Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 35 (2014) 861 – 870

Procedia Computer Science

18th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems - KES2014

Image inpainting based on self-organizing maps by using multi-agent implementation

Margarita Favorskaya[a,]*, Lakhmi C. Jain[b], Andrey Bolgov[a]

[a]*Siberian State Aerospace University, 31 Krasnoyarsky Rabochy av., Krasnoyarsk, 660014 Russian Federation*
[b] *University of Canberra and University of South Australian, South Australia, SA 5095, Australia*

Abstract

The image inpainting is a well-known task of visual editing. However, the efficiency strongly depends on sizes and textural neighborhood of “missing” area. Various methods of image inpainting exist, among which the Kohonen Self-Organizing Map (SOM) network as a mean of unsupervised learning is widely used. The weaknesses of the Kohonen SOM network such as the necessity for tuning of algorithm parameters and the low computational speed caused the application of multi-agent system with a multi-mapping possibility and a parallel processing by the identical agents. During experiments, it was shown that the preliminary image segmentation and the creation of the SOMs for each type of homogeneous textures provide better results in comparison with the classical SOM application. Also the optimal number of inpainting agents was determined. The quality of inpainting was estimated by several metrics, and good results were obtained in complex images.

Keyword: Kohonen SOM networks; clustering; image inpainting, texture, multi-agent system

1. Introduction

Video editing systems are the popular software tools, functions of which are extended and complicated in manual and automatic modes. Such systems are based on the achievements of image and video processing, clustering, neural networks, among others. The subject of the current research is a digital image inpainting under small regions with “missing” textures. The missing textures may be pointed by the end-user as a removal of non-desirable object with the area of not more than 5–10 % of frame area or a reconstruction of spots and cracks in the separate frames of old movies. The automatic execution of routine editing operations with high quality is the main goal of editing software tools.

* Corresponding author. Tel.: +7-391-291-9240; fax: +7-391-91-9147.
E-mail address: favorskaya@sibsau.ru.

doi:10.1016/j.procs.2014.08.253

The inpainting algorithms can be divided globally into a geometric inpainting, a texture-based inpainting, and some others [1]. In the pioneer work, Bertalmio et al. [2] proposed a geometric inpainting based on the partial differential equations. The missing area was filled by the linear structure along the isophotes. This method is appropriate for spatial inpainting into an image. The patch-based [3] and the exemplar-based [4] methods are successfully applied as a spatio-temporal inpainting in videos. The expensive texture-based inpainting applies the textural synthesis by using a non-parametric sampling [5], a global optimization [6], among others. The inpainting using discrete regularization on graphs proposed by Ghoniem et al. [7] and the image restoration by solving the Poisson equation [8] are the hybrid approaches.

In our case, a single image with the marked objects for removal exists. Therefore, only spatial methods for texture analysis with the following image inpainting are suitable. Data irredundancy is evident, and it is required to use the supervised or unsupervised learning for high quality texture inpainting within the reasonable time. Due to statistical nature of a texture, the assumption of background inpainting is performed by using the statistical and cluster analysis of a whole image or in a neighborhood area. The Kohonen SOMs were originally designed for data visualization. However, the Kohonen SOM network or its multiple modifications are used in many tasks including the image inpainting. One can read the review in details represented by Kohonen in [9] and the implementations in [10].

The remainder of this paper is organized as follows: Section 2 describes the brief review about the SOM networks applications for visual processing. The SOM adaptation for image inpainting task is discussed in Section 3. The proposed architecture of multi-agent system accelerating the computations is included in Section 4. Section 5 reports the experimental results and evaluation. Finally, conclusions are mentioned in Section 6.

2. Related work

The Kohonen SOM networks have been successfully applied for clustering and visualization analysis, for unsupervised visual object categorization, image data compression, color image segmentation, and textural analysis. The color image segmentation is organized as a clustering through the self-organizing feature maps with fuzzy measures [11] or by using R-tree self-organizing maps [12]. An image as a set of pixels or other structural elements usually cannot be represented as an input vector in the SOM. The geometrical variations such as transitions, rotation, scaling under the different lighting conditions are so wide that only invariant features can be applied. Kohonen mentioned two main types of algorithms [9]:

- The models are computed by a recursive, stepwise approximation process. The input data are loaded into the algorithm one a time as random or periodic sequence. This is the original SOM approach.
- The models using the batches are updated several times, after which the process is stabilized. The input data are interpreted as one batch. The duration of this approach is shorter than the stepwise computation. The batch-learning version of the SOM is reliable for practical applications.

Our contribution is the intermediate decision, when the several inpainting agents with the neighborhood batches relatively the missing area are concurrently initiated in a recursive, stepwise mode.

One of interesting applications is discussed in the research [13], where the self-organizing adaptive map adapts a growing, self-organizing network with a limited predefined knowledge. This algorithm works under the specific conditions, when a structure of object is homeomorphic. Such approach is useful in the task of texture reconstruction. Liou et al. introduced the Conformal Self-organizing Map (CSM), which is based on the SOM and has the potential to learn higher dimensional data [14]. The major difference between the CMS and the SOM is in the mapping of winner neurons: the SOM points each point x to a particular winning neuron in the feature space unlike the CSM chooses a corresponding point from a conformal mapping, which is a collection of flat triangles approximated a 3D object. Also the self-organizing mixture models based on an Expectation-Maximization (EM) algorithm for the topology maps building was proposed in [15]. Verbeek et al.

introduced the self-organizing quality of SOMs in the mixture modeling framework. These authors modified the EM algorithm for the SOM and represented the general algorithm for the mixture of Gaussians.

The Dynamic SOM (DSOM) was proposed by Rougier and Boniface, where the original time-dependent (learning rate and neighborhood) learning function is replaced by a time-invariant function [16]. The DSOM is a neural map with a hypercube or hexagonal lattice structure, the learning process of which reflects two main ideas:

- If a neuron is close to the data, then the learning process is not needed and the winner represents these data.
- If a neuron is not close to the data, then any neuron learns according to its own distance to the data.

The authors introduced the elasticity parameter that modulates the strength of the coupling between neurons. The low elasticity value leads to loose a coupling between neurons while the higher elasticity value results in a tight coupling.

The Parallel Bi-Directional Self-Organizing Neural Network (PBDSONN) architecture [17] can be considered as an extension of the SOM widely applied to image segmentation. The PBDSONN architecture is characterized by the bi-level sigmoidal activations and provides the extraction of pure color images from a noisy background. Several commercial packages are available such as SOM_PAK[1] and the SOM Toolbox[2]. These both packages contain the auxiliary analytical procedures and provide the versatile graphics means.

3. Image inpainting by using of SOM network

The statistical nature of texture is well correlated with the paradigms of the SOM network. The basic concepts the SOM network is located in Section 3.1. The SOM adaptation including the learning stage and the inpainting stage is represented in Section 3.2.

3.1. Basic concepts

The SOM network developed by Kohonen in 1979–1982 [18] has two layers of nodes. The input layer is fully connected to the Kohonen layer. The Kohonen layer as a core of the SOM network is usually represented by one or two-dimensional map. The different sub-areas of the Kohonen layer are associated with different clusters. The SOM is the unsupervised network, for which a self-organization process is started with randomly chosen weights of the nodes in the Kohonen layer. For each input vector during each training cycle, the mean-squared-distance between the input vector and the weight vectors of the nodes is calculated, and the winner node is determined in the neighborhood with a radius R. Kohonen suggested to use all area of network as the initial area to minimize the effect of initial random weights strongly determined by R value. Sometimes, a number of clusters is differed from a number of nodes in the SOM output map. Some extensions were proposed to overcome this limitation such as growing hierarchical SOM [19], the agglomerative contiguity-constrained clustering using k-mean algorithm based on a minimal distance criterion to merge together the neighboring nodes [20], the minimal variance criterion [21], or the unified distance matrix (U-matrix), which is constructed on the trained two-dimensional regular hexagonal grid [22]. Through a number of training cycles, the SOM adjusts the nodes' weights based on the lateral feedback connections according to the topological relations in the input data. Usually, a Gaussian type neighborhood adaptation function is used to decrease the learning process by a rare update of remote from a winner node in spatial and temporal domains. In our experiments, several maps are built simultaneously by a set of the agents in order to improve the final results.

[1] SOM_PAK Team (1990). http://www.cis.hut.fi/research/som_lvq_pak.shtml http://www.cis.hut.fi/research/som_pak/som_doc
[2] SOM Toolbox Team (1999). http://www.cis.hut.fi/projects/somtoolbox/documentation/
http://www.cis.hut.fi/projects/somtoolbox/package/papers/techrep.pdf

3.2. SOM adaptation for image inpainting task

The task of image inpainting based on the Kohonen SOM network includes two main stages: the learning of map and the missing data inpainting. Therefore, the strategy is based on two procedures:

- The preliminary building of the SOM, during which the "missing" pixels do not considered.
- The input vector with "missing" data comes to the input layer of the learned SOM network. Then the "missing" data are restored by the help of the surrounding nodes into the Kohonen layer.

The input feature vector enters simultaneously to all neurons of input layer in the unsupervised SOM network. The weights of synapses are interpreted as the coordinates of nodes positions. The output vector is formed under the principle – the winner takes all, i. e. the nearest to the input vector neuron has a non-zero value, and this value is reduced till to zero for neurons, which are located far from a winner node. During a leaning process, the weights of synapses are tuned in such manner that the nodes are situated into the local concentrations of data. This provides a description of cloud data clustering into a feature space. On the other hand, the connections between neurons correspond to the relations of neighborhood between clusters into a feature space.

There are two types of nodes' connection introduced by Kohonen. The rectangle grid means that each node has a connection with four neighbor nodes, and the hexagonal grid is characterized by six nearest nodes. The SOM building is changed by a bypass of the neighbor nodes. After an initialization of nodes location, these nodes begin to shift into a feature space according to the algorithm, mentioned below:

1. The input feature vector $x(t)$ is chosen randomly and enters in the input layer of the SOM network. The feature vector $x(t)$ is a set of brightness values of pixels. The function describing an image is preliminary transformed to the YUV color space, and the Y component (brightness) is analyzed.

2. The input vector is simultaneously transmitted to the nodes of 2D Kohonen layer. A node of map $m_{ij}(t)$ with the coordinates (i, j), nearest to the input vector $x(t)$, is determined by a function $M(t)$ describing the best matching unit from Eq. 1, where t is a training iteration, W and H are the width and the height of map, respectively, $d(x)$ is a measuring distance, for instance the Euclidean distance, between node and input pattern vectors. The node with the minimal distance $d(x)$ is considered as a winner.

$$M(t) = \min_{i=0..W}\left(\min_{j=0..H}\left(d\left(x(t), m_{ij}(t)\right)\right)\right) \tag{1}$$

3. A winner is shifted by the predetermined steps towards the input vector $x(t)$. During such shifting, it entrains the neighbor nodes of the map. If a radius distance equals 1, then four neighbor nodes for a rectangle grid and six neighbor nodes for a hexagonal grid will move together with the winner. The weighs of synapses $w_i(t)$ are updated by Eq. 2 recurrently, where $h_i(t)$ is a value of neighborhood function for iteration t.

$$\begin{array}{l} w_i(t) = w_i(t-1) + h_i(t-1)\cdot\left(x(t-1) - m_i(t-1)\right) \\ w_i(t) = w_i(t-1) \text{ otherwise} \end{array} \tag{2}$$

4. The algorithm is repeated the predetermined number of iterations.

During the learning stage, a step of an ordering tuning and a step of a fine-tuning are executed. First, a large number of neighborhood nodes is chosen, the nodes move as a collective motion, and a number of iterations is achieved thousand. As a result, the Kohonen layer is roughly reflected a structure of data. Second, a value 1–2 of radius distance is applied, and the local position of nodes is tuned. A number of iterations at this step are several thousand.

A collective motion is turned in such manner that a winner will shift to the input vector more then its neighbor nodes according to a decaying neighborhood function. If all neighbor nodes shift on the equaled values, then such neighborhood function is called a bubble neighborhood function with more number of displacements in the learning stage and less smooth grid. Also a shift value is unique decayed to the end of learning. As a result, the learned Kohonen map with the calculated weights of neurons will be created. Several maps can be learned simultaneously, and the best map may be chosen by use of one of estimation criteria.

For effective map learning, the several neighbor functions are applied such as the Gauss function (Eq. 3), the FrenchHat function (Eq. 4), and the Discrete function (Eq. 5), where *p* is a distance between nodes of map, *t* is a number of iteration, *a* is a predetermined constant value, usually *a* = 2.

$$h(p,t) = e^{\frac{p^2}{2\sigma^2(t)}} \tag{3}$$

$$h(p) = \begin{cases} 1 & \text{if } |p| \le a \\ -\frac{1}{3} & \text{if } a < |p| \le 3a \\ 0 & \text{if } |p| > 3a \end{cases} \tag{4}$$

$$h(p) = \begin{cases} 1 & \text{if } |p| \le 1 \\ \frac{1}{2} & \text{if } 1 < |p| \le 2 \\ \frac{1}{4} & \text{if } 2 < |p| \le 3 \\ \frac{1}{8} & \text{if } 3 < |p| \le 4 \\ 0 & \text{if } |p| > 4 \end{cases} \tag{5}$$

The minimal number of iterations enough for map learning and the minimal accuracy as a sum of weighting mean differences during the iterations are the stopping criteria for the map learning.

An initial image, a list with coordinates of reconstructed points, the learned Kononen map, and the minimal inpainting accuracy are the specified parameters for image inpainting stage. During inpainting, an input vector only with a single "missing" pixel is built. Then the minimal inpainting accuracy is calculated. The empirical coefficient for reliability of weighs vectors *k* is defined by Eq. 6, where *S* is a dimension of a weigh vector.

$$k = S/(S-1) \tag{6}$$

The reliable distance d_r between the reconstructed vector and the winner vector is computed by Eq. 7, where *d* is a measuring distance between the reconstructed vector and the winner vector.

$$d_r = k \cdot d \tag{7}$$

The parameter d_r determines the minimal accuracy for inpainting. The reconstruction procedure is repeated in cycle until all "missing" pixels will not be processed.

4. Architecture of multi-agent system

During experiments, two essential weaknesses of the Kohonen maps using were detected: the accuracy of results strongly depends on the tuning parameters of a map, and the algorithm has a low computational speed. The multi-agent approach solves well these problems. The Multi-Agent System (MAS) for image inpainting includes two main modules, the Learning Module and the Inpainting Module. The proposed MAS is based on the multi-mapping approach. As a result, a set of maps with different roughness of parameters tuning will be obtained that decreases an inpainting time due to more fast processing of "rough" maps.

The architecture of MAS with four types of agents is represented in Fig. 1. The brief description of agents' functionality is situated in Table 1. The Agent-Operator is a single user in this system. The Interface Agent is a program agent for user–software tool interactive communication. The Learning Agent is a program agent for learning a Kohonen map. The Inpainting Agent is a target program agent for image inpainting by the learned map using. Several Learning and Inpainting Agents are required for efficient MAS operation.

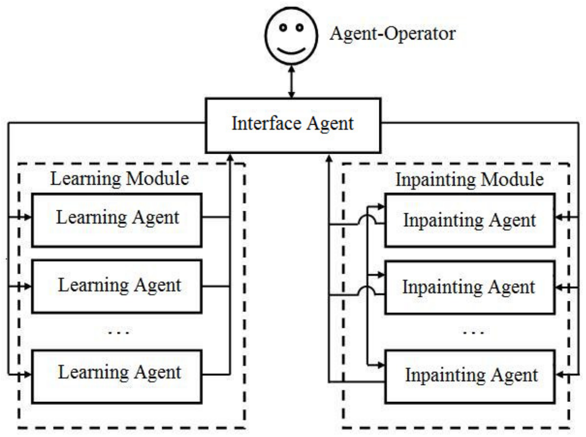

Fig. 1. architecture of MAS for image inpainting

Table 1. Description of agent functionality in MAS for image inpainting

Type of agent	Sensing	Actions	Tasks
Agent-Operator	The receiving of visual, text, and numeric data from Interface Agent	The clicking on buttons and switches	The choice of task
			The results estimation
Interface Agent	The interrupt of clicks from buttons and switches	The representation of visual, text, and numeric data in a screen	The coordination and control of tasks from Agent-Operator
	The obtaining of confirmation/ results from the target agents	The transition of sub-tasks to the target agents	The input/output of visual, text, and numeric data
Learning Agent	The receiving of current task for map learning	The transition of learned Kohonen map with task validation	The learning of Kohonen map
Inpainting Agent	The receiving of current task	The transition of inpainting image	The image inpainting
	The receiving of intermediate results from other agents	The task validation	The transition of intermediate results
		The transition of intermediate data between agents	The analysis of intermediate results
			The final data merging

5. Experimental results

The software tool "MAS-Image Inpainting" was designed into environment "Microsoft Visual Studio 2012" by using C# language with the possibilities of "Net Framework 3.5" platform. The software tool is executed in a multi-stream mode and includes three executable files: InterfaceAgent.exe, LearningAgent.exe, and InpaintingAgent.exe. The communication between the applications is provides by "RabbitMQ" platform [23]. The library "KohonenMapLibrary" involves all low-level instructions for the Kohonen maps learning and the image inpainting. The library "AgentLibrary" includes the classes for a working with the agents' states and the communication instruments. All scenarios are visualized in a graphic interface of InterfaceAgent.exe. The Kohonen maps produced by the "Matlab" realization and the software tool "MAS-Image Inpainting" were compared for verification the own designed software tool. The example is situated in Fig. 2.

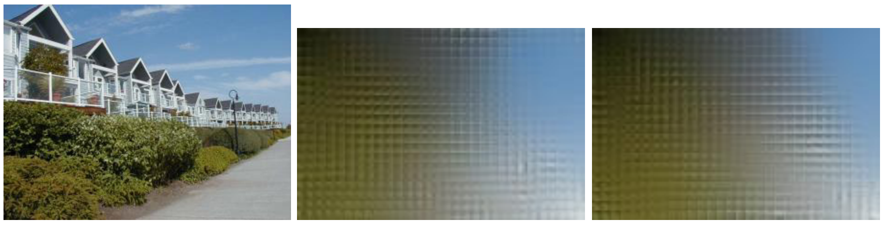
a b c

Fig. 2. (a) a test image with 640×480 pixels; (b) Kohonen map by using "Matlab"; (c) Kohonen map by using "MAS-Image Inpainting"

As one can see from Fig. 2, the received maps are identical and cover a palette of test image wholly. The execution time achieved 4,183 ms and 9,024 ms for "MAS-Image Inpainting" and "Matlab", respectively.

For experimental researches, 100 test images with different sizes and complexity were chosen from "CBIR database"[1]. All images were divided into ten sets with ten images into each set. The brief description of test sets is presented in Table 2.

Table 2. Description of test sets from "CBIR database"

Test set	Image sizes, pixel	Background complexity	Sizes of objects
1	480×320	Simple background	Objects with normal and large sizes
2	480×320	Cluttered background	Objects with small sizes
3	640×480	Simple background	Objects with normal and large sizes
4	640×480	Cluttered background	Objects with small sizes
5	768×512	Simple background	Objects with normal and large sizes
6	768×512	Cluttered background	Objects with small sizes
7	880×590	Simple background	Objects with normal and large sizes
8	880×590	Cluttered background	Objects with small sizes
9	1024×768	Simple background	Objects with normal and large sizes
10	1024×768	Cluttered background	Objects with small sizes

[1] http://sourceforge.net/projects/cbir-ngvmv/

Some results of inpainting are shown in Fig. 3 with the classical Kohonen SOM application and the preliminary segmentation of images. The last case provides better visual results.

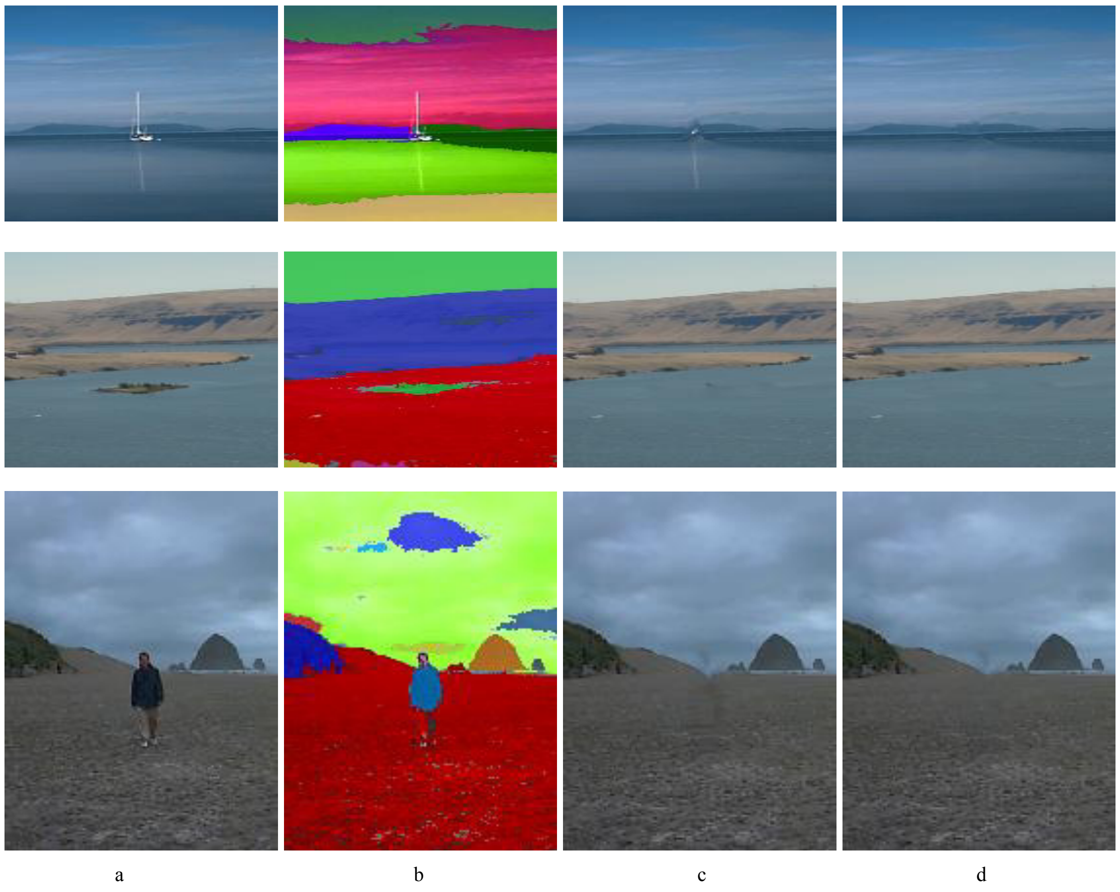

Fig. 3. (a) the initial images; (b) the segmented images; (c) the inpainting by the Kohonen SOM; (d) the inpainting by the Kohonen SOM with a preliminary segmentation

The methodology of experimental research included a removal of 5–10 % area into each of test images, then this "missing" area was inpainting by the software tool "MAS-Image Inpainting" and inpainting results were estimated relative to the initial test images by Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM) metrics. The computer Core i5 660 with 2 cores, frequency 3.33 Hz, RAM 8 Gb, core cash 256 Kb was used. During experiments, the optimal number of covering maps was estimated (Table 3).

The palette covering by one learned map does not achieve 50 % of initial image palette. Therefore, the multi-mapping is a reasonable technique for image inpainting. The four learned maps is the optimal number of maps, the accuracy of palette covering is more than 92 %. The covering results of eight maps are close. However, the computational cost is higher for the last case.

Table 3. Accuracy of palette covering

Accuracy of palette covering, %	Test sets									
	1	2	3	4	5	6	7	8	9	10
1 learned map	51.20	48.28	49.61	50.04	46.06	45.23	43.09	46.04	47.74	45.21
2 learned maps	74.89	67.04	72.27	62.30	66.36	70.41	60.02	68.18	61.53	67.85
4 learned maps	94.94	92.35	97.31	93.43	96.72	95.55	93.40	96.29	97.42	95.01
8 learned maps	98.74	98.93	99.23	96.73	97.41	95.72	97.42	98.86	98.32	98.02

The estimations of accuracy for test images inpainting are represented in Table 4.

Table 4. Accuracy of image inpainting

Accuracy of image inpainting	Number of agents	Test sets									
		1	2	3	4	5	6	7	8	9	10
MSE, %	1 agent	3.04	4.24	6.25	7.35	18.38	20.73	33.06	36.24	60.83	64.13
	4 agents	4.14	5145	2.55	3.26	16.17	17.95	27.78	29.18	45.61	46.82
PSNR, dB	1 agent	41.29	41.95	38.16	37.84	35.48	34.13	33.92	33.72	32.06	31.93
	4 agents	39.29	39.68	40.06	39.93	37.48	36.32	35.69	34.95	33.49	32.94
SSIM, %	1 agent	96.42	95.46	98.09	97.64	74.84	73.42	75.61	74.38	73.44	72.45
	4 agents	95.72	94.79	97.84	97.89	69.93	70.57	80.07	79.54	75.73	74.85

As one can see from Table 4, the multi-agent configuration provides better results especially with the increase of resolution and structural complexity in cluttered images.

The estimations of computational speed during learning and inpainting stages are mentioned in Table 5.

Table 5. Estimations of computational speed

Estimation of computational speed, ms	Number of agents	Test sets									
		1	2	3	4	5	6	7	8	9	10
Learning stage	1 agent	5,244	5,792	6,613	6.024	5,270	5,421	6,022	5,388	6,278	5,995
	2 agents	4,004	4,102	4,931	5,102	4,259	4,195	4,126	4,271	5,180	4,619
	4 agent	2,360	2,178	3,042	2,325	2,678	2,392	2,778	2,347	3,017	2,735
	8 agents	3,401	3,533	4,415	4,589	3,582	3,682	3,349	3,411	4,790	3,925
Inpainting stage	1 agent	3,710	3,792	3,944	4,024	4,683	4,421	4,128	4,388	3,725	3,995
	2 agents	3,004	3,102	2.931	3,102	3,269	3,195	3,126	3,271	2,584	2,615
	4 agent	2,092	2,178	2.163	2,325	2,292	2,392	2,090	2,147	1,687	1,735
	8 agents	2,401	2,533	2,415	2,589	2,582	2,682	2,349	2,411	1,801	1,925

The multi-agent approach shows a high computational speed. The optimal number of agent equals 4. The increase of agents' number does not provide the good results because the communication cost grows in the system.

6. Conclusion

The task of image inpainting is based on spatial information, and the Kohonen SOM network is well suitable as the unsupervised clustering technique. Our contribution is directed at the overcoming some weaknesses of the SOM application. The preliminary image segmentation provides better results with the SOM creation for homogeneous textures. Also the multi-mapping is a reasonable technique for image inpainting, when four learned maps provide the accuracy of palette covering more than 92 % of an image area. The multi-agent configuration demonstrates better results in accuracy estimated by MSE, PSNR, and SSIM metrics and in computational speed. The optimal number of agents equals 4. The large number of agents implies the increase of cost for communication between agents.

References

[1] Venkatesh MV, Cheung S, Zhao J. Efficient object-based video inpainting, *Pattern Recognition Letters* 2009;**30(2)**:168–179.

[2] Bertalmio, M., Bertozzi, A.L., Sapiro, G., 2001 "Navier–Stokes, fluid dynamics, and image and video inpainting," 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Kauai, Hawaii, USA, Vol. 1, 355–362.

[3] Boulanger J, Kervrann C, Bouthemy P. Space–time adaptation for patch-based image sequence restoration. *IEEE Trans PAMI* 2007;**29(6)**:1096–1102.

[4] Criminisi A, Perez P, Toyama K. Region filling and object removal by exemplar-based image inpainting. *IEEE Trans on Image Processing* 2004;**13(9)**:1200–1212.

[5] Efros, A., Leung, T., 1999. "Texture synthesis by non-parametric sampling." IEEE International Conference on Computer Vision, Corfu, Greece, 1033–1038.

[6] Wexler, Y., Shechtman, E., Irani, M., 2004. "Space–time video completion." 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Washington, USA, Vol. 1, 120–127.

[7] Ghoniem M, Chahir Y, Elmoataz A. Nonlocal video denoising, simplification and inpainting using discrete regularization on graphs. *Signal Processing* 2012;**90(8)**:2445–2455.

[8] Shen J, Jin X, Zhou C, Wang CCL. Gradient based image completion by solving the Poisson equation. *Computers & Graphics* 2007;**31**:119–126.

[9] Kohonen T. Essentials of the self-organizing map. *Neural Networks* 2013;**37**:52–65.

[10] Seiffert, U. and Jain L.C. (Editors) Self-Organising Neural Networks: Recent Advances and Applications. *Springer-Verlag, 2002*.

[11] Liu L, Wang B, Zhang L. An approach based on self-organizing map and fuzzy membership for decomposition of mixed pixels in hyperspectral imagery. *Pattern Recognition Letters* 2010;**31(11)**: 1388–1395.

[12] Rallabandi VPS, Sett SK. Image retrieval system using R-tree self-organizing map. *Data & Knowledge Engineering* 2007;**61(3)**:524–539.

[13] Piastra M. Self-organizing adaptive map: Autonomous learning of curves and surfaces from point samples. *Neural Networks* 2013;**41**:96–112.

[14] Liou CY, Kuo YT, Huang JC. Conformal self-organizing map on curved seamless surface. *Neurocomputing* 2008;**71(16–18)**:3140–3149.

[15] Verbeek JJ, Vlassis N, Krose BJA. Self-organizing mixture models. *Neurocomputing* 2005;**63**:99–123.

[16] Rougier N, Boniface Y. Dynamic self-organising map. *Neurocomputing* 2011;**74(11)**:1840–1847.

[17] Bhattacharyya S, Maulik U, Dutta P. A parallel bi-directional self-organizing neural network (PBDSONN) architecture for color image extraction and segmentation. *Neurocomputing* 2012;**86**:1–23.

[18] Kohonen T. *Self-Organizing Maps*. Springer: Berlin, 1995.

[19] [21] Rauber A, Merkl D, Dittenbach M. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks* 2002;**13(6)**:1331–1341.

[20] Murtagh F. Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering. *Pattern Recognition Letters* 1995;**16(4)**:399–408.

[21] Kiang MY. Extending the Kohonen self-organizing map networks for clustering analysis. *Computational Statistics & Data Analysis* 2001;**38(2)**:161–180.

[22] Yang L, Ouyang Z, Shi Y. A Modified Clustering Method Based on Self-Organizing Maps and Its Applications. *Procedia Computer Science* 2012;**9**:1371–1379.

[23] Videla A, Williams JW. *RabbitMQ in action*. New York: Manning; 2012.