An Algorithm for Measuring the Similarity of Histograms for Texture Image Segmentation

ALEXANDER GOLTSEV[1], OLEKSII HOLTSEV[2]
[1]Department of Neural Information Processing Technologies,
International Research and Training Centre for Information Technologies & Systems
of the NAS & MES,
Acad. Glushkov ave., 40, Kiev, 03187,
UKRAINE

[2]Department of Digital Environmental Monitoring Systems,
International Research and Training Centre for Information Technologies & Systems
of the NAS & MES,
Acad. Glushkov ave., 40, Kiev, 03187,
UKRAINE

Abstract: - A simple algorithm for measuring the similarity between multi-column histograms is presented. The proposed algorithm is intended for texture segmentation of images using histograms as texture features. The purpose of developing such a specialized algorithm is to more accurately determine the boundaries between neighboring texture segments. The algorithm is specially designed so that to express the similarity value as a percentage. The main peculiarity of the proposed algorithm is that when calculating the similarity value, it considers not only the corresponding histogram columns but also takes into account their neighboring components. Due to this, the algorithm more adequately evaluates the similarity of histograms. The proposed algorithm was implemented as a computer program as an integral part of the image segmentation model. The efficiency of the histogram comparison algorithm is indirectly confirmed by the texture segmentation results of the image segmentation model in image processing experiments.

Key-Words: - Image processing, Similarity of histograms, Texture features, Texture segmentation.

Received: May 14, 2023. Revised: December 15, 2023. Accepted: January 11, 2023. Published: February 20, 2024.

1 Introduction

This paper considers the problem of evaluating the similarity between histograms, which are used as texture features for the task of dividing an image into texture segments. The problem of texture segmentation of images is a key one for the analysis of natural visual scenes of various natures: landscapes, satellite photographs, and medical images used by medical professionals in diagnosing diseases.

The problem of texture segmentation of images has different complexity depending on the amount of information available about the processed image. For example, the solution to this problem is greatly facilitated, if the number of texture segments present in the image is indicated. Also, the task is greatly simplified if samples of the texture segments, that need to be extracted from the image, are provided. Using this information, the parameters of the segmentation algorithm can be appropriately tuned, for example, through training. This approach belongs to the category of supervised learning, it is presented in a significant number of publications, [1], [2], [3], [4], [5], [6], [7], [8], [9], [10].

Another approach to the texture segmentation problem implies that there is no predetermined set of texture classes, and the segmentation algorithm performs the extraction of texture regions without training, using some universal texture features. This approach belongs to the category of unsupervised texture segmentation, [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21, [22], [23].

The ability to evaluate the similarity between sets of features is one of the foundations of both natural and artificial intelligence. Calculation of the measure of similarity between sets of features is used to solve the texture segmentation task. For solving the similarity search problem (proximity search, best match retrieval), many methods have been developed that determine the similarity between such objects as vectors, sequences, trees, and graphs, [24], [25], [26], [27], [28].

The closest to the topic of this work are methods for determining the similarity/dissimilarity of vectors of large dimensions. Distance is a measure of the "dissimilarity" or "difference" of vectors - for example, the Euclidean distance, Manhattan distance, Hamming distance, Minkowski distance, Mahalanobis distance, Bulldozer's distance (earth mover's distance, EMD (see exhaustive review, [29]).

However, these well-known methods are difficult to use in the texture segmentation task. That's why we have tried to construct an efficient algorithm for comparison between histograms that can be easily applied to determine the similarity/dissimilarity of texture segments when solving the problem of texture segmentation.

The algorithm is based on the idea of taking into account not only the corresponding components of histograms but also the components of their close surroundings.

2 Histograms as Texture Features

To evaluate the texture characteristics of different areas of the image, texture windows of the same size are used that cover the entire image (with overlaps). The texture characteristics measured in the texture windows serve to assess the similarity/difference between testing areas of the image.

We use, first of all, the histogram of the brightness of all pixels of the texture window as texture features. Any image is an integer matrix, each element of which represents the brightness value of the corresponding image pixel. In black and white images, the brightness range is 0 – 255. So, the brightness histogram consists of 256 columns, according to the brightness range of the pixels in the image. The height of each histogram column represents the number of pixels in the texture window that have corresponding brightness values. The maximum height of the columns is equal to the number of pixels in the texture window. The texture windows of 15×15 pixels were used in the experiments. Therefore, the maximum height of the histogram column is 225.

At first glance, the concept of a histogram is equal to the concept of a vector. However, this is not quite so. The fact is that the values of the vector components, in the general case, are independent of each other and can take any values. Unlike a vector, the values of the histogram components are interconnected in such a way that the sum of all components of the histogram is constant. This interdependence of the histogram components makes it possible to somewhat improve the method for calculating the similarity between histograms in comparison with the methods for calculating the similarity between vectors.

The brightness histogram adequately describes the distinguishing texture peculiarities of a homogeneous segment of a fine-grained texture, providing its non-unique, but very informative description. The histogram remains invariant when changing the coordinates of the texture window inside a homogeneous texture segment. Also, in all natural images, the histograms of different texture segments differ significantly from each other.

3 General Description of the Texture Segmentation Algorithm

In [19], [20], [21], [22], a texture segmentation algorithm is described, which extracts all homogeneous fine-grained texture segments sequentially, in an iterative process. In each iteration, first of all, the initial seed point, belonging to the most homogeneous texture segment present in the image, is detected. Subsequently, this seed point is expanded by sequentially attaching the surrounding pixels of the image to it.

A set of texture features is extracted from the initial seed point and its close surroundings and stored to perform the extraction procedure for this homogeneous texture segment. This set of features is considered characteristic (typical, representative) for this segment and is used for subsequent comparison with the surrounding areas of the image. If the set of features extracted from the testing surrounding area (pixel) coincides with the characteristic one, then this area is appended to the seed point of the extracting texture segment. The process of such a sequential comparison (with subsequent seed point pixels) continues until the boundaries of a homogeneous texture segment are reached.

4 Justification of the Relevance of the Task

Since the main texture feature used in the segmentation algorithm is the brightness histogram, the key operation of this algorithm is to evaluate the similarity between such histograms of two adjacent texture windows.

Let us consider two histograms P and D, each of which consists of the same number of components (columns) I (i = 1, 2, …, I). Let us denote the heights of the i-th columns of both histograms as P_i and D_i, accordingly. The value of the intersection

DOI: 10.37394/23209.2024.21.11

between any two histogram columns is designated by Δ, and the intersection between the corresponding columns P_i and D_i – by Δ_i. Δ_i is equal to the smallest height of the two columns P_i and D_i, i.e., $\Delta_i = \min(P_i, D_i)$. Alternatively, Δ_i can also be represented by the formula

$$\Delta_i = (P_i + D_i - |P_i - D_i|) / 2, \tag{1}$$

where $i = 0, 1, 2, ..., I$.

The similarity of the histograms P and D calculated by different methods will be denoted by the letter R, and the similarity calculated by summing the intersections of the corresponding columns will be $R^{(0)}$

$$R^{(0)} = \sum_{i=0}^{I} \Delta_i , \tag{2}$$

The need to develop a specialized algorithm for the calculation of the measure of similarity between histograms is substantiated as follows.

Figure 1 (Appendix) shows an example of two brightness histograms of two texture windows belonging to different textures. The centers of the windows are marked with white squares. Namely, the upper histogram belongs to the "grass" texture, whereas the lower one corresponds to the "asphalt" texture.

As can be seen from Figure 1 (Appendix), there are no intersections between the histogram columns at all. Therefore, these histograms are completely different. For the texture segmentation procedure, the degree of similarity between them should be equal to zero. However, well-known methods for assessing the similarity/difference between vectors give very significant similarity values between such histograms.

Below we consider only the Euclidean distance which is the simplest measure of similarity/dissimilarity between multi-column vectors. The Euclidean distance (Dist) is calculated by the formula

$$\text{Dist} = \sqrt{\sum_{i=0}^{I} \left(P_i - D_i\right)^2}, \tag{3}$$

where $i = 0, 1, 2, ..., I$.

The Euclidean distance between the histograms shown in Figure 1 (Appendix) is equal to 82.41. To convert this value into a similarity percentage between histograms, we use the following reasoning.

The maximum difference (distance) Dist_{max} between the brightness histograms of two texture windows occurs when one texture window is located on the black area of the image, and the second window is located on the white one. In this case, in the histogram of the black texture window, only one component with index 0 will have maximum value.

In the histogram of the white texture window, a single non-zero column (with maximum height) will correspond to the component with index 255. These two columns will have the same height, equal to the number of pixels in the texture window. Texture windows consisting of 15 × 15 = 225 pixels were used in the experiments. Therefore, according to Eq. (3), the maximum Euclidean distance (i.e., the maximum difference value) between these vectors is $\text{Dist}_{max} = \sqrt{(225^2 + 225^2)} = 318.2$. The minimum Euclidean distance, of course, is equal to zero ($\text{Dist}_{min} = 0$), which corresponds to complete 100% similarity. So, the similarity percentages between histograms are calculated as follows.

If the Euclidean distance between the histograms is 82.41, then the percentage difference between them is determined by the value Dist = (Dist / Dist_{max}) 100% = (82.41 / 318.2) 100% = 25.9%. Accordingly, the percentage of similarity between the histograms is 100% – Dist = 100% – 25.9%, = 74.1%. Thus, there is a significant difference between the desired percentage of similarity – 0% and the actual percentage of similarity calculated based on the Euclidean distance – 74.1%. With such a high percentage of similarity, it would be difficult for the segmentation algorithm to separate the “asphalt” and “grass” texture segments. At the same time, it is evident that if the percentage of similarity between these histograms was 0%, it would be much easier for the segmentation algorithm to find the boundary between these texture regions.

5 Actual Description of the Histogram Comparison Algorithm

The simplest method for calculating similarity between histograms is a pair-wise comparison of all corresponding columns of the histograms. However, implementing this method is only a valid solution for low-dimensional histograms. For multi-column histograms. For multi-column histograms, the total size of the intersection between the corresponding histogram columns does not adequately evaluate the similarity between them.

The proposed algorithm for histogram similarity estimation is based on the obvious idea of taking into account not only the corresponding components of both histograms but also the components of their close surroundings.

The algorithm consists of successive comparisons of each i-th component of the histogram P - column P_i, with (1 + 2M) components of the histogram D, where M is the radius of the close surroundings of each i-th component of the histogram P. Let us denote the similarity of the histograms P and D by $R^{(M)}$, at the beginning, $R^{(M)} = 0$.

At each comparison act, the i-th component of the histogram P_i is compared with the component of the histogram D, which is shifted by a certain number of indexes to the right or left relative to i. If we denote the shift number by m, then the P_i component is compared with the D_j component, where $j = i+(-)m$, $m = 0, 1, 2, 3, \ldots, M$.

Before comparing the P_i and D_j components, the height D_j is reduced by multiplying it by some coefficient K_m, $K_m < 1$. The intersection size Δ (between the P_i component and the D_j component) is used in the process of calculating the similarity between histograms $R^{(M)}$. That is, the more $m=|i-j|$, the less impact the D_j component should have on the value $R^{(M)}$. For this purpose, the coefficients $K_m < 1$ are introduced, which reduce the proportion of the size of the intersection Δ between the compared columns of both histograms.

Reducing coefficients $K_0, K_1, K_2, \ldots, K_m, \ldots, K_M$ correlate with each other as follows: $K_0 > K_1 > K_2 > K_3 > \ldots K_m > \ldots > K_M$, whereas $K_0 = 1$. When comparing the components P_i and D_j, the size of the intersection between them Δ_m is calculated according to the following description. If $P_i \geq D_j K_m$, then the size of their intersection is $\Delta_m = D_j K_m$. If $P_i < D_j K_m$, then $\Delta_m = P_i$. Thus, for each component P_i, $(1 + 2M)$ values of Δ_m are calculated. Among these $(1 + 2M)$ values of Δ_m, the maximum value is chosen, denoted by Δ^i_{max}:

$$\Delta^i_{max} = \operatorname*{MAX}_{m=0}^{1+2M} (\Delta_m) \tag{4}$$

If, $\Delta^i_{max} \geq P_i$, then P_i is added to the similarity value of histograms $R^{(M)}$. If, however, $P_i > \Delta^i_{max}$, then the value of Δ^i_{max} is added to $R^{(M)}$.

Thus, in the general case, the similarity value of histograms is determined by summing the intersections of all maximum values Δ^i_{max} calculated for each component P_i, $i = 1, 2, \ldots, I$, with the corresponding P_i values. The similarity value of histograms can be represented by the following formula (see Eq. (1) for reference)
where $i = 0, 1, 2, \ldots, I$.

$$R^{(M)} = \sum_{i=0}^{I} \left(\Delta^i_{max} + P_i - | \Delta^i_{max} - P_i | \right) / 2, \tag{5}$$

The radius of close surroundings M depends on the size of the histogram I and on the initially specified trend of the algorithm. The larger the radius M, the more pronounced the tendency of the algorithm to consider the compared histograms as similar. On the contrary, as M decreases, the tendency to consider histograms dissimilar increases.

For texture segmentation processing, it is desirable to get the similarity between histograms of texture windows as a percentage. That is, if the percentage of similarity is small, then the compared texture windows should be attributed to different textures. Conversely, if the similarity percentage approaches 100%, then these texture windows should be classified as belonging to the same texture. Let us denote the percentage of histogram similarity as S.

In the case of comparing identical histograms, the sum of all intersections of the corresponding components of the histograms $R^{(0)}$ will be equal to the sum of all components of each of the histograms, which is equal to 225 in the case of the brightness histogram.

When comparing the brightness histograms, by pair-wise comparison of all their corresponding columns, the similarity value should be determined by dividing the total intersection size $R^{(0)}$ by the total intersection size between the same histograms P and P or D and D, which in both cases is equal to 225. The corresponding formula is $S = R^{(0)} / 225$.

It should be clear that when taking into account the close surroundings of the components, the percentage of similarity between the histograms should be calculated using a similar formula. Namely, by dividing $R^{(M)}$ (see Eq. (4)), by the same total size of the intersection between equal histograms P and P or D and D, which is 225. That is, $S = R^{(M)} / 225$. Obviously, in the case of comparing identical histograms, $S = 1$. When this number is multiplied by 100, the similarity value between histograms becomes expressed as a percentage, where 100% means complete sameness.

Usually, when comparing completely different histograms, there are no intersections between their columns at all, i.e. $R^{(0)} = 0$ (see the example in Figure 1, Appendix). Most often, in this case, the similarity value between such histograms, calculated by the proposed algorithm $R^{(M)}$, also becomes equal to zero, accordingly, $S = 0\%$. Thus, the range of S is 0% – 100%.

6 Experimental Comparison of Similarity Measures of Brightness Histograms

To illustrate the properties of the proposed algorithm for measuring the percentage of similarity between brightness histograms, the following experiments were carried out.

In the image shown in Figure 1 (Appendix), pairs of texture windows were located in segments of different textures. As can be seen in Figure 1 (Appendix), the histograms of these texture windows differ significantly from each other, mainly in that the non-zero components of both histograms are located at mismatched positions within the histograms. From the point of view of common sense and a functional point of view (for solving the problem of texture segmentation), these histograms should be qualified as completely different. And, accordingly, the algorithm proposed in this paper specially designed for efficient separation of texture segments, estimates the percentage of similarity between them equal to zero.

In a series of experiments, pairs of texture windows belonging to the same texture were also considered. Corresponding brightness histograms were formed, which are shown in Figure 2 (Appendix). As can be seen in Figure 2 (Appendix), the non-zero components of both histograms occupy approximately the same position within the histograms, and the corresponding columns have significant intersections.

The Euclidean distance between these histograms calculated by Eq, (3) is 48.0, and the percentage of similarity, converted from this number, is 85%. The similarity value of the histograms shown in Figure 2 (Appendix), calculated by pair-wise component comparison according to Eq. (2) and denoted by $R^{(0)}$, is equal to 57%.

In the process of calculating the percentage of similarity according to the proposed algorithm, the radius of the close surroundings M was limited to 10. The reducing coefficients K_1, K_2, K_3,…, K_m ,…, K_{10} were calculated by the formula $K_m = (1 – 0.1\ m)$. The use of such coefficients led to the following result. The percentage of similarity of the histograms produced by the proposed algorithm is 77%.

Thus, the proposed algorithm produces the percentage of histogram similarity $R^{(M)}$ which is increased by about 20% in comparison with $R^{(0)}$. (All mentioned percentages are the result of averaging over a series of experiments.)

The following inference follows from the experiments. For the algorithm proposed in this paper, the percentage of similarity between histograms of different textures (such as "asphalt" and "grass") is 0%, and the percentage of similarity between histograms of the same textures (such as "grass" and "grass") is in the range of 60%. – 80%.

That is, the diapason in the percentage of similarity from different textures to the same ones is approximately 70%. At the same time, for the algorithm based on the Euclidean distance, the percentage of similarity between the histograms of different textures (such as "asphalt" and "grass") is 74%. The percentage of similarity for the same textures (such as "grass" and "grass") is 85%. That is, for the algorithm based on the Euclidean distance, the difference in the similarity percentage between similar textures is only 11%.

Thus, the difference in the similarity percentage between different textures with the similarity percentage between similar textures for the proposed algorithm is 70% versus 11% for the algorithm based on the Euclidean distance. This excess of the range of percentage differences of the proposed for the algorithm based on the Euclidean distance. This excess of the range of percentage differences of the proposed algorithm over that of the Euclidean distance algorithm makes the advantage of the proposed algorithm obvious.

In other words, the proposed algorithm turns out to possess greater sensitivity when it comes to the detection of histogram dissimilarities in the context of texture comparison: histograms of similar textures would still be detected as similar, whereas comparison of dissimilar texture histograms would produce a rather low or even zero similarity percentage value indicating textures being dissimilar.

7 Conclusion

A simple algorithm for measuring the similarity between histograms is presented. The algorithm is intended for texture segmentation of images using brightness histograms as texture features. It is specially designed so that to express the measure of similarity as a percentage. The algorithm was used as an integral part of the texture segmentation models, [19], [20], [21], [22].

The reason for developing a specialized algorithm for measuring the similarity between brightness histograms is the degree of similarity/difference between them. In contrast, the proposed algorithm provides a 100% diapason of percentage similarity between the histograms being compared (from complete similarity to complete difference). This is the main advantage of the algorithm and its contribution/novelty that makes it possible to attain a more accurate determination of the boundaries between the texture segments present in the analyzed image.

The proposed algorithm is not complicated. Of course, it is not as simple as the initial version, in which the histograms were compared in a pair-wise manner, namely by calculating the sum of the intersections of the corresponding histogram columns (having the same indices). Nevertheless, in the software implementation of the proposed algorithm, only addition, subtraction, and multiplication operations are used, due to that the algorithm is fast and computationally effective.

The efficiency of the algorithm for texture segmentation of images into homogeneous texture regions is confirmed by the segmentation results in experiments on processing different natural images. The results obtained in the experiments demonstrate the effectiveness of the segmentation algorithm and show that this algorithm performs correct (from a human point of view) texture segmentation of a wide range of images, [19], [20], [21], [22]. Thus, the effectiveness of the key operation of the segmentation algorithm, the histogram comparison algorithm, is also indirectly confirmed.

Figure 3 (Appendix) is presented here, as an example which demonstrates the results of the texture segmentation of a natural image (upper part of the figure). At the bottom of the figure, the largest homogeneous texture segments extracted by the segmentation algorithm are shown in different colors. Areas of the image containing small texture areas and borders between texture segments are marked in white.

One of the interesting directions of future research is connections with the approaches developed in the framework of Hyper Dimensional Computing, [24], [25], [26], [27], [28].

References:

[1] E. M. Kussul, D. A. Rachkovskij, T. N. Baidyk, On image texture recognition by associative-projective neurocomputer, *Proceedings of the Intelligent Engineering Systems through Artificial Neural Networks (ANNIE'91),* St. Louis, Missouri, U.S.A., 1991, pp. 453–458.

[2] E.M. Kussul, T.N. Baidyk, V.V. Lukovitch, D.A. Rachkovskij, Adaptive neural network classifier with multifloat input coding, *Proceedings of the 6th International Conference "Neuro-Nimes 93",* Nimes, 1993, pp. 209–216.

[3] J. Ruiz-del-Solar, Neural-based architectures for the segmentation of textures, *Proceedings of the 15th International Conference on Pattern Recognition,* Barcelona, Spain: 2000, Vol. 3, pp. 1080–1083, http://dx.doi.org/10.1109/ICPR.2000.903733.

[4] D.R. Martin, C.C. Fowlkes, J. Malik, Learning to detect natural image boundaries using local brightness, colour, and texture cues, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 5, 2004, pp. 530-549.

[5] F.H.C. Tivive, A. Bouzerdoum, Texture classification using convolutional neural networks, *Proceedings of IEEE Region 10 Conference,* Hong Kong, China, 2006, pp. 1–4, http://dx.doi.org/10.1109/TENCON.2006.343944.

[6] O.S. Al-Kadi, Supervised texture segmentation: A comparative study, *Proceedings of the IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT),* Amman, Jordan, 2011, pp. 1–5, http://dx.doi.org/10.1109/AEECT.2011.6132529.

[7] A Angelova, S. Zhu, Efficient object detection and segmentation for fine-grained recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* New Jersey, USA, 2013, pp. 811–818, https://doi.org/10.1109/CVPR.2013.110.

[8] T.H. Kim, K.M. Lee, S.U. Lee, Learning full pairwise affinities for spectral segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 35, No. 7, 2013, pp. 1690–1703.

[9] Ashwani Kumar Aggarwal, Learning texture features from GLCM for classification of brain tumor MRI images using Random Forest Classifier, *WSEAS Transactions on Signal Processing*, Vol. 18, 2022, pp. 60–63, https://doi.org/10.37394/232014.2022.18.8.

[10] V. Barrile, E. Genovese, E. Barrile, Bioengineering and geomatics: Automatic brain image segmentation using two-stage

pipeline with SNN and watershed algorithm, *WSEAS Transactions on Biology and Biomedicine*, Vol. 20, 2023, pp. 197–203, https://doi.org/10.37394/23208.2023.20.20.

[11] Fauzi, M.F., & Lewis, P.H. (2003). A Fully Unsupervised Texture Segmentation Algorithm. *British Machine Vision Conference*, Corpus ID: 1875753.

[12] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient graph-based image segmentation. *International Journal of Computer Vision*, Vol. 59, No. 2, 2004, pp. 167–181, doi: 10.1023/B:VISI.0000022288.19776.77.

[13] D.A. Clausi, H. Deng, Design-based texture feature fusion using Gabor filters and co-occurrence probabilities, *IEEE Transactions on Image Processing,* Vol. 14, No. 7, 2005, pp. 925–936.

[14] H. Wei, M. Bartels, Unsupervised segmentation using Gabor wavelets and statistical features in LIDAR data analysis, *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'2006),* Hong Kong, Vol. 1, 2006, pp. 667–670.

[15] L. Wolf, X. Huang, I. Martin, D. Metaxas, Patch-based texture edges and segmentation, *Proceedings of the 9th European Conference on Computer Vision,* Graz, Austria, 2006, Part II, pp. 481–493, doi: 10.1007/11744047_37.

[16] A.Y. Yang, J. Wright, Y. Ma, S.S. Sastry, Unsupervised segmentation of natural images via lossy data compression, *Computer Vision and Image Understanding*, Vol. 110, No. 2, 2008, pp. 212–225, http://dx.doi.org/10.1016/j.cviu.2007.07.005.

[17] S Todorovic, N. Ahuja, Texel-based texture segmentation, *Proccedings of the 12th IEEE International Conference on Computer Vision (ICCV),* Kyoto, Japan, 2009, pp. 841–848, http://dx.doi.org/10.1109/ICCV.2009.5459308.

[18] J. Melendez, D. Puig, M.A. Garcia, Multi-level pixel-based texture classification through efficient prototype selection via normalized cut, *Pattern Recognition*, Vol. 43, No. 12. 2010, pp. 4113–4123.

[19] A. Goltsev, V. Gritsenko, Algorithm of sequential finding the characteristic features of homogeneous texture regions for the problem of image segmentation. *Cybernetics and Computer Engineering*, 173, 2013, pp. 25-34 (in Russian).

[20] A. Goltsev, V. Gritsenko, E. Kussul, T. Baidyk, Finding the texture features characterizing the most homogeneous texture segment in the image, *Lecture Notes in Computer Science*, 9094, Part I, 2015, pp. 287-300, doi: 10.1007/978-3-319-19258-1_25.

[21] A. Goltsev, V. Gritsenko, D. Húsek, Extraction of homogeneous fine-grained texture segments in visual images. *Neural Network World*, Vol. 27, 2017, pp. 447-477, https://doi.org/10.14311/NNW.2017.27.024.

[22] A. Goltsev, V. Gritsenko1, D. Húsek, Segmentation of visual images by sequential extracting homogeneous texture areas, *Journal of Signal and Information Processing,* Vol. 11, 2020, pp. 75-102, https://doi.org/10.4236/jsip.2020.114005.

[23] G. Ge, A novel parallel unsupervised texture segmentation approach, *SN Applied Sciences*, Vol. 5, No. 156, 2023, https://doi.org/10.1007/s42452-023-05366-z.

[24] D.A. Rachkovskiy, S.V. Slipchenko, I.S. Misuno, E.M. Kussul, T.N. Baidyk, Sparse binary distributed encoding of numeric vectors, *Journal of Automation and Information Sciences*, Vol. 37, No. 11, 2005, pp. 47-61, https://doi.org/10.1615/J%20Automat%20Inf%20Scien.v37.i11.60.

[25] D.A. Rachkovskiy, S.V. Slipchenko, E.M. Kussul, T.N. Baidyk, Sparse binary distributed encoding of scalars, *Journal of Automation and Information Sciences*, Vol. 37, No. 6, 2005, pp. 12-23, https://doi.org/10.1615/J%20Automat%20Inf%20Scien.v37.i6.20.

[26] D. A. Rachkovskij, Shift-equivariant similarity- preserving hypervector representations of sequences, *arXiv preprint arXiv*: 2112.154753.

[27] D.A. Rachkovskij, Representation of spatial objects by shift-equivariant similarity-preserving hypervectors, *Neural Computing and Applications*, Vol. 34, No. 24, 2022, pp. 22387-22403, https://doi.org/10.1007/s00521-022-07619-1.

[28] D. Kleyko, D. A. Rachkovskij, E. Osipov, A.Rahimi, A survey on hyperdimensional computing aka vector symbolic architectures, Part II: Applications, cognitive models, and challenges. *ACM Computing Surveys*, Vol. 55, No. 9, 2023, pp. 1-52, https://dl.acm.org/doi/10.1145/3558000.

[29] M. M. Deza, E. Deza, Encyclopedia of Distances, Springer Berlin, Heidelberg, 2016,

p.756, https://doi.org/10.1007/978-3-662-52844-0.

APPENDIX

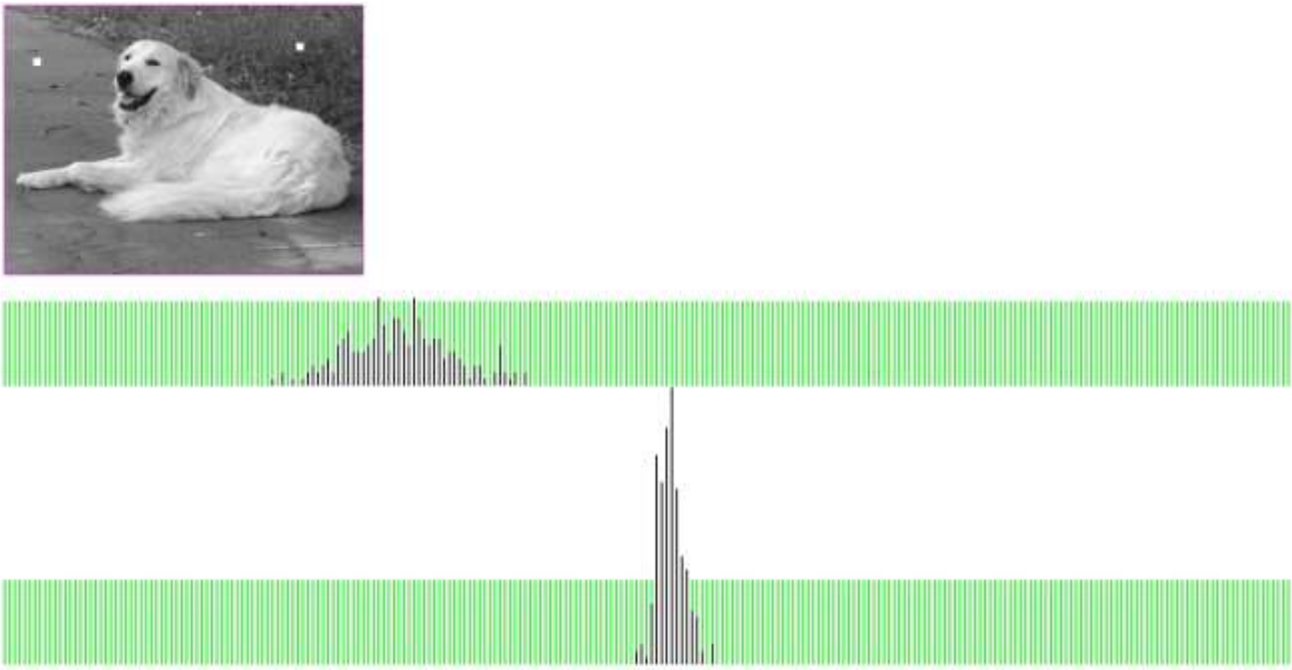

Fig. 1: Brightness histograms of two texture windows, the upper of which belongs to the "grass" texture and the other – to the "asphalt" texture

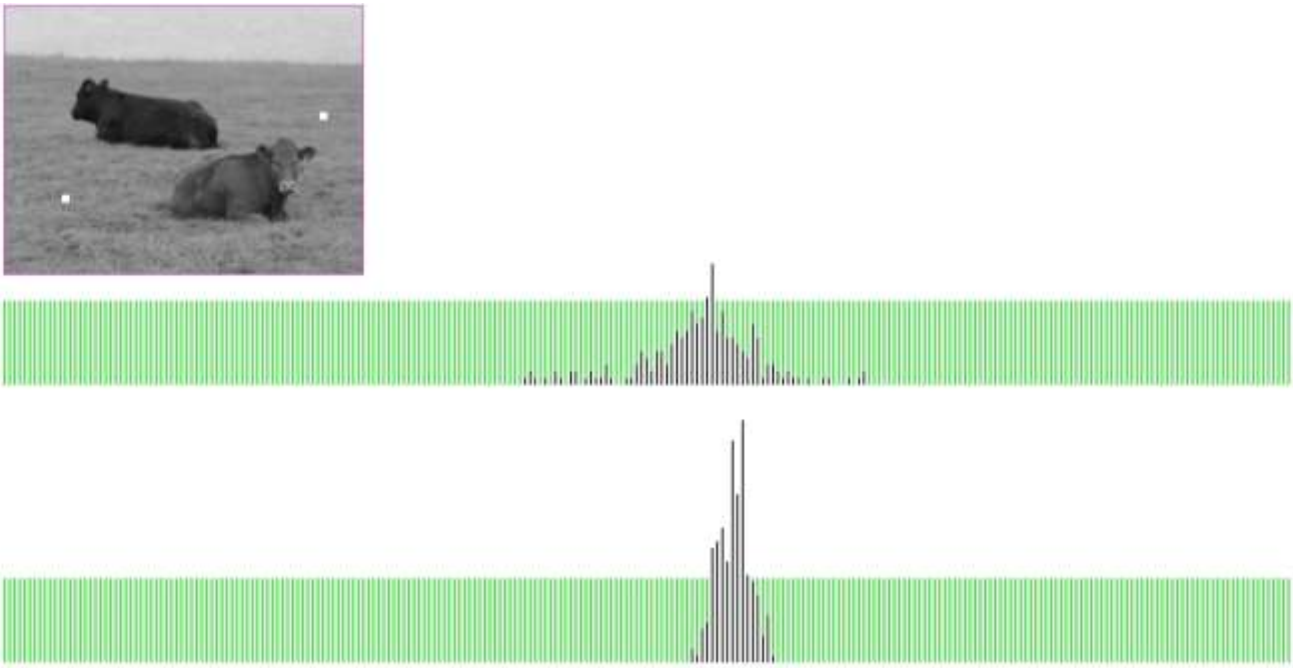

Fig. 2: Brightness histograms of two texture windows belonging to the same “grass” texture.

Fig. 3: The results of the texture segmentation on the example of processing a black-and-white image (upper part of the figure). The bottom half of the figure shows the largest texture segments highlighted in different colors

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)
The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself
No funding was received for conducting this study.

Conflict of Interest
The authors have no conflicts of interest to declare that are relevant to the content of this article.