Scientific Research Publishing

Journal of Signal and Information Processing, 2020, 11, 75-102
https://www.scirp.org/journal/jsip
ISSN Online: 2159-4481
ISSN Print: 2159-4465

Segmentation of Visual Images by Sequential Extracting Homogeneous Texture Areas

Alexander Goltsev[1], Vladimir Gritsenko[1], Dušan Húsek[2]

[1]International Research and Training Center of Informational Technologies and Systems, National Academy of Sciences of Ukraine, Kiev, Ukraine
[2]Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, Czech Republic
Email: agoltsev@adg.kiev.ua, dusan@cs.cas.cz

How to cite this paper: Goltsev, A., Gritsenko, V. and Húsek, D. (2020) Segmentation of Visual Images by Sequential Extracting Homogeneous Texture Areas. *Journal of Signal and Information Processing*, **11**, 75-102.
https://doi.org/10.4236/jsip.2020.114005

Received: August 12, 2020
Accepted: November 7, 2020
Published: November 10, 2020

Abstract

The purpose of the research is to develop a universal algorithm for partial texture segmentation of any visual images. The main peculiarity of the proposed segmentation procedure is the extraction of only homogeneous fine-grained texture segments present in the images. At first, an initial seed point is found for the largest and most homogeneous segment of the image. This initial seed point of the segment is expanded using a region growing method. Other texture segments of the image are extracted analogously in turn. At the second stage, the procedure of merging the extracted segments belonging to the same texture class is performed. Then, the detected texture segments are input to a neural network with competitive layers which accomplishes more accurate delineation of the shapes of the extracted texture segments. The proposed segmentation procedure is fully unsupervised, *i.e.*, it does not use any a priori knowledge on either the type of textures or the number of texture segments in the image. The research results in development of the segmentation algorithm realized as a computer program tested in a series of experiments that demonstrate its efficiency on grayscale natural scenes.

Keywords

Texture Feature, Texture Window, Homogeneous Fine-Grained Texture Segment, Extraction of Texture Segment, Texture Segmentation

1. Introduction

The principal problem in the field of image processing is that of object-ground separation, *i.e.* the task of detecting the area of an object in the image. Then, this object should be recognized. Any object is recognized, to a large degree, by its

shape. Therefore, in order to recognize an object in a real-world visual scene, it is necessary, at first, to detect its borders in the image. In natural images, borders of texture segments are, often, borders of objects or their parts. Thus, the first step of the object recognition should be division of the image into separate texture segments. When texture segments are delineated, it becomes possible to start solving such tasks as object-ground separation, object recognition and image understanding.

In the majority of works devoted to the texture segmentation problem, statistical analysis of input images is performed for description, recognition and segmentation of textures. In the statistical approach, the image is usually processed by a sliding window within which various statistical characteristics are measured (e.g. [1] [2] [3] [4] [5]).

In the approach which falls into category of unsupervised texture segmentation the segmentation algorithm uses some universal texture features to extract any texture regions (e.g. [6] [7] [8] [9] [10]). For the majority of the unsupervised segmentation algorithms it is necessary to specify the number of the texture segments to be extracted.

The complexity of the texture segmentation problem may be facilitated by providing a segmentation device with a number of distinctive patches of those textures that should be recognized and segmented in the image. Using these patches, it is possible to measure characteristics of the indicated textures and to adjust parameters of the segmentation device according to them. Adjustment of segmentation parameters may be done by means of supervised learning. In this approach, the segmentation device is preliminarily learnt on a training set of the texture class samples (e.g. [11] [12]). Many supervised feed-forward neural network-based models for texture segmentation and recognition are of this approach (e.g. [13] [14] [15] [16] [17]). Some of them use supervised neural network classifiers, such as [15] [16] [18] [19] [20], and associative (assembly) neural networks [13] [14] [21] [22].

The textures may be subdivided into fine-grained and coarse (-grained) ones. Image regions representing objects with discontinuities in depth, in material, or in illumination, etc., correspond to coarse texture segments. And a fine-grained texture is characterized by a smooth variation of image brightness and fine granularity. Texture segmentation of images may be performed as by means of coarse texture region extraction, as by extraction of homogeneous fine-grained texture segments. The presented approach belongs to the latter one. It is worth to note that the discrimination of textures to "coarse" or "fine-grained" is, to a large extent, rather relative. Indeed, by increasing the degree of image resolution, fine-grained segments may turn into coarse ones. In the presented work, the size and image resolution are fixed.

It is well known that the concept of "texture" is intuitive and it does not have any formal or commonly accepted definition (e.g. [17]). Therefore, there is no possibility to evaluate formally texture segmentation results. Various researchers understand somewhat different things by the term "texture". In many publica-

tions, the term "texture" is used in relation to any areas occupied by the entire objects such, for example, as faces, flowers, animals, trees, buildings, bicycles and so on. So, in this works, the term "texture" is equivalent to that of the object. This understanding of the texture notion is dominating in the world. And therefore, all databases for texture and object segmentation and recognition have been constructed for testing different segmentation and recognition algorithms within this understanding of texture.

We have a different opinion about the concept of texture. We define the "homogeneous texture segment" as an image area all small parts of which have similar texture characteristics (texture features). And, in this approach, we use only homogeneous texture areas as texture segments.

Let us notice, that the idea to start the analysis of visual images by extraction of homogeneous texture areas is almost conventional. For example, the following sentence is a quote from [23]: "The task of partitioning a natural image into regions with homogeneous texture, commonly referred to as image segmentation, is widely accepted as a crucial function for high-level image understanding, significantly reducing the complexity of content analysis of images".

It is worth to note that the problem of image segmentation has been tried to solve not only by means of texture extraction. For example, the language model LDA (Latent Dirichlet Allocation), based on the design of spatial documents, have been used to segment images of greenhouse plants [24]. Color analysis methods have been proposed for image segmentation (e.g. [25] [26]). The method for automatic segmentation of color images presented in [25], despite the differences in details, is very similar in its basic ideas to our approach.

The competitive layer models (CLM) have been also applied for image (texture) segmentation (e.g. [27] [28]). In [27] [28], the segmentation procedure is based on features that are local Gabor filter responses at different spatial frequencies and orientations. Generally, a competitive layer model is proposed for grouping and clustering data of different types. The CLM architecture was first introduced as a model for spatial feature linking in [29]. Here we mention CLM not only as effective technique for solving the problem of image segmentation, but, also, because we use the same architecture of competitive layers to improve the results of texture segmentation obtained in our previous works [30] [31] [32]. However, our system of competitive layers is not a full CLM, since it uses somewhat other algorithms and solves somewhat other tasks. The prototype of this neural network is described in [33] [34].

The present paper is a continuation of our previous researches [30] [31] [32]. The aim of all works of this series is to develop a universal algorithm that should segment any input image into a number of homogeneous fine-grained texture segments. In other words, the proposed algorithm should perform partial texture segmentation of the image by delineating the homogeneous texture areas present in the image while its inhomogeneous areas remain unsegmented.

The algorithms of sequential extraction of all homogeneous texture segments of an input image are described in [31]. This set of algorithms belongs to a re-

gion growing approach (e.g. [35]-[41]), more precisely, to the pixel-based region growing method (e.g. [41]) which is rather computationally expensive.

According to [31], initially, a set of texture features characterizing the largest and most homogeneous texture segment of the input image is found. To do that, an exemplary patch of pixels is firstly detected in the image. Somewhat different versions of this procedure are described in [30] [32]. The texture features extracted from the exemplary patch are considered to be those that best characterize the selected texture segment to be extracted. Then, the exemplary patch is used as starting points from which the expansion of this segment starts taking into account its characterizing feature set. After extracting the segment, its region is excluded from further consideration. Then, the next homogeneous texture segment is extracted by the same algorithms. This process of sequential segmentation is completed when no more sizable homogeneous segments remains in the image. The process is fully unsupervised, *i.e.*, it does not use a priori knowledge on either the type of textures or the number of texture segments present in the image.

The segmentation procedure described in [31] does partly solve the task of preliminary segmentation of an input image into a set of homogeneous segments. However, since the segmentation procedure is purely local, it extracts separately all texture segments of the same texture class that are not adjacent. As a result of that, some number of homogeneous texture segments of the same texture class may be marked by different colors in the image. In the present work, we consider the processes described in [31] as the first stage of entire segmentation procedure. At the second stage, the procedure of merging the extracted segments belonging to the same texture class is performed.

Also, the segmentation procedure of [31] segments the homogeneous regions rather roughly. In this paper we include a neural network with competitive layers which accomplishes more accurate delineation of the shapes of the extracted texture segments.

Many algorithms of a post-processing type are proposed, e.g. [10] [25] [36] [40] [42] [43] [44]. For example, the methods described in [42] [43] use the segmentation results obtained by such segmentation algorithms as the mean shift [45] and graph-based image segmentation [46] as the initial segments. There are different types of post-processing algorithms, e.g. the merge importance based method [44], the Markov random field based method [40], the overall coding length minimization method [10].

After accomplishment of the merging operation, the final third stage of our entire segmentation procedure starts; it is processing of the merged segments by means of a neural network with competitive layers. Due to the neural network functioning, the shapes of homogeneous texture regions present in the image become delineated more precisely. In addition, a larger area of the image becomes segmented than that reached at the first stage of the entire texture segmentation procedure.

The paper is organized as follows. Section 2 gives an overview of the entire

segmentation procedure and briefly characterizes algorithms of all its stages. Section 3 contains a short description of the texture features used to evaluate texture characteristics of different image points and describes formation of an etalon feature pattern. In Section 4 we explain algorithms of computation of a degree of integral similarity between exemplary patch of the segment to be extracted and all texture windows of this segment. Section 5 depicts the procedure of merging the segments of the same texture class. Section 6 introduces a neural network with competitive layers and algorithms of its functioning. Section 7 demonstrates experimental results of different stages of the entire texture segmentation procedure. Section 8 is devoted to discussion and conclusions.

2. An Overview of the Texture Segmentation Process

A set of predetermined texture features is used to describe peculiarities of textures. These features allow the segmentation algorithm to distinguish different textures between one another. The same texture features are applied in all stages of the texture segmentation process. The procedure of feature extraction is performed by means of a set of sliding texture windows that cover the whole image (with overlaps between one another). A texture window is a comparatively small square frame which serves to evaluate generalized texture characteristics of the image within the window. Now, the window-based method for evaluation of texture characteristics is the most prevalent technique.

At first, the largest and most homogeneous texture segment is extracted in the image. The first operation of the segmentation process is finding a set of texture features characterizing this segment (see [30] [32]). The essence of this operation is as follows.

In order to evaluate homogeneity of different points of the image, it is covered with a number of test windows. The test window is also a square frame but it is larger than the texture window. Therefore, each test window contains quite a large number of overlapping texture windows within it. In each test window, the degree of texture homogeneity is measured by means of sequential comparisons between all texture windows contained in the test window under consideration. Then, among all test windows of the image, a window with the maximum degree of homogeneity is selected. Inside this test window, an initial seed spot (starting seed) is detected.

The initial seed spot is, actually, a patch of pixels. The detected initial spot is considered as a compact area of the exemplary seed pixels that certainly belongs to the texture segment to be extracted. The procedure of finding the exemplary pixels is described in details in [32]. Earlier (and outdated) version of this procedure is presented in [30].

Sets of texture features are measured in the detected exemplary pixels using the texture windows associated with them. Then, these features are averaged in order to form an exemplary feature set (representing feature set) which best characterizes the homogeneous texture segment to be extracted. Using this exemplary feature set, the segmentation algorithm starts to delineate the segment.

For each segment under processing, the detected initial seed pixels are gradually expanded by means of including the pixels of its neighborhood into the segment. In other words, the label of belonging of the image pixels to the segment gradually spreads from the initial seed pixels to the borders of the segment. In this region growing procedure, the algorithm examines each appending pixel by comparison of its texture features with the exemplary feature set. If these feature sets are similar enough, the neighboring pixel is appended to the growing segment, if not, the pixel is excluded from it. The procedure for growing the seed region continues until the segment reaches its borders. Note, that this procedure is sequential, pixel-based, and, therefore, is rather computationally expensive.

As a result of this expansion procedure, a more or less homogeneous and (usually) simply connected fine-grained texture area is delineated. After delineation of the current segment, its area is excluded from the further consideration. Thereby, each subsequent segment does not intersect with the segments that were extracted earlier. Extracting the next texture segment begins again with a search for a new set of representing seed pixels belonging to the largest and most homogeneous texture segment present in the rest of the image. The segmentation process is completed when the image contains no more sizable homogeneous areas. After that, the coarse texture and non-texture areas remain unclassified in the image.

Almost the same segmentation procedure is used in the present work with small changes indicated in Sections 3 and 4. Since, this procedure is presented in detail in [31], we describe it here briefly. Since the segmentation procedure is designed to extract only fine-grained homogeneous texture segments, it cannot extract a coarse texture segment as an entire area. Moreover, the algorithm should divide every coarse texture area into a number of smaller homogeneous texture patches. This inherent peculiarity distinguishes the proposed segmentation procedure from related techniques that can classify a mixture of areas with a fine-grained texture and coarse ones as belonging to the same texture segment.

As mentioned above, now we subdivide the whole segmentation process into three stages, the first of which is the extraction of all significant texture segments present in the analyzed image. After completion of that, the next task arises. Since some number of the extracted segments may be of the same texture, naturally, they should be merged into the same texture class.

The merging procedure is performed as follows. The averaged texture characteristics of all extracted texture segments are computed and compared in pairs. The comparison between two segments within a pair is accomplished using four texture characteristics. If all four characteristics of two compared segments are similar enough, all pixels of both segments are combined as belonging to the same texture class. In the figures presented in Section 7, this is expressed by marking the segments with the same color. In our approach, the merging procedure is considered as the second stage of the whole segmentation procedure.

After the merging procedure, the obtained texture segments are input to a

neural network with competitive layers. Processing the segments by the neural network with competitive layers is the final third stage of the whole segmentation procedure. The goal is to achieve more accurate shapes of homogeneous texture segments present in the image. Prototypes of the neural network with competitive layers are described in [33] [34].

The layered network contains such number of neural layers that is equal to the number of texture segments obtained after the merging procedure. Each layer of the network is used to represent only one texture segment. As a result of inputting all merged texture segments into the neural network, each segment becomes represented in its own layer by means of activation of the corresponding neurons in the layer. Then, the network neurons begin to interact with each other by excitatory and inhibitory manner. Excitatory connections link the neurons of the same layer. The interaction between neurons of different layers is competitive (inhibitory). The network functioning is performed in a series of iterations; its number is variable and is not limited. As a result of the network functioning, configurations of active neurons change in the layers. When the configurations of active neurons in all layers become stable, the process ends. In addition to more precise delineation of shapes of homogeneous texture segments present in the image, processing leads to such a result that a larger area of the image becomes segmented than after the first stage of the whole segmentation procedure.

3. Texture Features, Their Representation, and Formation of the Etalon Feature Pattern

Now, a brief description of the features is proposed. Let us notice that the algorithms described below can work with any other feature set, provided that this set is represented in the same format of normalized vectors (see below).

Only grayscale images are processed; each of them is of $N = I \times J$ pixels, with the range within 0 - 255 brightness values. In order to evaluate texture characteristics of different image points, the whole image is covered by overlapping square texture windows of the same size. For simplification, we postulate that the number of texture windows is equal to the number of image pixels N and each image pixel corresponds to the center of the associated texture window.

A set of M texture features (feature types) is computed in every texture window. At the beginning of processing a given image, all texture features are computed in all N texture windows covering the image and saved for further usage.

The following simple set of texture features is used. The first part of the feature set is the averaged histogram of the brightness of all pixels of the texture window; it consists of 11 bins. The second part is a histogram of the orientation of all pixels of the texture window. This histogram is computed based on filtering the image by the Scharr filter [47]. Orientation histogram consists of 9 bins. The next feature is the average brightness of the texture window. The last texture feature is the average non-uniformity brightness in the window. This feature is calculated by averaging the outputs of all Scharr filters of the window. The feature is useful to distinguish texture patches of the same average brightness but

with different internal texture structure.

So, a total of $M = 22$ texture features are computed in each texture window. All these M values constitute the feature set representing the texture peculiarities of the corresponding window.

Thus, we use the same set of texture features as in [30] [31] [32]. However, in the present work, unlike [31], the feature description of the whole image (of all N texture windows) is presented in somewhat simpler format. Videlicet, each feature value is represented as an integer value in an integer three-dimensional array $\boldsymbol{G}[i][j][m]$ of $I \times J \times M$ elements (instead of a binary four-dimensional array in [31]). The array $\boldsymbol{G}$ is calculated only once at the beginning of the texture segmentation procedure and saved for further usage.

As mentioned in Section 2, extraction of every homogeneous texture segment starts with finding the exemplary patch (seed pixels) of the sought-for texture segment ([32]). Let us introduce a binary matrix $\boldsymbol{P}^{\text{EXMPL}}(k)[i][j]$ of $N = I \times J$ elements to represent the exemplary seed patch of the k-th segment. In this matrix, its one-valued elements indicate the image pixels belonging to the patch. The matrix is used to create some etalon of texture features (etalon feature pattern) which describes the texture peculiarities of the segment to be extracted. An integer vector $\boldsymbol{W}(k)[m]$ of M elements is introduced to represent the etalon feature pattern of the k-th segment. Actually, the vector contains averaged combination of all feature sets extracted from the exemplary seed patch $\boldsymbol{P}^{\text{EXMPL}}(k)$. Unlike the array $\boldsymbol{G}[i][j][m]$, which is computed only once, the vector $\boldsymbol{W}(k)$ is formed anew for each k-th segment. The procedure of formation of the vector $\boldsymbol{W}(k)$ is described in a (C++ like) pseudo-code in **Algorithm 1**.

Algorithm 1. Combining all feature sets extracted from the exemplary seed patch $\mathrm{P}^{\text{EXMPL}}$.

```
Input:   binary matrix P^EXMPL(k)[I][J];   // Exemplary seed patch for the k-th segment.
         integer array G[I][J][M];         // Feature description of the whole image.
Output:  integer vector W(k)[M];           // Etalon feature pattern: combination of features
                                           // representing the k-th segment.
W(k) = 0;                                  // Zeroing vector W(k).
U = 0;                                     // Zeroing number of pixels U in P^EXMPL(k).
{
    for (i = 0; i < I; i++)                // Cycle through X coordinates of image.
        for (j = 0; j < J; j++)            // Cycle through Y coordinates of image.
            if (P^EXMPL(k)[i][j] = = 1)    // Testing: if pixel belongs to P^EXMPL(k).
            {
                U++;                                  // Incrementing the number of pixels U.
                for (m = 0; m < M; m++)               // Cycle through all features.
                    W(k)[m] += G[i][j][m];            // Summation in the m-th element of
                                                      // the vector W(k).
            }
}
for (m = 0; m < M; m++)                    // Cycle through all features.
    W(k)[m] = W(k)[m] / U;                 // Averaging m-th feature value of vector W(k).
```

4. Computation of the Degree of Similarity between the Representing Patch and Texture Windows

In the first stage of the whole segmentation procedure, the process of extraction of all homogeneous fine-grained texture segments is sequential and iterative, one segment per iteration. The algorithm of extraction of every k-th texture segment is based on comparison between the etalon feature pattern $\boldsymbol{W}(k)$ and feature sets fixed in the integer three-dimensional array $\boldsymbol{G}[i][j][m]$. Let us introduce an integer matrix $\boldsymbol{E}(k)[i][j]$ of $N = I \times J$ elements to represent results of this comparison. In the notations $\boldsymbol{W}(k)$ and $\boldsymbol{E}(k)$, the letter k serves only to indicate that it is the process of extracting the k-th segment in which the matrix $\boldsymbol{E}$ and the vector $\boldsymbol{W}$ are currently used. In fact, each element of the matrix $\boldsymbol{E}(k)$ reflects the degree of similarity between the etalon feature pattern $\boldsymbol{W}(k)$ of the k-th texture segment and the feature set of each (i, j)-th texture window covered the image. Therefore, we name the matrix $\boldsymbol{E}(k)$ a matrix of similarity (similarity matrix). The similarity matrix $\boldsymbol{E}$ takes a great part in all stages of the texture segmentation procedure. In particular, it is used in the process of extraction of every texture segment in the same manner as in [31].

The procedure of the matrix $\boldsymbol{E}(k)$ formation is described in a (C++ like) pseudo-code in **Algorithm 2**.

Algorithm 2. Formation of the matrix $\boldsymbol{E}(k)$.

```
Input:   integer array G[I][J][M],      // Feature description of the whole image.
         integer vector W(k)[M];        // Etalon feature pattern: combination of features
                                        // representing the k-th segment.
Output:  integer matrix E(k)[I][J];     // Degree of similarity between feature patterns
                                        // of texture windows and etalon feature pattern.
E(k) = 0;                               // Zeroing the matrix E(k).
for (i = 0; i < I; i++)                 // Cycle through X coordinates of image.
    for (j = 0; j < J; j++)             // Cycle through Y coordinates of image.
        for (m = 0; m < M; m++)         // Cycle through all M features.
            {                           // Finding max and min between G[i][j][m]
                                        // and W(k)[m].
            if (G[i][j][m] > = W(k)[m])
                {
                max = G[i][j][m];       // Calculation of the maximum value.
                min = W(k)[m];          // Calculation of the minimum value.
                }
            else
                {
                min = G[i][j][m];       // Calculation of the minimum value.
                max = W(k)[m];          // Calculation of the maximum value.
                }
            E(k)[i][j] + = min / max;   // Calculating similarity of m-th feature.
            }
```

As follows from Algorithm 2, the procedure of the matrix $\boldsymbol{E}(k)$ formation includes a series of sequential comparisons in pairs of feature sets. One part of every pair is constant; it is the etalon feature pattern $\boldsymbol{W}(k)$. The other part of the pair changes in each comparison; it is the feature set of the (i, j)-th texture window. The similarity value between these feature sets is estimated by the sum of the ratios between the minimum and maximum values for all M features. So, the similarity value is a single integral characteristic by means of which we evaluate the similarity between both compared feature sets. The similarity value is fixed in the (i, j) element of the matrix $\boldsymbol{E}(k)[i][j]$. Those elements of the matrix $\boldsymbol{E}(k)$, that have much higher values than its other elements, correspond to the k-th texture segment.

Figure 1 illustrates this description by a demonstration experiment with a real image. The figure shows the process of extraction of the first texture segment in the image. The figure consists of two parts that are in mutual coordinate correspondence. The left part is the input image. The first texture segment extracted by the segmentation algorithm (in the first iteration) is the ground area. The location of the exemplary seed patch $\boldsymbol{P}^{\text{EXMPL}}(1)$, found by using the algorithm described in [32], is indicated by a black square in the left part of the figure. The right part of the figure presents the similarity matrix $\boldsymbol{E}(1)$ computed according to Algorithm 2. Thus, in this first iteration, the matrix $\boldsymbol{E}(1)$ is computed specially to extract the ground texture segment in the image. In the right part of the figure, the values of the matrix $\boldsymbol{E}(1)$ elements are expressed by the intensity of white. As seen in Figure 1, the elements of the matrix $\boldsymbol{E}(1)$ corresponding to the ground texture segment have much higher values than all other its elements.

So, we consider the value of each (i, j)-th element of the similarity matrix $\boldsymbol{E}(k)$ as a measure of its belonging to the sought-for k-th texture segment. This value is used in the process of extracting each k-th texture segment in order to make a decision about including (or excluding) the considered image pixel (associated with the (i, j)-th texture window) into the segment. That is, starting from the

(a) (b)

Figure 1. (a) A photograph of a bench and a black square in the figure that indicates location of the exemplary seed patch detected by the algorithm of finding a set of texture features characterizing the most homogeneous texture segment present in the input image. (b) The similarity matrix $\boldsymbol{E}(1)$ computed at the first iteration of the segmentation process.

example patch, the initial seed region of the sought for k-th texture segment is gradually grown by including pixels of its neighborhood into the segment. More precisely, in the seeded region growing process, the (i, j)-th element of the matrix $\boldsymbol{E}(k)$ is compared with some decreasing threshold. If the value of the (i, j)-th element of the matrix $\boldsymbol{E}(k)$ exceeds the threshold, the currently considered neighboring (i, j)-th test pixel is assigned the label of the sought-for k-th segment, if not, the pixel is skipped.

The subsequent procedures of delineation of each homogeneous segment are performed exactly in the same way as in [31].

Let us introduce a binary matrix $\boldsymbol{R}(k)[i][j]$ of $N = I \times J$ size to represent the k-th texture segment that has been extracted during the k-th iteration of the first stage of the whole segmentation procedure: the pixels belonging to the k-th segment are represented by one-valued elements of the matrix $\boldsymbol{R}(k)$, other its pixels are zero-valued. Thus, according to above description, the last operation of extracting the k-th texture segment is a sequential comparison of all elements of the matrix $\boldsymbol{E}(k)[i][j]$ with a threshold D:

$$\boldsymbol{R}(k)[i][j] = \mathbf{1}(\boldsymbol{E}(k)[i][j] - D), \tag{1}$$

where D is the threshold which is gradually decreases in the process of extraction of each texture segment; $\mathbf{1}(x)$ is the unit step function:

$$\mathbf{1}(x) = \begin{cases} x = 1, & \text{for } x \geq 0, \\ x = 0, & \text{for } x < 0. \end{cases}$$

5. Algorithm for Combining the Segments of the Same Texture

As mentioned above, the second stage of the whole segmentation process is merging the segments of the same texture class. The merging procedure is rather simple. First, a list of all extracted segments is formed, which is sorted in descending order of their size. Then, the pair of the largest segments is compared in their texture characteristics, whether they are similar enough to be referred to the same texture class. The comparison procedure is as follows.

To assess the similarity of textures while testing two segments, four values are used. Each of these values is some measure of similarity between the segments which is computed by comparing the texture feature sets of both segments. The first value evaluates the similarity of the segments by brightness histograms; it is computed by comparison between all corresponding bins of the histograms. The second value evaluates the proximity of the prevailing orientations within the segments; it is calculated by comparing all corresponding bins of the orientation histograms. The third value reflects the proximity between the average brightness of the segments. The fourth value describes the closeness between the average non-uniformity brightness of the segments (on the base of Scharr filtration). Each value of these four ones is computed according to the pseudo-code of **Algorithm 2**. Tested segments are recognized as belonging to the same texture on-

ly in the case if each of the values exceeds a predefined threshold. The thresholds are determined experimentally.

If the segments of a pair of the currently tested segments are enough similar, they are merged as follows. Videlicet, if the textures of the s-th and the u-th segments are found to be sufficiently similar, and the s-th segment is larger than the u-th one, all pixels of both segments are combined in the binary matrix $\boldsymbol{R}(s)$ representing the largest s-th segment according to the formula

$$\boldsymbol{R}(s)[i][j] = \boldsymbol{R}(s)[i][j] \vee \boldsymbol{R}(u)[i][j], \tag{2}$$

where $\vee$ is disjunction.

After that, the matrix $\boldsymbol{R}(u)$ is zeroed: $\boldsymbol{R}(u) = 0$; and the u-th segment is excluded from the list of the segments sorted in descending order of their size, and the list is updated. Then, a new tested pair of the largest segments is chosen and the same comparison procedure is performed to find out whether they are of the same texture, or not.

In Section 7, we present experiments with real images that demonstrate results of the merging procedure.

6. A Neural Network with Competitive Layers

In our segmentation procedure, a neural network with competitive layers is used in order to improve the segmentation results, that is, to attain a more precise delineation of the shapes of all homogeneous texture segments extracted in the image. As mentioned above, our system of competitive layers is not a full competitive layer model (CLM) [27] [28] [48]. Although our system and CLM are very similar, they have some differences, at that, our system is rather simpler. The main difference is as follows. In CLM, each neuron in every layer is laterally interacting with all other neurons of the same layer by means of a complex function which is excitatory for short distances and weakly inhibitory for larger distances. Unlike that, in our neural network, the lateral interaction between the neurons of the same layer is only excitatory and local with a simple transfer function. Namely, this interaction between the neurons is transmitted by local excitatory connections with the weights that are equal throughout the network. There is no learning in this neural network.

The neural network with competitive layers consists of several neural layers which number is equal to the number of the extracted segments remaining after the previous merging procedure; let us designate this number by K. Every neural layer is intended to represent all pixels of a certain texture segment and, therefore, contains such number of neurons that is equal to the number of image pixels: $N = I \times J$. So that, each neuron of every neural layer corresponds to one pixel of the image. In other words, all neural layers have one-to-one correspondence with one another and with the input raster. And the configuration of neural activity in the layers represents the shapes of the corresponding texture segments.

Let us introduce such a functional unit of the neural network as a competitive neural column. Each competitive neural column includes all neurons of all net-

work layers that correspond to the same pixel of the input raster.

Figure 2 illustrates this description depicting the competitive neural network of 8 layers and one competitive neural column.

There is competitive (inhibitory) interaction between all neurons of every competitive neural column which crosses the network layers. This competitive interaction is realized by means of a "winner-take-all" (WTA) procedure which is used to select the column's neuron with the maximum excitation. Then, the selected neuron is set active. The WTA procedure is performed in each neural column of the network independently from all others. So, only one neuron within each competitive neural column can be active. If some neuron of the layer is active, this means that the corresponding image pixel belongs to a certain texture segment.

The network uses an iterative procedure which includes interactions of neurons inside each layer through excitatory connections and competitive WTA interrelations between all neurons of every neural column. The network has unchangeable connection structure. All connection weights are fixed and equal between one another.

An output of each neuron of the network is binary. Activity of all neurons is calculated synchronously in a series of time steps. The number of time steps varies for each processed image. Let us designate the time steps by t; $t = 1, 2, 3, \cdots, T$, where T is the final time step for the currently processed image.

Each neuron of the network has the same structure of lateral connections with other neurons of its layer. More exactly, each (i, j)-th neuron the k-the layer has mutual excitatory connections with all neurons of the same k-the layer that lie within the square of Z size with the (i, j)-th center. All these mutual excitatory connections have equal gradual weights throughout the network. Spreading the neural activity within the network layers by means of these lateral excitatory connections exacts a strong influence on the excitation levels of all network neurons.

All K segments represented in the matrix $\boldsymbol{R}$ are used for input excitation of the competitive neural network. More exactly, all these segments are used in the process of formation of a set of K similarity matrices $\boldsymbol{E}(k)(k = 1, 2, 3, \cdots, K)$ according to description of Section 4, where each matrix $\boldsymbol{E}(k)$ is computed by sequential application of Algorithm 1 and Algorithm 2. Now, to form the matrix $\boldsymbol{E}(k)$, the same Algorithm 1 and Algorithm 2 are also applied in sequence, but with the following small modification. As distinct from the case of Section 4, the entire k-th extracted segment represented in the matrix $\boldsymbol{R}(k)$ is used instead of the exemplary seed patch $\boldsymbol{P}^{\text{EXMPL}}$ (see Algorithm 1). Consequently, some new etalon feature pattern $\boldsymbol{W}(k)$ is created on the basis of all texture windows belonging to the k-th segment. As above, the etalon $\boldsymbol{W}(k)$ reflects the peculiarities of the k-th texture segment. Thus, each (i, j)-th element of the matrix $\boldsymbol{E}(k)$ represents the degree of similarity between the k-th texture segment and the (i, j)-th texture window of the image under consideration.

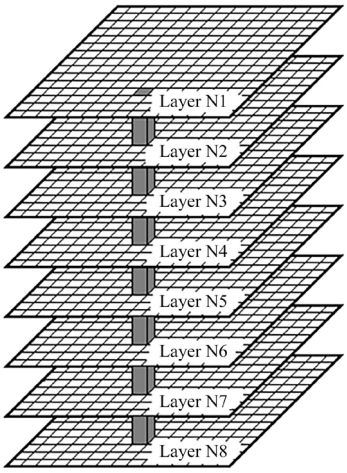

Figure 2. A schematic picture of a neural network with competitive layers. In the figure, the network is depicted as consisted of 8 neural layers. One neural column is highlighted in gray.

So, K matrices $\boldsymbol{E}(k)$ are used for input excitation of the neural network with competitive layers. That is, at every t-th time step, the value of each (i, j)-th element of the similarity matrix $\boldsymbol{E}(k)$ is fed to the input of the (i, j)-th neuron of the k-th neural layer.

As mentioned above, the segmentation procedure is intended to delineate only homogeneous fine-grained texture segments present in the analyzed image. Since, in all real images, there are boundary regions and regions of coarse (-grained) textures, these regions should be excluded from consideration. Let us introduce a binary matrix $\boldsymbol{Q}[i][j]$ of $N = I \times J$ elements to partition the homogeneous fine-grained texture segments and those image areas that certainly are not homogeneous. These areas are partitioned in the matrix $\boldsymbol{Q}$ in such a way that its one-valued elements represent the homogeneous texture segments and its zero-valued elements represent the areas that should not be processed.

To form the matrix $\boldsymbol{Q}[i][j]$, the following operations are performed. For each (i, j)-th competitive neural column, the maximal value $\boldsymbol{E}^{\max}[i][j]$ is calculated (among all K elements of the (i, j)-th column)

$$\boldsymbol{E}^{\max}[i][j] = \underset{k=1}{\overset{K}{\mathrm{MAX}}}\ \boldsymbol{E}(k)[i][j]. \tag{3}$$

And, the matrix $\boldsymbol{Q}$ is formed according to the following formula

$$\boldsymbol{Q}[i][j] = \mathbf{1}\left(\boldsymbol{E}^{\max}[i][j] - L\right), \tag{4}$$

where L is some constant threshold, which is determined experimentally.

So, Equations (3) (4) mean that for each (i, j)-th competitive neural column, all K values of the (i, j)-th elements of the matrices $\boldsymbol{E}(k)$ are compared with the threshold L. If all these values are less than L, the corresponding (i, j)-th pixel of the analyzed image is attributed as belonging to the boundary regions or regions of coarse (-grained) textures and, therefore, are excluded from further processing. This is expressed in zeroing the (i, j)-th element of the matrix $\boldsymbol{Q}$ by

Equation (4).

To describe the image processing by the neural network with competitive layers, we introduce the following matrices. Let us designate by $\boldsymbol{H}^t(k)[i][j]$ the level of input excitation of the (i, j) th neuron of the k-th layer at the t-th time step ($t = 1, 2, 3, \cdots, T$). We also denote by $\boldsymbol{P}^t(k)[i][j]$ the binary matrix for representation of the binary output neural activity of the k-th layer at the t-th time step. Actually, the neural activity configuration of the k-th layer $\boldsymbol{P}^t(k)$ depicts the shape of the k-th texture segment at the t-th time step. This configuration changes during the network recalculation.

The zero time step of the network recalculation is somewhat different from the subsequent steps. That is, at the zero time step there are no active neurons in the network at all. Therefore, there are no excitatory interactions between the neurons of the same layers through lateral connections. The only excitation effects at the input of network neurons are those that are obtained from the corresponding elements of the similarity matrix $\boldsymbol{E}$. So, for the zero time step, $\boldsymbol{P}^0(k) = 0$ for all k; $k = 1, 2, 3, \cdots, K$. Accordingly, the level of input excitation of the (i, j)-th neuron of the k-th layer at the first time step is calculated by the formula

$$\boldsymbol{H}^1(k)[i][j] = \boldsymbol{Q}[i][j]\boldsymbol{E}(k)[i][j]. \tag{5}$$

The matrix-multiplier $\boldsymbol{Q}$ is introduced in Equation (5) to underline the fact that those competitive neural columns which correspond to the zero-valued elements of the matrix $\boldsymbol{Q}$ are not processed.

The "winner-take-all" procedure is carried out in every processed neural column at each t-th time step. That is, the maximum excitation level $\boldsymbol{H}^t(\max)[i][j]$ is calculated among K neurons of every (i, j)-th competitive neural column at each t-th time step, according to the formula

$$\boldsymbol{H}^t(\max)[i][j] = \underset{k=1}{\overset{K}{\mathrm{MAX}}}\ \boldsymbol{H}^t(k)[i][j]. \tag{6}$$

Then, the most excited neuron is chosen among all K neurons of each neural column. The chosen neuron is set active (with one-valued output), while all other neurons of the column are set inactive (with zero-valued output). So, as a result of the WTA procedure, activity of each (i, j)-th neuron of the k-th layer at the t-th time step is computed by the formula

$$\boldsymbol{P}^t(k)[i][j] = \mathbf{1}\left(\boldsymbol{H}^t(k)[i][j] - \boldsymbol{H}^t(\max)[i][j]\right). \tag{7}$$

Note that Equations (6)-(7) describe the WTA procedure at an arbitrary t-th time step, including the first one. Consequently, at the first step of the network recalculation, the configuration of active neurons $\boldsymbol{P}^1(k)$ in all K layers represents the shapes of the segments computed on the basis only the similarity matrices $\boldsymbol{E}(k)$.

In the process of subsequent recalculation of the network, at each time step t, every (i, j)-th neuron of the k-th layer receives an exciting effect from the following two sources. The first source is constant for all time steps, it is the excitation that comes from the (i, j)-th element of the similarity matrix $\boldsymbol{E}(k)$. The

second excitation component for every (i, j)-th neuron comes from all those active neurons of the same k-th layer that surround this neuron and have lateral excitatory connections with it. These active neurons are located inside the square of size Z with center (i, j). They exert their excitatory effect through connections with gradual weights that are equal throughout the network; let us designate these weights as U, $U < 1$. So, the input excitation level of the (i, j)-th neuron of the k-th layer at the $(t + 1)$-th time step is calculated by the formula

$$\boldsymbol{H}^{t+1}(k)[i][j] = \boldsymbol{Q}[i][j]\left(\boldsymbol{E}(k)[i][j] + U\sum_{l=i-Z/2}^{i+Z/2}\sum_{f=j-Z/2}^{j+Z/2}\boldsymbol{P}^{t}(k)[l][f]\boldsymbol{E}(k)[l][f]\right). \tag{8}$$

In Equation (8), the operation, denoted by double-sum symbols, describes the summation of the excitations obtained from all active neurons that are located inside the square of size Z, which surrounds the considered (i, j)-th neuron. As above, the matrix-multiplier $\boldsymbol{Q}$ is used to underline the fact that those competitive neural columns that correspond to the zero-valued elements of the matrix $\boldsymbol{Q}$ are not processed.

Then, the "winner-take-all" procedure is carried out in every processed neural column of the network according to Equations (6)-(7). The configuration of neural activity in the k-th layer $\boldsymbol{P}^{t}(k)$ reflects the shape of the corresponding k-th texture segment at the t-th time step.

In the next iteration, the active neurons spread their excitatory influence in the corresponding layers. Other neurons of the (i, j)-th neural column, being inactive, do not influence the neural activity re-distribution within their layers. With increasing the number of time steps, the configuration of active neurons at all network layers changes less and less. The process stops after a variable number of iterations, when the configuration of active neurons becomes stable. This means that the segmentation procedure is completed. Thus, the final configuration of active neurons in all K layers are the resultant shapes of homogeneous texture segments extracted in the image as a result of the whole segmentation procedure.

Note that during the process of the network recalculation, it may turn out that a few initial segments may disappear in the corresponding neural layers. Of course, this can usually happen with relatively small segments. This statement is illustrated by the experiments shown in the next section.

7. Experiments

The computer program simulating the whole texture segmentation procedure has been designed. The program has the following basic parameters. Experiments deal with grayscale images of 427 × 320 pixels each as in [31] [32]. Texture window of 15 × 15 pixels is used. A total of $M = 22$ texture features are extracted from every texture window as it is described in Section 3.

The program has been tested on natural images of different types. The test

images have been taken from Internet and several databases, such, for example, as The Prague Texture Segmentation Datagenerator and Benchmark, MSRC_ObjCategImageDatabase, the Berkeley segmentation database, and the Stanford dataset.

Examples of texture segmentation results are shown in Figures 3-6. With these figures, we would like to give readers an impression of the main stages of the whole texture segmentation procedure in several natural images. For this, every figure is composed of six parts; all of them are in mutual coordinate correspondence.

Figures 3-6 are organized as follows. Each figure is arranged in two lines. A grayscale photograph (input image) is placed in the left corner of the top line. The middle part of the top line shows the texture segments extracted as a result of the first stage of the whole segmentation procedure.

The top right corner part presents the results of combining the segments of the same texture into the same texture class according to the merging procedure described in Section 5.

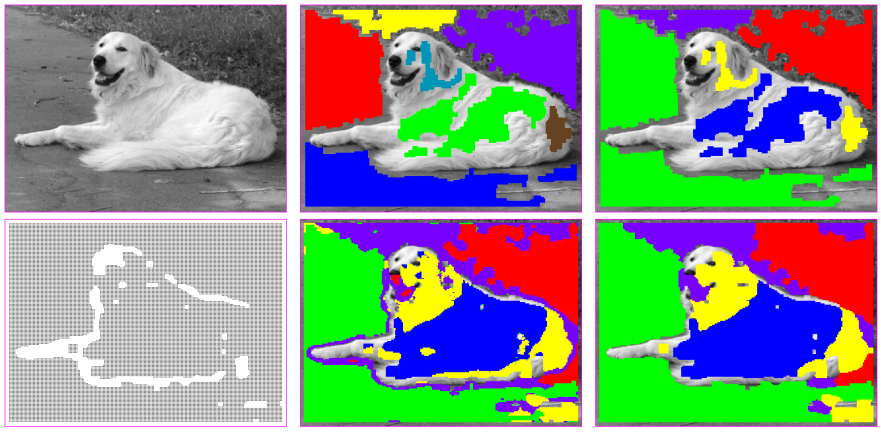

Figure 3. Grayscale photograph of the dog (the left corner part of the top line) and the main stages of its texture segmentation.

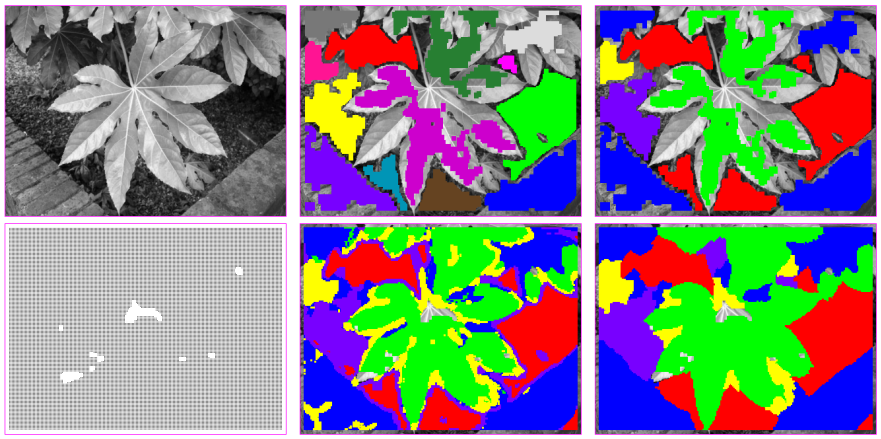

Figure 4. Grayscale photograph of the plant (the left corner part of the top line) and the main stages of its texture segmentation.

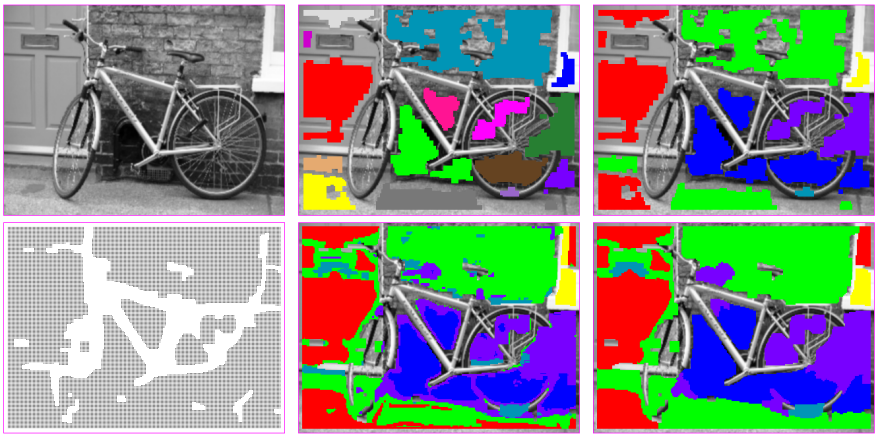

Figure 5. Grayscale photograph of a scene with a bicycle (the left corner part of the top line) and the main stages of its texture segmentation.

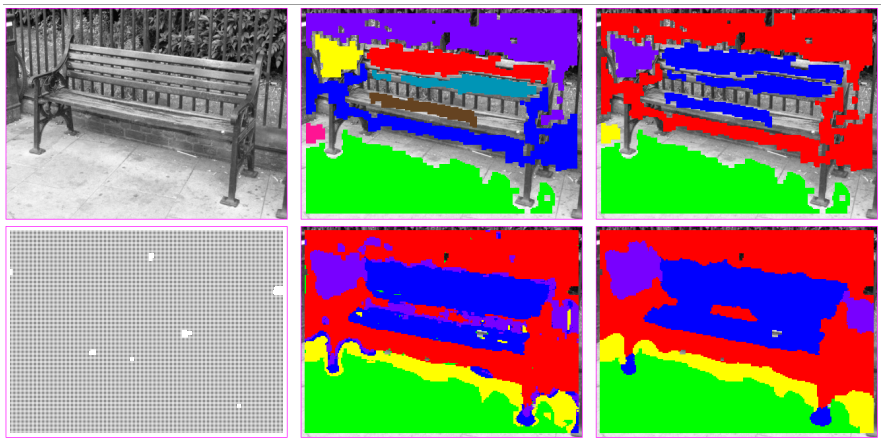

Figure 6. Grayscale photograph of the bench (the left corner part of the top line) and the main stages of its texture segmentation.

In the lower left corner part, the binary matrix $\boldsymbol{Q}$ is shown (see Section 6), which separates homogeneous fine-grained texture regions and such areas of the image, that certainly are boundary regions or regions of coarse (-grained) textures. The latter regions are not processed by the neural network with competitive layers and are painted white, while the homogeneous texture areas are indicated by dots.

The middle part of the bottom line displays (in different colors) the initial neural activity $\boldsymbol{P}^{1}(k)$ obtained in the first time step of the recalculation of the neural network with competitive layers.

The lower right corner part demonstrates the final segmentation results attained after image processing by the neural network with competitive layers.

All texture segments obtained in different stages of the segmentation procedure are displayed in different colors (over the input image as a background) in the following order: green, red, dark-blue, violaceous, yellow, light-blue, brown, pink, light-grey, dark-grey, bronze, violet, cyan, amethyst, fawn, pine green, plum, pear.

Let us underline that all texture segmentation results shown in **Figures 3-6** are attained with the same values of parameters without tuning them for each image individually.

Comparing all six parts of **Figures 3-6** separately, we may conclude the following. The experimental results of **Figures 3-6** demonstrate that the proposed segmentation procedure effectively accomplishes the extraction of homogeneous texture segments in images of different types. Doing so, the segmentation procedure usually performs rather reasonable segmentation of the input images from a human point of view.

The difference between the middle and the right corner parts of the bottom line on every of **Figures 3-6** clearly illustrates the effect of using the neural network with competitive layers.

Comparing the middle part of the top line and the lower right corner part one can see the advantages of the whole segmentation procedure in comparison with the segmentation results achieved after its first stage. These advantages, obviously, consist in a more accurate delineation of the shapes of the homogeneous texture segments present in the image.

As seen in **Figures 3-6**, the larger the texture segment, the more correctly it is extracted. And, the smaller the texture segment, the worse it is delineated. The reason for this fact is evident: it is a rather large size of the texture window (15×15 pixels) versus to the relatively small size of the image (427×320 pixels).

We also include **Figures 7-10** in a set of demonstration experiments to give the reader an opportunity to visually compare the segmentation results achieved by the proposed texture segmentation procedure and the segmentation results obtained by the graph-based image segmentation algorithm [46], which is widely used (and recognized as an effective and popular segmentation method). The original images and segmentation results produced by the graph-based image segmentation algorithm of **Figure 7** and **Figure 8** are taken from [25]; those of **Figure 9** are taken from Internet (Example Results, in the page Image Segmentation—Brown CS [48]); those of **Figure 10** are taken from [46].

Figures 7-10 are organized as follows. As above, each figure consists of six parts that are in mutual coordinate correspondence and are arranged in two lines. The original color input image is placed in the left corner part of the top line. Let us underline that since our segmentation algorithm is designed to process only grayscale images, we had to convert the original color images into the grayscale ones. The grayscale image converted from the original color one is presented in the middle part of the top line of each figure. The same grayscale image serves as a background for the left corner part and middle parts of the bottom line.

The top right corner part of each figure displays the segmentation results obtained by the graph-based image segmentation algorithm.

The texture segments extracted by our whole segmentation procedure at its first stage are shown in the left corner part of the bottom line by different colors (over the grayscale image as a background).

The middle part of the bottom line of each figure depicts the initial neural activity $\boldsymbol{P}^1$ obtained in the first time step of the image processing by the neural network with competitive layers.

The lower right corner part demonstrates the final segmentation results attained after the image processing by the neural network with competitive layers. So, this part presents the final shapes of all homogeneous texture segments that our whole texture segmentation procedure was able to extract in the analyzed image. White spots in this part of the figures correspond to the boundary regions or regions of coarse(-grained) textures that were not processed by the neural network with competitive layers.

In the figures, all texture segments are displayed in different colors in the following order: green, red, dark-blue, violaceous, yellow, light-blue, brown, pink, light-grey, dark-grey, bronze, violet, cyan, amethyst, fawn, pine green, plum, pear.

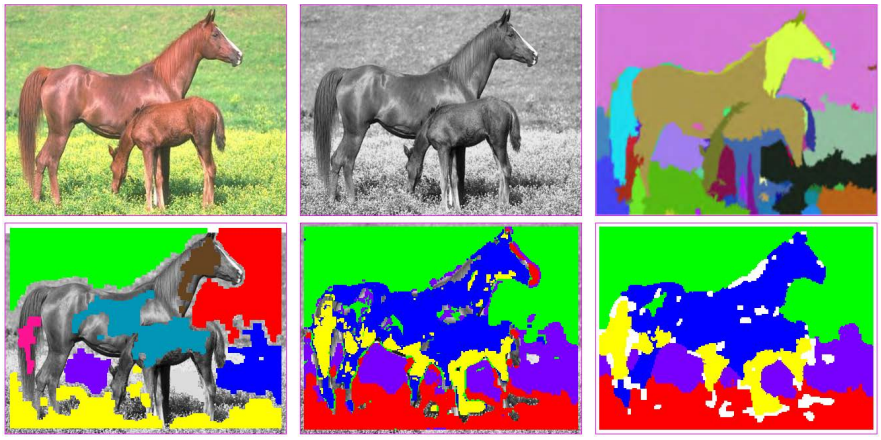

Figure 7. An original color photograph of hoses (the left part of the top line), results of its segmentation by the graph-based image segmentation algorithm and basic phases of processing of the corresponding grayscale image (converted from the original one) by the proposed texture segmentation procedure.

Figure 8. A color photo of a house, results of its segmentation by the graph-based segmentation algorithm and processing of the corresponding converted grayscale image by the proposed texture segmentation procedure.

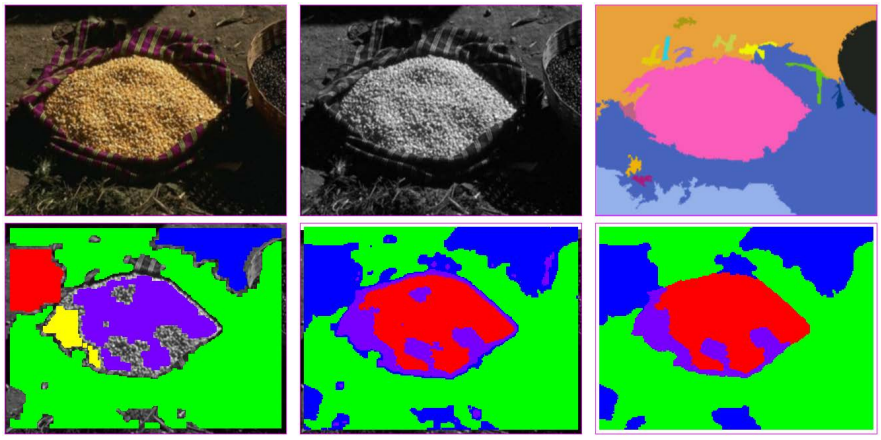

Figure 9. A color photo of a scene with seeds, results of its segmentation by the graph-based segmentation algorithm and processing of the corresponding grayscale image by the proposed texture segmentation procedure.

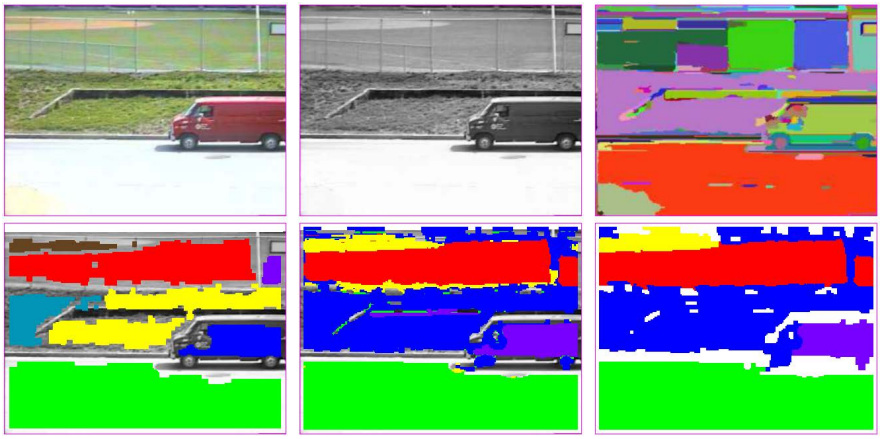

Figure 10. A color photograph of a street scene (the left part of the top line), results of its segmentation by the graph-based image segmentation algorithm and basic phases of processing of the corresponding grayscale image (converted from the original one) by the proposed texture segmentation procedure.

Note that in a series of experiments shown in **Figures 7-10**, the graph-based image segmentation algorithm takes advantage of all information available on the original color input images, while our texture segmentation procedure uses only reduced information of grayscale images converted from the color images.

8. Discussion and Conclusions

The purpose of this research is to develop a universal procedure for partial texture segmentation of any visual images. The developed procedure segments any input image into a number of non-intersecting areas of homogeneous fine-grained texture.

The main advantages of the proposed procedure are as follows. It is fully unsupervised, that is, it processes the input image without any a priori knowledge, neither of the type of textures, nor of the number of texture segments present in

the image. The procedure segments arbitrary images of all types, that is, to switch from one type of image to another, no changes to the procedure parameters are required. In particular, the segmentation results shown in Figures 3-10 are not obtained by means of tuning any parameters for each photo individually, but all these images have been processed with the same parameters.

Another important advantage of the procedure is that in most cases it extracts homogeneous fine-grained texture segments in a similar way as people do. This result is confirmed by a series of experiments demonstrating this ability of the procedure on a wide range of images; some of these experiments are presented above. Of course, not all images are segmented ideally from a human point of view, but in most cases the main (largest) segments of a homogeneous texture are delineated correctly. However, humans always use some high-level features for image segmentation, such as the context of the scene being analyzed and the knowledge of the belonging of the considered segment to a specific object that the humans previously recognized in the image. This can explain the presence in Figures 3-10 the segments that a human would not consider as separate segments in these images.

The number and shapes of the homogeneous fine-grained texture segments extracted by the segmentation procedure in the input image depend on many circumstances. One of them is the procedure's parameters. In particular, some parameters of the first stage of the segmentation procedure determine the degree of homogeneity of the texture segments that will be extracted in the images. For example, such parameters can be chosen that provide the extraction of a larger number of smaller but more homogeneous segments.

An important parameter, which strongly influences the segmentation process, is the texture window size (in relation to the image size). The larger the window size, the less homogeneous and coarser texture segments are extracted. However, a very small texture window does not adequately describe the texture.

Another circumstance, which largely determines the number and shapes of the extracted texture segments, is an internal texture structure of the image. This means that if the input image consists of separate homogeneous fine-grained texture segments with rather different texture characteristics, the segmentation algorithm reliably extracts the same segments, even if the algorithm's parameters vary in a wide range. However, if the segment is bordered with other segment with very similar texture characteristics, the procedure may extract both them as one segment.

As mentioned above, we divide the whole texture segmentation procedure into three stages. The first stage is most important, since it determines, to a large extent, the subsequent stages of the segmentation process. The first stage of the whole texture segmentation procedure is performed by means of the seeded region growing approach which is a purely local method. That is, the delineation of each texture segment is performed in turn, separately from all others, *i.e.*, without global view of the problem. And this also applies to the segments of the

same texture class present in the analyzed image. Due to this peculiarity, the remote texture segments of the same texture class cannot be assigned to the same class at the first stage of the whole texture segmentation procedure in principle. Thus, this peculiarity leads to the necessity to use some post-processing method in order to merge the texture segments of the same class (see Section 5).

The proposed segmentation procedure is not free of restrictions and disadvantages. Indeed, comparing the middle and the right corner parts of the top line in **Figures 3-6**, one can make the following conclusion. Although in many cases the merging procedure successfully combines segments belonging to the same texture class, the experimental results show that this operation is still far from perfect. Let us consider **Figure 3** as an example. As seen in the figure, the segmentation procedure could not delineate the dog's body as one texture segment, whereas, from a human point of view, all parts of the dog's body are of the same texture class. Instead of this, at the first stage of the segmentation procedure, the dog's body is divided into three different segments painted in light blue, green and brown in the top middle part of the figure. Then, in the second stage of the segmentation procedure, the proposed merging operation could not recognize that all these three segments are of the same texture; only two of them—light blue and brown were combined by this merging operation and become painted yellow, as seen in the top right corner part of the figure.

We see the problem, but we cannot solve it in the given set of texture features. Indeed, for the example of **Figure 3**, the sets of texture features of the light blue, green and brown parts of the dog's body are really too different for merging these segments into the same texture class. Of course, it is possible to reduce the requirements for similarity of segments to be merged. But in this case, the segments of textures that are different from a human point of view could be also merged by this operation. Presumably, the problem can be solved by means of using a larger number of texture features in the process of comparison between those texture segments that are candidates for merging. But, in the present work, we use the same parameters for all stages of the segmentation procedure, and, in the merge operation, we apply such parameters that definitely prevent over-merging. Note, that there is no over-merging in all figures.

At the third stage of the whole texture segmentation procedure, the neural network with competitive layers realizes more global image processing. Although, only local operations are accomplished in the network within the layers at every time step of the network recalculation, but configurations of active neurons change in the layers with each new time step. These changes in the configuration of active neurons in the layers transfer the influence of each initially active neuron, iteration by iteration, to other neurons that can be located at a longer distance than the distance of the lateral excitatory connections Z. More or less global image processing depends on the weight U of lateral intra-layer excitatory connections and the parameter Z. Thus, in general, the network with competitive layers processes the input image taking into account not only the

neighborhood information but also more global information about the entire image.

The experimental results presented in the paper are applicable only to qualitative assessment of the effectiveness of the segmentation algorithm in such terms as correct (or incorrect) extraction of segments from a human point of view. This is so because of the following reasons. The proposed texture segmentation procedure is intended to extract only homogeneous fine-grained texture segments and it cannot extract a coarse texture segment as a whole region. Moreover, it should divide every coarse texture region into a number of small homogeneous fine-grained texture segments. However, in all databases used for testing the algorithms for texture and object segmentation and recognition, the image region occupied by any object is considered as a region of the same texture and is supplied with its ground-truth. For example, in **Figure 3** the ground-truth for a dog is the whole region of its body; the ground-truth for a bicycle of **Figure 5** is the whole bike, and so on. Therefore, according to the rules of all databases, the correct segmentation results in **Figure 3** and **Figure 5** would be the extraction of the whole body of the dog and the whole bike, respectively. Since the proposed segmentation procedure divides any processed area into a number of homogeneous segments of fine-grained texture, it should be evident that no quantitative assessment of the proposed segmentation procedure can be done using these databases.

The image processing by the proposed segmentation procedure leads to a significant reduction in the uncertainty of the internal structure of the analyzed image because of the following reasons. In a typical case, the input image contains a number of homogeneous fine-grained texture segments. And all of them will be extracted by the procedure. In most cases, only a small part of the analyzed image remains unsegmented, which consists of coarse textures and boundary regions. Thus, as a minimum, the procedure provides helpful additional information about the image that might significantly facilitate the solution of such tasks as segmentation, analysis, object-ground separation and, finally, recognition of the objects present in the image.

Although the proposed image processing algorithm provides rather reasonable texture segmentation of any input images, the obtained segmentation results are still not ideal. Directions for further research include faster comparison of brightness and orientation histograms that is the key operation in our region growing procedure (see [31] and Section 2). Since the histograms are represented as binary vectors, this will allow us to use the methods for fast similarity estimation of binary vectors (e.g. [49] [50]), as well as index structures for fast similarity search (e.g. [51]). However, the next obvious step of our work will be the extension of the proposed segmentation algorithm to all three components of color images (RGB) (separately). The goal is to take advantage of very valuable information about the color of all image pixels in addition to its texture peculiarities and, due to this, achieve a more complete and accurate segmentation of the input images.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Alkama, S., Chahir, Y. and Berkani, D. (2015) Label Maps Fusion for the Marginal Segmentation of Multi-Component Images. *Neural Network World*, **25**, 405-426. https://doi.org/10.14311/NNW.2015.25.021

[2] Caenen, G., Ferrari, V., Zalesny, A. and Van Gool, L. (2002) Analyzing the Layout of Composite Textures. *Proceedings of the* 2*nd International Workshop on Texture Analysis and Synthesis*, Copenhagen, June 2002, 15-19.

[3] Donoser, M. and Bischof, H. (2008) Using Covariance Matrices for Unsupervised Texture Segmentation. *Proceedings of the* 19th *International Conference on Pattern Recognition*, Tampa, 8-11 December 2008, 1-4. https://doi.org/10.1109/ICPR.2008.4761350

[4] Malik, J., Belongie, S., Leung, T. and Shi, J.B. (2001) Contour and Texture Analysis for Image Segmentation. *International Journal of Computer Visio*, **43**, 7-27. https://doi.org/10.1023/A:1011174803800

[5] Wolf, L., Huang, X.L., Martin, I. and Metaxas, D. (2006) Patch-Based Texture Edges and Segmentation. *Proceedings of the* 9*th European Conference on Computer Vision*, Graz, 7-13 May 2006, 481-493. https://doi.org/10.1007/11744047_37

[6] Fauzi, M.F.A. and Lewis, P.H. (2003) A Fully Unsupervised Texture Segmentation Algorithm. *Proceedings of British Machine Vision Conference*, Norwich, September 2003, 519-528. https://doi.org/10.5244/C.17.53

[7] Mahbubur Rahman, M.D. (2012) Unsupervised Natural Image Segmentation Using Mean Histogram Features. *Journal of Multimedia*, **7**, 332-340.

[8] Todorovic, S. and Ahuja, N. (2009) Texel-Based Texture Segmentation. *Proceedings of the* 12*th IEEE International Conference on Computer Vision*, Kyoto, 29 September-2 October 2009, 841-848. https://doi.org/10.1109/ICCV.2009.5459308

[9] Wei, H. and Bartels, M. (2006) Unsupervised Segmentation Using Gabor Wavelets and Statistical Features In LIDAR Data Analysis. *Proceedings of the* 18*th International Conference on Pattern Recognition*, Hong Kong, 20-24 August 2006, 667-670.

[10] Yang, A.Y., Wright, J., Ma, Y. and Sastry, S.S. (2008) Unsupervised Segmentation of Natural Images via Lossy Data Compression. *Computer Vision and Image Understanding*, **110**, 212-225. https://doi.org/10.1016/j.cviu.2007.07.005

[11] Angelova, A. and Zhu, S. (2013) Efficient Object Detection and Segmentation for Fine-Grained Recognition. *Proceedings of the* 2013 *IEEE Conference on Computer Vision and Pattern Recognition*, Portland, 23-28 June 2013, 811-818. https://doi.org/10.1109/CVPR.2013.110

[12] Melendez, J., Puig, D. and Garcia, M.A. (2010) Multi-Level Pixel-Based Texture Classification through Efficient Prototype Selection via Normalized Cut. *Pattern Recognition*, **43**, 4113-4123. https://doi.org/10.1016/j.patcog.2010.06.014

[13] Goltsev, A. (1996) An Assembly Neural Network for Texture Segmentation. *Neural Networks*, **9**, 643-653. https://doi.org/10.1016/0893-6080(95)00136-0

[14] Goltsev, A. and Wunsch, D.C. (1998) Inhibitory Connections in the Assembly Neural Network for Texture Segmentation. *Neural Networks*, **11**, 951-962. https://doi.org/10.1016/S0893-6080(98)00053-7

[15] Kussul, E.M., Rachkovskij, D.A. and Baidyk, T.N. (1991) On Image Texture Recognition by Associative-Projective Neurocomputer. *Proceedings of Conference on Intelligent Engineering Systems through Artificial Neural Networks*, St. Louis, 10-13 November 1991, 453-458.

[16] Kussul, E.M., Baidyk, T.N., Lukovitch, V.V. and Rachkovskij, D.A. (1993) Adaptive Neural Network Classifier with Multifloat Input Coding. *Proceedings of the* 6*th International Conference*, Nimes, 25-29 October 1993, 209-216.

[17] Ruiz-Del-Solar, J. (2000) Neural-Based Architectures for the Segmentation of Textures. *Proceedings of the* 15*th International Conference on Pattern Recognition*, Barcelona, 6 August 2002, 1080-1083. https://doi.org/10.1109/ICPR.2000.903733

[18] Kussul, E.M., Kasatkina, L.M., Rachkovskij, D.A. and Wunsch, D.C. (1998) Application of Random Threshold Neural Networks for Diagnostics of Micro Machine Tool Condition. *Proceedings of the IEEE International Joint Conference on Neural Networks*, Anchorage, May 1998, 241-244.

[19] Lukovich, V.V., Goltsev, A.D. and Rachkovskij, D.A. (1997) Neural Network Classifiers for Micromechanical Equipment Diagnostics and Micromechanical Product Quality Inspection. *Proceedings of the* 5*th European Congress on Intelligent Techniques and Soft Computing*, Aachen, 8-11 September 1997, 534-536.

[20] Rachkovskij, D.A. and Kussul, E.M. (1998) DataGen: A Generator of Datasets for Evaluation of Classification Algorithms. *Pattern Recognition Letters*, **19**, 537-544. https://doi.org/10.1016/S0167-8655(98)00053-1

[21] Frolov, A.A., Husek, D. and Rachkovskii, D.A. (2006) Time of Searching for Similar Binary Vectors in Associative Memory. *Cybernetics and Systems Analysis*, **42**, 615-623. https://doi.org/10.1007/s10559-006-0098-z

[22] Gol'tsev, A.D. (1991) Structured Neural Networks with Learning for Texture Segmentation in Images. *Cybernetics and Systems Analysis*, **27**, 927-936. https://doi.org/10.1007/BF01246527

[23] Mobahi, H., Rao, S.R., Yang, A.Y., Sastry, S.S. and Ma, Y. (2011) Segmentattion of Natural Images by Texture and Boundary Compression. *International Journal of Computer Vision*, **95**, 86-98. https://doi.org/10.1007/s11263-011-0444-0

[24] Wang Y. and Xu L.H. (2018) Unsupervised Segmentation of Greenhouse Plant Images Based on Modified Latent Dirichlet Allocation. *PeerJ*, **6**, e5036. https://doi.org/10.7717/peerj.5036

[25] Hisashi, S. (2017) Automatic Color Image Segmentation Using a Square Elemental Region-Based Seeded Region Growing and Merging Method. *arXiv preprint*, 1711.09352.

[26] Long, P., Lu, H.X. and Wang, A. (2018) A Novel Unsupervised Two-Stage Technique in Color Image Segmentation. *Chinese Journal of Electronics*, **27**, 405-412. https://doi.org/10.1049/cje.2018.01.011

[27] Ontrup, J. and Ritter, H. (1998) Perceptual Grouping in a Neural Model: Reproducing Human Texture Perception. University of Bielefeld, Bielefeld, 59 p.

[28] Wersing, H., Steil, J.J. and Ritter, H. (2001) A Competitive-Layer Model for Feature Binding and Sensory Segmentation. *Neural Computation*, **13**, 357-387. https://doi.org/10.1162/089976601300014574

[29] Ritter, H. (1990) A Spatial Approach to Feature Linking. *Proceedings of the International Neural Network Conference*, Paris, 9-13 July 1990, 898-901. https://doi.org/10.1007/978-94-009-0643-3_124

[30] Goltsev, A. and Gritsenko, V. (2013) Algorithm of Sequential Finding the Characte-

ristic Features of Homogeneous Texture Regions for the Problem of Image Segmentation. *Cybernetics and Computer Engineering*, **173**, 25-34.

[31] Goltsev, A., Gritsenko, V. and Húsek, D. (2017) Extraction of Homogeneous Fine-Grained Texture Segments in Visual Images. *Neural Network World*, **27**, 447-477. https://doi.org/10.14311/NNW.2017.27.024

[32] Goltsev, A., Gritsenko, V., Kussul, E. and Baidyk, T. (2015) Finding the Texture Features Characterizing the Most Homogeneous Texture Segment in the Image. In: Rojas, I., Joya, G. and Catala, A., Eds., *Advances in Computational Intelligence. IWANN* 2015. *Lecture Notes in Computer Science*, Vol. 9094, Springer, Cham, 287-300.

[33] Goltsev, A., Rachkovskij, D. and Wunsch, D. (2000) A Neural Network for Segmentation of Line Drawings into Lines of Different Orientations. *Intelligent Engineering Systems through Artificial Neural Networks*, **10**, 217-224.

[34] Goltsev, A. and Rachkovskij, D. (2001) A Recurrent Neural Network for Partitioning of Hand Drawn Characters into Strokes of Different Orientations. *International Journal of Neural Systems*, **11**, 463-475. https://doi.org/10.1142/S0129065701000862

[35] Cui, W.H., Guan, Z.Q. and Zhang, Z.Y. (2008) An Improved Region-Growing Algorithm for Image Segmentation. *Proceedings of International Conference on Computer Science and Software Engineering*, Vol. 6, Wuhan, 12-14 December, 2008, 93-96. https://doi.org/10.1109/CSSE.2008.891

[36] Fan, J.P., Zeng, G.H., Body, M. and Hacid, M.S. (2005) Seeded Region Growing: An Extensive and Comparative Study. *Pattern Recognition Letters*, **26**, 1139-1156. https://doi.org/10.1016/j.patrec.2004.10.010

[37] Kamdi S. and Krishna, R.K. (2012) Image Segmentation and Region Growing Algorithm. *International Journal of Computer Technology and Electronics Engineering*, **2**, 103-107.

[38] Mancas, M., Gosselin, B. and Macq, B. (2005) Segmentation Using a Region-Growing Thresholding. *Image Processing*: *Algorithms and Systems IV*, San Jose, CA, 17 January 2006, 388-398.

[39] Peng, Q. (2015) An Image Segmentation Algorithm Research Based on Region Growth. *Journal of Software Engineering*, **9**, 673-679. https://doi.org/10.3923/jse.2015.673.679

[40] Qin, A.K. and Clausi, D.A. (2010) Multivariate Image Segmentation Using Semantic Region Growing with Adaptive Edge Penalty. *IEEE Transactions on Image Processing*, **19**, 2157-2170. https://doi.org/10.1109/TIP.2010.2045708

[41] Wan, S.Y. and Higgings, W.E. (2003) Symmetric Region Growing. *IEEE Transactions on Image Processing*, **12**, 1007-1015. https://doi.org/10.1109/TIP.2003.815258

[42] Kim, T.H., Lee, K.M. and Lee, S.U. (2013) Learning Full Pairwise Affinities for Spectral Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35**, 1690-1703. https://doi.org/10.1109/TPAMI.2012.237

[43] Li, Z.G., Wu, X.M. and Chang, S.F. (2012) Segmentation Using Superpixels: A Bipartite Graph Partitioning Approach. *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, Providence, 16-21 June 2012, 789-796. https://doi.org/10.1109/CVPR.2012.6247750

[44] Ojala, T. and Pietikainen, M. (1999) Unsupervised Texture Segmentation Using Feature Distributions. *Pattern Recognition*, **32**, 477-486. https://doi.org/10.1016/S0031-3203(98)00038-7

[45] Comaniciu, D. and Meer, P. (2002) Mean Shift: A Robust Approach toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**, 603-619. https://doi.org/10.1109/34.1000236

[46] Felzenszwalb, P.F. and Huttenlocher, D.P. (2004) Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, **59**, 167-181. https://doi.org/10.1023/B:VISI.0000022288.19776.77

[47] Jähne, B., Scharr, H. and Körkel, S. (2000) Principles of Filter Design. In: Jähne, B., Scharr, H., Körkel, S., Jähne, B., Haussecker, H. and Geissler, P. Eds., *Handbook of Computer Vision and Applications*, *Vol.* 2: *Signal Processing and Pattern Recognition*, Academic Press, San Diego, 125-152.

[48] http://cs.brown.edu/~pff/segment/

[49] Gritsenko, V.I. and Rachkovskij, D.A. (2019) Methods for Vector Representation of Objects for Fast Similarity Estimation. Naukova Dumka, Kiev. (In Russian)

[50] Rachkovskij, D.A. (2017) Binary Vectors for Fast Distance and Similarity Estimation. *Cybernetics and Systems Analysis*, **53**, 138-156. https://doi.org/10.1007/s10559-017-9914-x

[51] Rachkovskij, D.A. (2017) Index Structures for Fast Similarity Search for Binary Vectors. *Cybernetics and Systems Analysis*, **53**, 799-820. https://doi.org/10.1007/s10559-017-9983-x