in Figure 1(b). The learning radius and the learning step were unchanged in the course of the iteration process. The radius was 15 times greater than $r(T)$ and the step was close to zero and equal to $\delta = 0.005$. The boundary nodes did not move in this experiment, but were allowed to become a winner. As is shown in Figure 1(d), by the black line in the diagram, the smoothness of the last mesh is much better because the sine values are comparatively small and have less sign inversions. But the resulting mesh is inappropriate for the use in numerical simulations because of a poor approximation of the physical domain border.

This experiment is a bright demonstration of the border effect in the SOM, which appears when a learning radius is large. Since the necessary condition for obtaining sufficiently smooth adaptive mesh is a large learning radius, the main problem while smoothing is to handle the border effect.

A general scheme of the adaptive mesh construction with employment of the smoothing technique proposed below is as follows. An adaptive mesh is constructed by the composite algorithm with a learning radius suitable for a proper mesh nodes distribution. The boundary nodes of this mesh are distributed along the border of G. Starting from this mesh, a SOM-like procedure is applied during a fixed number of iterations with a constant learning rate, i.e., $r(t) = r$, $\delta(t) = \delta$, $\eta_{q_m}(t, q_i) = \eta_{q_m}(q_i)$, where the learning radius r is comparatively large and the learning step δ is small. This procedure adjusts locations only of the interior mesh nodes and can be regarded as the last stage of the composite algorithm.

4. Border effect elimination

After termination of the composite algorithm, all mesh nodes are distributed over the physical domain according to a given mesh density function. Consequently, we here assume that for this mesh the following condition of equal winning percentage (EWP) is satisfied (at least approximately) [7]: each neuron has the same chance to be a winner for a randomly generated input point. The probability to be a winner is equal to $1/N$. In the theory of self-organizing neural networks, this condition serves as a quality criterion for the mapping produced by the SOM.

Let us consider an interior neuron q_i, for which a distance to the border of the computational domain is greater than the learning radius r. Since the mesh Q_N is a rectangular uniform, all the neurons q_j from $B_r(q_i)$ as well as all strengths of the lateral connections $\eta_{q_j}(t, q_i)$ are symmetrically located around q_i. Therefore, as follows from the EWP condition, the neuron q_i has the same probability to be influenced by any other neuron q_j. In the physical domain, this means that the node x_i has the same probability to move symmetrically in all directions being guided by neurons from $B_r(q_i)$. Since s is close to zero, then the mutual influence between neurons q_i and $q_j \notin B_r(q_i)$ is assumed to be negligibly small.

If a distance from q_i to the border of the computational domain is less than r, then there is not enough neurons in $B_r(q_i)$ for symmetry. In this case, most of the neurons in $B_r(q_i)$ make the neuron q_i move mainly to the center of the physical domain. To balance the asymmetry, the neuron q_i needs to move aside the border of G.

To evaluate the asymmetry, let us consider the following characteristic of the neuron q_i:

$$\alpha_i = \sum_{j=1}^{N} \eta_{q_j}(q_i). \tag{3}$$

For each node, this characteristic is a sum of lateral connection strengths with all other nodes. If q_i is near the border of Q, then there is not enough terms in sum (3). Therefore, α_i is decreasing near the border of Q. It can be clearly seen from the diagram in Figure 2(c) (a dashed line). All the nodes located at a distance greater than r from the border have the same value of this characteristic.

To eliminate the border effect, it is necessary to balance this asymmetry and to achieve the same value of α_i for all the neurons. We propose the technique that allows us to use the boundary nodes as representatives of missing neurons near the border of Q.

Let us imagine that for each boundary neuron, there are K virtual neurons located outside the computational domain. These virtual neurons do not exist in the algorithm but will help to understand the underlying idea of the proposed technique. The exact locations of virtual neurons are unknown. The only available information is that a distance between the k-th virtual neuron and the corresponding boundary neuron q_m is equal to kd_Q, $k = 1, \ldots, K$, where $K = \lceil r/d_Q \rceil$ is the smallest integer not less than r/d_Q.

To involve the virtual neurons into the learning process, the following questions are to be resolved:

1. In what conditions a virtual neuron becomes a winner?
2. What are the strengths of lateral connections between the neurons q_i, $i = 1, \ldots, N$, and the virtual ones?
3. What are directions and magnitudes of the mesh nodes displacements in the physical domain when the winner is a virtual neuron?

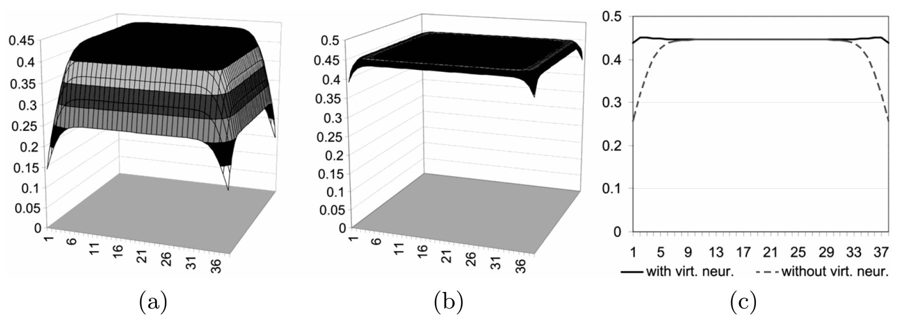

Figure 2. Characteristic of lateral connections symmetry; (a) values of α_i without virtual neurons; (b) values of $\bar{\alpha}_i$ with virtual neurons; (c) the cut of diagrams of (a) (dashed grey line) and (b) (solid black line)

Answer to the question 1. In the case of the presence of virtual neurons, the winner selection cannot be based only on closeness to a random point, because there are no points outside the physical domain. Therefore, at each iteration, it is first of all necessary to decide from what kind of neurons the winner is to be selected. Since the EWP condition is assumed to be satisfied, virtual neurons have the same probability to become a winner as all other neurons. The probability of virtual neurons to become a winner is equal to $N_{\mathrm{b}}K/(N_{\mathrm{b}}K + N_{\mathrm{int}})$, and hence, an interior neuron can be a winner with probability $1 - N_{\mathrm{b}}K/(N_{\mathrm{b}}K + N_{\mathrm{int}})$. To select a winner among virtual neurons, a random point y is generated on the border of G, a boundary node which is the closest one to y is determined, and then the kth virtual neuron randomly selected from a set of virtual neurons, which correspond to the determined boundary node, is assigned to be the winner.

Answer to the question 2. To define the lateral connections strengths between virtual and ordinary neurons, it is necessary to know distances between them in the computational domain. A distance between the k-th virtual neuron and the neuron q_i is assumed to be equal to $d(q_m, q_i) + kd_Q$, where q_m is the boundary neuron corresponding to the virtual neuron. This distance is approximate, because the exact location of this virtual neuron in the computational domain is unknown. The lateral connection between the kth virtual neuron of the boundary neuron q_m and the neuron q_i is taken to be equal to: $\eta_{q_m,k}(q_i) = s^{((d(q_m,q_i)+kd_Q)/r(t))^2}$.

Answer to the question 3. To specify the directions and magnitudes of the mesh nodes displacements in the physical domain, when the winner is a virtual neuron, it is proposed to use a random point y on the border of G, which has been generated for the winner selection among virtual neurons. Let us remind that only the interior mesh nodes can move at the smoothing stage. For each interior node x_i, the direction of its displacement is given by the vector $y - x_i(t)$, i.e., the node x_i moves toward the point y located on the border of G. The magnitude of the displacement is equal to $\delta\eta_{q_m,k}(q_i) \cdot v_i(t) \cdot d(y, x_i(t))$, where $v_i(t) = 1 + kd_Q/d(q_m, q_i)$ and q_m is the boundary neuron which corresponds to the virtual winner. This value was found on the ground of the assumption that the ratio between $d(q_m, q_i)$ and $d(y, x_i(t))$ is equal to the ratio between the $d(q_m, q_i) + kd_Q$ and $v_i(t) \cdot d(y, x_i(t))$. Since the rule is applied only to the interior nodes, then $d(q_m, q_i) \neq 0$.

Taking into account virtual neurons, characteristic (3) changes and is equal to:

$$\bar{\alpha}_i = \sum_{j=1}^{N} \eta_{q_j}(q_i) + \sum_{k=1}^{K} \sum_{m=1}^{N_{\mathrm{b}}} \eta_{q_m,k}(q_i), \tag{4}$$

where $m = 1, \ldots, N_{\mathrm{b}}$ is an index of a boundary node.

In Figure 2(c), the diagrams of α_i and $\bar{\alpha}_i$ for a middle mesh line are shown. It can be seen that $\bar{\alpha}_i$ is almost constant for all neurons. Therefore, the proposed technique balances the asymmetry of a lateral connection near the border.

5. Smoothing stage algorithm

In this section, the algorithm of the smoothing stage is proposed. This algorithm is a SOM-like procedure with constant learning parameters. The learning radius r is chosen to be comparatively large, but it is bounded by the curvature of the border of G. The learning step is to be small, because a fine tuning is needed for smoothing and it does not essentially impair the mesh density approximation.

In addition, when a virtual neuron is a winner, an imaginary random point is outside the physical domain, which can lead to the mesh nodes crossing the border of G. To exclude this effect, a learning step should satisfy the following condition: $\delta(1 + kd_Q/d(q_m, q_i)) < 1$ for any boundary neuron q_m and interior neuron q_i. This means that

$$\delta < \min_{m,i,k} \frac{d(q_m, q_i)}{d(q_m, q_i) + kd_Q} = \frac{d_Q}{d_Q + Kd_Q} = \frac{1}{1+K}.$$

Algorithm. Repeat the following operations during a fixed number of iterations:

1. Generate a random number α from [0,1] with uniform probability distribution.
2. If $\alpha \in [0, N_\mathrm{b}K/(N_\mathrm{b}K + N_\mathrm{int})]$, then perform a SOM-like procedure which consists in the following:
   a) Generate a random point y from G using the probability distribution $p(x)$.
   b) Select a winner node $x_m(t)$ among all the neurons. If $x_m(t)$ is a boundary neuron, then replace a random point with the weights of the winning neuron: $y := x_m(t)$.
   c) Adjust the weights only of the interior neurons according to the rule
   $$x_i(t+1) = x_i(t) + \delta\eta_{q_m}(q_i)(y - x_i(t)).$$
3. If $\alpha \in [N_\mathrm{b}K/(N_\mathrm{b}K + N_\mathrm{int}), 1]$, then perform the following operations:
   a) Generate a random point y from the border of G with the probability distribution $p(x)|_{\partial G}$.
   b) Select the boundary node $x_m(t)$ which is closest to the point y.

c) Choose randomly the number k from $\{1, \ldots, K\}$.

d) Adjust the weights only of the interior neurons according to the rule

$$x_i(t+1) = x_i(t) + \delta\eta_{q_m,k}(q_i)(1 + kd_Q/d(q_m, q_i))(x_m(t) - x_i(t)).$$

Figure 3 shows examples of adaptive meshes constructed by the composite algorithm and then smoothed by the proposed algorithm at the smoothing stage.

It has again to be pointed out that virtual neurons do not exist, and thus, there is no need to change the structure of a fixed mesh when counteracting the border effect. Additionally, our efforts have been directed towards making the learning rule as simple as possible because of the following reasons:

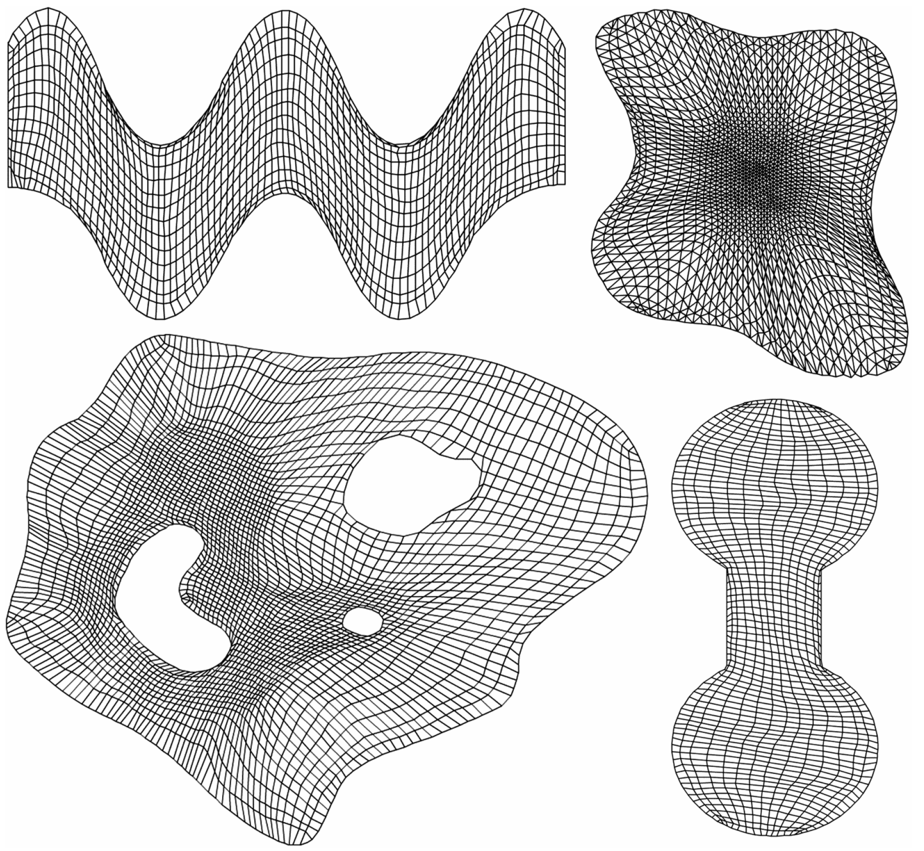

Figure 3. Examples of adaptive meshes constructed by the composite algorithm and then smoothed by the proposed technique

- to save an inherent parallelism of the SOM algorithm which consists in that all neurons are processed according to the same rule independent of each other;
- to avoid problems when constructing a mesh on a complex multiply-connected domain, i.e., the ones with a single or multiple holes, since the border effect is to be controlled at each of the borders.

6. Conclusion

In this paper, smoothness of the adaptive meshes produced by the SOM-based composite algorithm has been studied. The technique which allows us to use a large learning radius without border effect and, thus, to obtain sufficiently smooth adaptive meshes is proposed. This technique counteracts the border effect without changing the structure of a neuron layer and with preservation of the inherent parallelism of a SOM neural network.

In the future, the proposed smoothing stage is to be integrated in the composite algorithm and studied thoroughly in 3D case, where the problem of smoothness and appearance of the border effect is even more important. We believe that the proposed technique is applicable in other areas of SOM applications where the smoothness of maps and the border effect are critical for obtaining the qualitative results.

References

[1] Kohonen T. Self-Organizing Maps. — Berlin; Heidelberg; New York: Springer, 2001. — (Springer Series in Information Sciences; 30).

[2] Lebedev A.S., Liseikin V.D., Khakimzyanov G.S. Development of methods for generating adaptive grids // Vychislitelnye tehnologii. — 2002. — Vol. 7, No. 3. — P. 29.

[3] Manevitz L., Yousef M., Givoli D. Finite element mesh generation using self-organizing neural networks // Special Issue on Machine Learning of MicroComputers in Civil Engineering. — 1997. — Vol. 12, No. 4. — P. 233–250.

[4] Nechaeva O. Composite algorithm for adaptive mesh construction based on self-organizing maps. Artificial neural networks — ICANN 2006 // Lect. Notes Comp. Sci. — Berlin; Heidelberg: Springer, 2006. — Vol. 4131. — P. 445–454.

[5] Ritter H. Self-Organizing Maps on Non-Euclidean Spaces // Kohonen Maps / E. Oja, S. Kaski, eds. — Amsterdam: Elsevier BV, 1999. — P. 97–109.

[6] Nechaeva O.I. Neural network approach for adaptive mesh construction // Proc. of VIII National scientific conference "NeuroInformatics–2006". Part 2. MEPhI, Moscow. — 2006. — P. 172–179.

[7] Fritzke B. Some competitive learning methods. — April, 1997. — (Technical Report. Systems Biophysics / Inst. for Neural Comp., Ruhr-Universitat Bochum).