Segmentation of earth remote sensing images based on agglomerative pixel clustering using the minimum increment of the total squared error as a decision function

I.G. Khanykov [1], V.A. Nenashev [1], M.V. Kharinov [2]
[1] State University of Aerospace Instrumentation, 67 Bolshaya Morskaya street, Saint Petersburg, 190000, Russia;
[2] Federal Research Center of Russian Academy of Sciences, 39 14th Line V.O., Saint Petersburg, 199178, Russia

Abstract

Among cluster analysis methods applied to grayscale image segmentation, Arifin's algorithm is particularly notable. This algorithm enables partitioning the original image into a number of clusters ranging from N_{max} to 1 in linear time, where N_{max} represents the number of grayscale levels in the original image. Arifin's algorithm incrementally enlarges pixel clusters by combining a *"minimum pair"* at each iteration within the calculation cycle, characterized by the minimum distance *Dist* between clusters. The original method calculates *Dist* by taking the product of interclass and intraclass variances; however, this approach involves cumbersome formulas and lacks a quality assessment for the resulting partitions. This study introduces two modifications to Arifin's algorithm that simplify the calculation of *Dist* for identifying "minimum pairs" and performing cluster merging sequences, while also enabling evaluation of cluster partition quality. The first modification uses the increment of the total squared error as a distance function for *Dist,* whereas the second modification employs partial entropy. In the first approach, partition quality is evaluated based on the total squared error or standard deviation, while in the second, it is assessed by the information content.

A computer program has been developed to implement the first modification of Arifin's algorithm, incorporating a difference formula to compute partition quality. This program has been tested using standard test images of various sizes, including full-scale aerospace images (three images at 1024×1024 pixels and one at 2050×2050 pixels). The practical significance of the modification, which leverages the increment of the total squared error to create a series of suboptimal piecewise-constant partitions, lies in reducing the number of operations per cycle needed to determine the “minimum pair” among all adjacent clusters, thereby halving processor time.

Keywords: clustering, segmentation, objective function, total squared error, variance, entropy.

Citation: Khanykov IG, Nenashev VA, Kharinov MV. Segmentation of earth remote sensing images based on agglomerative pixel clustering using the minimum increment of the total squared error as a decision function. Computer Optics 2026; 50(1): 1591. DOI: 10.18287/COJ1591.

Introduction

Image segmentation is the process of dividing an image’s pixels into groups based on a specified feature. This feature may represent the proximity of a pixel to a group of pixels in terms of brightness, color, or spatial location. More complex values, such as gradient magnitude or statistical measures, may also be used as features. Additionally, feature combinations can help determine pixel group membership.

In the context of artificial intelligence research, segmentation is a foundational task within computer vision, occupying the initial stage of image processing. Subsequent tasks, such as object detection and feature recognition, rely on segmentation outcomes. Currently, three primary segmentation types are recognized: semantic, instance and panoptic [1]. Semantic segmentation involves grouping pixels into clusters, assigning a cluster label to each pixel. Instance segmentation distinguishes each object within a class of detected objects in the image. Panoptic segmentation combines both semantic and instance segmentation approaches. Segmentation has applications [2] in areas such as video surveillance, traffic monitoring, security systems, medical diagnostics, and surgery.

The primary purpose of segmentation is to simplify the original image by creating an approximation. An open question remains as to how closely a segmented image should approximate the original or how many pixel clusters (homogeneous groups) should be included.

Among segmentation methods, a special place is held by those that generate multiple partitions into clusters and also possess a function for assessing the quality of these partitions. Clustering methods enable the generation of many partitions. All clustering methods for image segmentation, based on the principle of forming homogeneous groups, can be divided into divisive, agglomerative, and iterative.

Divisive algorithms split the original set of pixels into subgroups. Examples of such algorithms are the methods of Otsu [3], Kapur [4], and Kittler [5]. Their advantage lies in their ease of implementation, but their drawback is their high computational complexity, which increases exponentially and renders the methods practically unsuitable for grouping image pixels into four or more clusters. Algorithmically, the methods of Otsu [3], Kapur [4], and Kittler [5] are similar. The emergence of each subsequent threshold necessitates recalculating the values of the previous ones, leading to exponential growth in computational complexity.

The fundamental difference between the methods of Otsu [3], Kapur [4], and Kittler [5] lies in their approach to calculating threshold values. The Otsu method [3] uses the parameters of the area and brightness of clusters, the Kapur method [4] uses only the area, and the Kittler method [5] uses the parameters of the area of clusters, their brightness, and the deviation of brightness values.

Agglomerative methods, on the contrary, enlarge pixel clusters with each step. Examples include the methods of Ward [6] and Arifin [7]. Ward's method [6] involves exhaustive search through all pairs of clusters, resulting in quadratic computational complexity.

If the pixels are pre-grouped and ordered by brightness, the exhaustive search can be reduced to a limited search, as in Arifin's method [7], which considers only pairs of adjacent ordered clusters. Consequently, the computational complexity decreases to a linear function, making Arifin's method [7] suitable for processing very large images (Arifin's method is invariant to image size).

The material of this paper is organized as follows. Section 1 considers related papers. Section 2 elaborates on the idea of agglomerative clustering of pixels using Arifin's method and addresses the challenge of finding a way to calculate the distance between pairs of adjacent clusters to generate an optimal or near-optimal sequence of piecewise-constant partitions. Section 3 outlines the main steps of the agglomerative pixel clustering program. Section 4 presents the original method and two modified methods for calculating the distance between pairs of adjacent clusters. Section 5 compares these methods by processing a single data set with three different distance functions using a single quality function to determine the best method of distance calculation. Examples are provided of generating suboptimal series of piecewise-constant partitions for standard test images of Earth remote sensing. Section 6 presents an analysis of the obtained results. Section 7 provides a brief overview of the program implementation. Section 8 concludes the research work.

1. Related papers

Study [8] addresses the classification of Earth remote sensing images by employing superpixels formed from surface image pixels. A key issue highlighted by the authors is the presence of "mixed superpixels" near the boundaries of captured objects (perceptual units), containing pixels from multiple object classes. The occurrence of such "mixed superpixels" interferes with the recognition process, leading to misclassification regardless of the classification methods employed. In [8], the SLIC algorithm (Simple Linear Iterative Clustering) is proposed to "purify" these mixed superpixels through color quantization, dividing them into smaller, "pure" superpixels that contain pixels from a single object class. The experiments conducted in [8] utilized images from the open international BSDS500 standard test image database [9] and satellite imagery from WorldView.

The primary limitation of study [8] lies in the low processing efficiency of the software tools developed, with the authors noting that their application takes approximately 2000 seconds to complete the task.

Study [10] explores the formation of homogeneous pixel clusters without a priori information by using divisive pixel clustering. The authors encountered challenges in determining multiple threshold values and extrapolating the multithreshold processing method to color images. Previously, threshold selection methods (such as Otsu's and Kapur's methods) were applied independently to each color channel. The authors propose: (1) combining the objective functions of Otsu and Kapur with the Forest Optimization Algorithm (FOA) and Particle Swarm Optimization (PSO); and (2) using threshold values derived from these optimization algorithms to segment the color space into smaller subcubes, each evaluated as an individual cluster.

A drawback in the results of study [10] is that the homogeneous regions formed through pixel clustering inadequately represent clustered areas.

The authors of study [11] investigated the JSEG method, which utilizes color and texture characteristics of the image to define growth rules. Although JSEG operates quickly, it suffers from excessive segmentation (oversegmentation) of images. The hybrid WJSEG algorithm, which incorporates wavelet transformation, was developed to address this issue. WJSEG was tested on multispectral and panchromatic images, demonstrating improved segmentation accuracy compared to JSEG and reducing the problem of oversegmentation. However, WJSEG is sensitive to noise and exhibits computational inefficiencies. Study [11] also identifies that hybrid remote sensing image segmentation methods, including metaheuristic approaches for segmentation accuracy improvement, have high computational demands, impacting time and memory consumption. To address these issues, the authors propose a fuzzy segmentation structure implemented on a GIS platform, which mitigates computational complexity and enhances segmentation reliability. Named Fast Generalized Fuzzy C-means (FGFCM), the algorithm identifies local spatial relationships between pixels, and the Triple Center Relation Validity Index is used to determine the optimal cluster count. Study [11] reports on the testing of this method using satellite images from the southwestern region of Naples, Italy.

Study [12] addresses the detection of ship targets in radar imagery. The authors contend with the presence of speckle noise, which disrupts the effective use of Fuzzy C-means clustering, lowers detection rates, and causes clustering results to become trapped in local minima. A novel method, Block Thumbnail Particle Swarm Optimization Clustering (BTPSOC), is proposed, applying PSO to image block miniatures to avoid local minima. Results from the BTPSOC algorithm are compared to those from Convolutional Neural Networks (CNNs).

The study in [13] investigates pixel clustering for remote sensing images obtained from the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) and Thermal Mapping (TM) device. The author identifies limitations of the C-means clustering algorithm, including: (1) the lack of automatic class number determination; and (2) slow class convergence. It is noted that methods like ISODATA and ISMC enable self-iterative class number determination. However, they face challenges in adjusting distance parameters when the object size varies. In [13], a Wavelet-Feature Correlation Ratio Markov Clustering Algorithm (WFCRMCA) is proposed to differentiate pixels based on spectral similarity. This study also examines the spectral characteristics of objects, noting that, while spectra vary under different conditions, they maintain consistent characteristics within specific spectral positions. These characteristics serve as primary parameters for assessing spectral differences between remotely sensed pixels. The WFCRMCA algorithm offers statistical control over clustering accuracy. In [13], the Correlation Ratio (CR) coefficient is applied to capture similarity between two wavelet-transformed samples. When describing sharp spectral features, wavelet-correlation coefficients can distinguish pixels along spectral dimensions. Expanding the spectral range of multispectral images increases the number of characteristic points, enriching class feature sets. WFCRMCA forms the clustering space and the initial class centers with evenly selected pixels, effectively avoiding the initial parameter issues typical of K-means. This allows WFCRMCA to quickly establish optimal class centers over the entire area by iteratively reducing clustering scale and temperature.

In [14], a semantic segmentation algorithm based on the artificial neural network Bisenet is employed to improve aircraft positioning accuracy. The approach incorporates not only individual images but also a segmented panoramic view of the area.

Study [15] employs various machine learning methods to identify errors in aircraft onboard systems, utilizing clustering algorithms like DBSCAN and K-means.

In [16], a modification of multithreshold processing technology is presented, which reduces the number of operations per calculation cycle by half, effectively doubling performance. This technology was tested on the standard test image "Lena".

Study [17] applies segmentation to develop a method for selecting a safe landing site for an unmanned aerial vehicle. The main limitation of this method is the extended system training time required.

Study [18] explores search algorithms for video streams, where segmentation is based on a sliding average with variable window sizes. The primary advantage of this approach is its simplicity of use, though it requires parameter adjustments for each specific task class.

Study [19] utilizes image binarization and angle determination techniques to identify object contours in selective laser melting technology.

The research [20, 21] presents results from integrating optical and radar images of Earth's surface using contour analysis and pixel clustering methods in a multi-position onboard system.

Study [22] explores combining Gaussian mixture models (GMM) with unsupervised deep learning techniques. The standard GMM uses a Gaussian distribution to model pixel values. Region values are determined by approximating the parameter vector, minimizing the negative log-likelihood function (NLL) relative to the GMM. The authors of study [22] first noted that the iterative expectation-maximization (EM) method is typically used for this approximation. However, they identified a key limitation: GMMs assume independence between pixels, failing to account for correlations among neighboring pixels. The study proposes replacing the EM algorithm with convolutional neural networks (CNNs), enabling direct parameter estimation from the image. According to [22], this modification can rapidly predict label probabilities compared to iterative optimization methods. The expectation step in the EM algorithm's iterative process is substituted with a gradient step based on network parameters. Notably, the CNN is trained on the original image, which is subsequently segmented, eliminating the need for extensive a priori image data.

The study in [23] presents a technology for improving the accuracy of real image segmentation by integrating well-known algorithms such as K-means and ISODATA with a newly developed segmentation method. The proposed method utilizes a histogram of the synthesized interference image, with the binarization threshold defined as the arithmetic mean of the two maxima in the histogram. The synthetic interference image is generated using a double stochastic model, which calculates the correlation coefficients of each pixel in two directions: along rows and columns. An advantage of this model is its ability to analyze the impact of arbitrarily defined object shapes. However, the model includes parameters whose effects on the segmentation process require separate investigation. Additionally, exploring the use of divisive and agglomerative clustering methods to determine thresholds for segmenting the histogram of the synthesized interference image into clusters is of interest.

Study [24] focuses on cylindrical images, such as those encountered in pipeline inspections, blood vessel analysis, and rotating object studies. Spiral scanning of cylindrical images is employed to eliminate large jump in brightness at image junctions, forming a model of a quasi-periodic process. These images are modeled using autoregressive models, which rely on correlation functions dependent on autoregression parameters. The double stochastic model employed in this study introduces random inhomogeneities within the image. Additionally, [24] proposes pseudogradient algorithms for modal identification (estimation) of hidden control images, which are essential for managing the parameters and properties of the observed resulting image.

2. The idea of agglomerative pixel clustering

The idea of agglomerative multi-threshold pixel clustering, as presented in [7], involves the sequential combination of pairs of adjacent clusters that are pre-ordered.

For 8-bit and 24-bit grayscale images, a brightness histogram is used (at the software level, this involves working with an array of brightness intensities). Each column of the histogram represents a different cluster of pixels. There are N_{max} clusters in total, corresponding to the number of shades of gray present in the image. During the histogram pass cycle, all adjacent pairs of clusters are examined to identify the pair with the minimum distance $Dist(i, i+1)$, $i = [1; 255]$. At the end of each pass, the identified "minimum pair" is combined. The pass cycle repeats N_{max} times until a single cluster is formed.

Fig. 1 schematically illustrates the identification and combination of a "minimum pair". The initial histogram is shown in Fig. 1*a*. The abscissa axis O*x* represents brightness values *z*, and the ordinate axis O*y* represents the number of pixels *h*(*z*) of each brightness. All *N* brightness values present are numbered and form *N* separate clusters. During a pass through the histogram, all distances between pairs of adjacent clusters are calculated: *Dist*(1,2), *Dist*(2,3), ..., *Dist*(*N*−1,*N*).

Let's assume, as seen in Fig. 1*a*, that the "minimal pair" are clusters (3,4). Then they are combined into one enlarged cluster 3. All subsequent clusters are renumbered, as noted in Fig. 1*b* red. The cycle of agglomerative clustering is repeated.

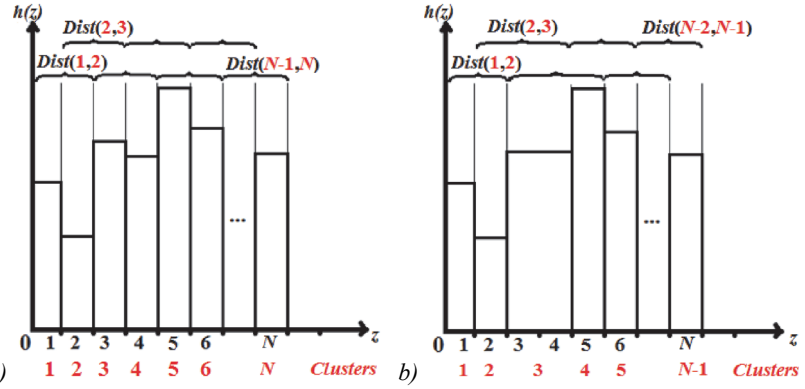

Fig. 1. An example of combining of adjacent clusters represented by a brightness histogram: a) original histogram, b) histogram after combine step

3. Main steps of agglomerative clustering program

The main steps of the agglomerative clustering program considered in this research are as follows:

0. Convert the image to grayscale 8-bit or 24-bit[1];
1. Form an array *Z* of gray levels *z* (brightness intensities)[2];
2. Calculate the initial number N_{max} of non-empty shades of gray present in array *Z* (of the converted image)[3]. Initially, a separate non-empty cell of array *Z* is a separate cluster and contains a separate gray level *z*;
3. Repeat *N* times in a cycle of passing through brightness array *Z*:
    a. Calculate distances *Dist* for each adjacent pair of clusters;
    b. Combine the "minimal pair" of clusters found during a pass through array *Z* by filling average intensities in all cells of enlarged cluster;
    c. Renumber the clusters[4];
    d. Generate and upload to the hard drive an approximation of the image obtained by combining the previously found "minimal pair".

[1]For color images a completely different program has to be built.

[2]An array of brightness intensities forms a brightness histogram for the image or its approximations obtained in the loop.

[3]The range of valid values for *N* is [1; 256]. In general N_{max} is 256.

[4]Cells of the same cluster contain equal values.

The following question arises: how to calculate the distance *Dist* between pairs of adjacent clusters?

4. Methods for calculating the distance between pairs of adjacent clusters

This section discusses three methods for calculating the distance *Dist* between pairs of adjacent clusters: the original and two modifications. In the original method, the criterion for choosing a "minimum pair" is the minimum of the product of interclass and intraclass variances. In the first modification, it is the minimum increment of the total squared error. In the second modification, the decision criterion is the minimum of partial entropy.

Method 1. The original method [7] of calculating the distance between pairs of adjacent clusters involves considering the brightness histogram as a function of probability density. In this method, the distance *Dist* between pairs of adjacent clusters is calculated through the product of variances (formula (1)):

$$Dist(1,2) = \sigma^2{}_A(1\cup 2)\times\sigma^2{}_I(1\cup 2), \quad (1)$$

where 1, 2 – a pair of adjacent clusters, $\sigma^2{}_A(1\cup 2)$ – intraclass variance, and $\sigma^2{}_I(1\cup 2)$ – interclass variance.

Intraclass variance $\sigma^2{}_A(1\cup 2)$ is calculated as follows (formula (2)):

$$\sigma_A^2(1\cup 2) = \frac{P(1)P(2)}{\left(P(1)+P(2)\right)^2}\times[m(1)-m(2)]^2, \quad (2)$$

where $P(1)$, $P(2)$ – the probabilities of clusters 1 and 2, $m(1)$ and $m(2)$ – the average brightness values of clusters 1 and 2.

Auxiliary quantities $P(i)$ and $m(i)$ are calculated as follows. The cluster probability $P(i)$ is determined through the sum of the probabilities $p(z)$ of the brightness components z included in it (formula (3)) in a given range from $T_{k-1}+1$ to T_k:

$$P(i) = \sum_{z=T_{k-1}+1}^{T_k} p(z). \quad (3)$$

Setting the range from $T_{k-1}+1$ to T_k indicates that the clusters formed on the histogram are continuous. For control purposes, remember that the sum of the probabilities (formula (4)) of all clusters is equal to 1:

$$\sum_{i=1}^{N} P(i) = 1. \quad (4)$$

The average brightness value of the cluster $m(i)$ is calculated as follows (formula (5)):

$$m(i) = \frac{1}{P(i)}\sum_{z=T_{k-1}+1}^{T_k} zp(z). \quad (5)$$

The interclass variance $\sigma^2{}_I(1\cup 2)$ is calculated as follows (formula (6)):

$$\sigma_I^2(1\cup 2) = \frac{1}{P(1)+P(2)}\times\sum_{z=T_{k1-1}+1}^{T_k}\left[\left(z-M(1\cup 2)\right)^2 p(z)\right]. \quad (6)$$

To determine the value of the interclass variance $\sigma^2{}_I(1\cup 2)$, in addition to the cluster probabilities $P(1)$, $P(2)$ and the average brightness values for the clusters $m(1)$, $m(2)$, the value of the so-called "global average" $M(1\cup 2)$ will be required. It means the average brightness of such a cluster that could be formed as a result of combining the original pair of clusters 1 and 2, and is calculated as follows (formula (7)):

$$M(1\cup 2) = \frac{P(1)m(1)+P(2)m(2)}{P(1)+P(2)}. \quad (7)$$

The disadvantages of the original method of calculating the distance *Dist* between pairs of adjacent clusters are obvious. Firstly, it is easy to make mistakes in complex calculation formulas. Secondly, the original method does not involve assessing the quality of the current division into clusters due to the lack of appropriate functionality. For these reasons, two modifications have been proposed.

Method 2. The first modification is to use the increment of the total squared error ΔE as a function of the distance *Dist* between pairs of adjacent clusters (formula (8)):

$$Dist(1,2) = \Delta E(1\cup 2) = \frac{n_1 n_2}{n_1+n_2}|I_1 - I_2|. \quad (8)$$

where n_1 и n_2 – the number of pixels in clusters 1 and 2, I_1 and I_2 – the average brightness intensities of clusters 1 and 2, | | – the modulus operation.

Calculation of the quality of the current partition is obtained by simply adding the values of the increment $\Delta E_{step\ i\text{-}1}$ and the total squared error $E_{step\ i\text{-}1}$ of the previous partition (formula (9)):

$$E_{stepi} = E_{stepi-1} + \Delta E_{stepi-1}, E_{step0} = 0. \quad (9)$$

The root mean squared error σ is related to the total squared error E by a simple formula (10):

$$\sigma = \left(\frac{E}{S}\right)^{\frac{1}{2}}, \quad (10)$$

where S – the total number of pixels in the image.

The advantages of the first modification are obvious. Firstly, a simple formula is used to calculate the distance *Dist* between pairs of adjacent clusters. Secondly, the accumulated value of the increment of the total squared error ΔE serves as a functional for the quality of partitions. Thirdly, this modification halves the number of operations per combining step compared to the original calculation method [7, 16]. Fourthly, it is this modification that makes it possible to generate a sequence of suboptimal partitions (see Section 5, Figs. 2, 11).

Method 3. The second modification is to use the partial entropy H to calculate the distance *Dist* between pairs of adjacent clusters (formulas (11), (12)):

$$Dist(1,2) = H(1 \cup 2) = P_{(1U2)}\, log\, P_{(1U2)}, \quad (11)$$

$$P_{(1U2)} = \sum_{i=1}^{k} p_i, \quad (12)$$

where $P_{(1}\bigcup_{2)}$ – the probability of an enlarged cluster consisting of clusters 1 and 2, p_i – the probability of brightness included in the enlarged cluster, k – the number of brightness levels in the enlarged cluster.

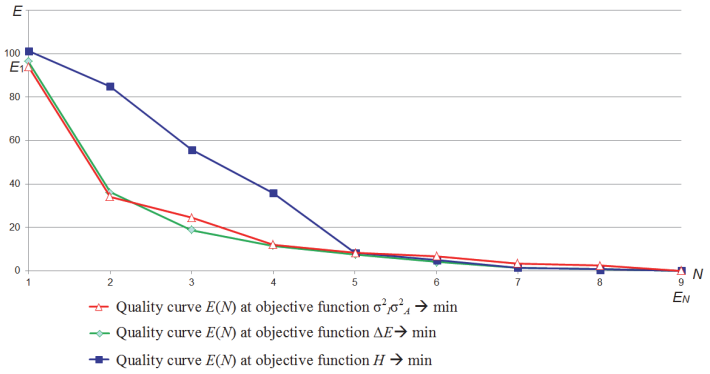

Fig. 2. Quality graphs for three agglomerative clustering algorithms using different objective functions for calculating the "minimum pair" of clusters

The calculation of the quality of the partition is carried out through the formula (13) for the amount of information Y, the calculated difference between the constant a priori H_{apri} entropy (formula (14)) and the variable a posteriori H_{apos} entropy (formula (15)):

$$Y = H_{apri} - H_{apos}, \quad (13)$$

$$H_{apri} = -\sum_{i=1}^{n\sum_i} p_i log, \quad (14)$$

$$H_{apos} = -\sum_{i=1}^{n'\sum_i} P_i log, \quad (15)$$

where n – the initial number of clusters, n' – the number of clusters after the combining step, $1 \leq n' < n$.

The advantages of the second modification are similar to the first. Firstly, a very simple formula is used for calculating the distance *Dist*. Secondly, the amount of information Y (formula (13)) serves as a functional for the quality of partitions, assessing the reduction in uncertainty. Thirdly, this modification provides an alternative approach to the evaluation of image quality.

5. Examples of combining sequences

This section discusses examples of agglomerative multithreshold data processing. In the first example, different methods of calculating the distance *Dist* between pairs of adjacent clusters are compared on a test data set, and the best method for forming clusters is determined. In the second example, the chosen method of calculating the distance *Dist* for forming pixel clusters is applied to a set of standard Earth remote sensing test images converted to grayscale. As a result, series of piecewise-constant partitions in shades of gray are generated.

Example 1. Let the distribution of 50 pixels be as shown in Tab. 1. The first row Z contains brightness levels z, and the second row contains the number of pixels $h(z)$ for each brightness level z.

Tab. 1. Pixel distribution

Z	100	101	102	103	104	105	106	107	108	109	110
h(z)	3	1	4	8	10	5	0	0	15	3	1

Tab. 2, 3 and 4 show sequences of combinations using different functions to calculate the distance *Dist* between pairs of adjacent clusters for the same set of pixels.

Tab. 2. The sequence of pixel cluster combinations in the original distance calculation method: $\sigma^2/\sigma^2_A \rightarrow min$

	(Intensities)									
Partition number	100	101	102	103	104	105	108	109	110	**Clusters** (Intensities)
1)	**1**(100)	**2**(101)	**3**(102)	**4**(103)	**5**(104)	**6**(105)	**7**(108)	**8**(109)	**9**(110)	
2)	**1**(100)	**2**(101)	**3**(102)	**4**(103)	**5**(104)	**6**(105)	**7**(108,17)		**8**(110)	
3)	**1**(100)	**2**(101,8)		**3**(103)	**4**(104)	**5**(105)	**6**(108,17)		**7**(110)	
4)	**1**(100)	**2**(101,8)		**3**(103)	**4**(104,33)		**5**(108,17)		**6**(110)	
5)	**1**(100)	**2**(101,8)		**3**(103)	**4**(104,33)		**5**(108,26)			
6)	**1**(100)	**2**(102,54)			**3**(104,33)		**4**(108,26)			
7)	**1**(100)	**2**(103,5)					**3**(108,26)			
8)	**1**(103,16)						**2**(108,26)			
9)	**1**(105,1)									

Tab. 3. The sequence of pixel clusters combinations in the first modification of the distance calculation method: $\Delta E \rightarrow min$

	(Intensities)									
Partition number	100	101	102	103	104	105	108	109	110	**Clusters** (Intensities)
1)	**1**(100)	**2**(101)	**3**(102)	**4**(103)	**5**(104)	**6**(105)	**7**(108)	**8**(109)	**9**(110)	
2)	**1**(101,25)		**2**(102)	**3**(103)	**4**(104)	**5**(105)	**6**(108)	**7**(109)	**8**(110)	
3)	**1**(101,25)		**2**(102)	**3**(103)	**4**(104)	**5**(105)	**6**(108)	**7**(109,25)		
4)	**1**(101,25)		**2**(102,67)		**3**(104)	**4**(105)	**5**(108)	**6**(109,25)		
5)	**1**(101,25)		**2**(102,67)		**3**(104,33)		**4**(108)	5(109,25)		
6)	**1**(101,25)		**2**(102,67)		**3**(104,33)		**4**(108,26)			
7)	**1**(102,06)				**2** (104,33)		**3**(108,26)			
8)	**1**(103,16)						**2**(108,26)			
9)	**1**(105,1)									

Tab. 4. The sequence of pixel clusters combinations in the second modification of the distance calculation method: $H \rightarrow min$

	(Intensities)									
Partition number	100	101	102	103	104	105	108	109	110	**Clusters** (Intensities)
1)	**1**(100)	**2**(101)	**3**(102)	**4**(103)	**5**(104)	**6**(105)	**7**(108)	**8**(109)	**9**(110)	
2)	**1**(100,25)		**2**(102)	**3**(103)	**4**(104)	**5**(105)	**6**(108)	7(109)	**8**(110)	
3)	**1**(100,25)		**2**(102)	**3**(103)	**4**(104)	**5**(105)	**6**(108)	**7**(109,25)		
4)	**1**(101,125)			**2**(103)	**3**(104)	**4**(105)	**5**(108)	**6**(109,25)		
5)	**1**(101,125)			**2**(103)	**3**(104,333)		**4**(108)	**5**(109,25)		
6)	**1**(101,125)			**2**(103)	**3**(106,167)			**4**(109,25)		
7)	**1**(101,125)			**2**(105,5)				**3**(109,25)		
8)	**1**(104,739)							**2**(109,25)		
9)	**1**(105,1)									

The first column lists the partition order number, beginning with 1, which corresponds to the initial partition. Columns two through ten (intensities 100–110) in Tab. 2, 3, and 4 display the intensity values from Tab. 1 that have non-zero frequencies. These values form separate clusters. In contrast, intensities 106 and 107 are grouped with the initial cluster at intensity 105, as they do not influence the calculations. The ordinal numbers of the clusters are highlighted in bold. After each merger, clusters are renumbered, with the average brightness value of each cluster indicated in parentheses. The color coding provides a visual guide to follow the sequence of cluster mergers more easily.

Tab. 2 shows the combination sequence using the original method of calculation, which uses the minimum product of interclass and intraclass variances as the decision criterion: $\sigma^2_I\sigma^2_A \rightarrow$ min.

Tab. 3 shows the sequence of cluster combinations using the first modified version of the distance calculation, which uses the minimum increment of the total squared error: $\Delta E \rightarrow$ min.

Tab. 4 shows the sequence of cluster combinations using the second modified version of the distance calculation, which uses the minimum private entropy for the decision on merging: $H \rightarrow$ min.

In all cases, a different order of combining clusters is observed. The following questions arise: Which sequence of combinations is better? Is there an optimal sequence? How close are the calculated sequences to the optimal one?

To answer these questions, algorithms implementing various functions for calculating the distance *Dist* are evaluated using a unified assessment of partition quality. Either the amount of information Y or the total squared error E can be used for this purpose. The choice of the latter is due to several reasons:

a) From a programming perspective, the amount of information Y is more difficult to use. More testing conditions are needed since Y requires calculating the sum of the posterior entropy H_{apos} for each cluster in each partition. The total squared error E, on the other hand, is simpler to calculate, requiring only the accumulation of the values of the increments ΔE of the combined pairs of clusters from each partition.

b) From a mathematical analysis perspective, it is more convenient to use the graph of the total squared error E. It is similar to a falling exponential and more convenient for building a mathematical model. The graph of E starts at the ordinate axis Oy, decreases, and ends at the abscissa axis Ox (see Fig. 2 and 11), while the graph of the entropy function H starts at the side of the plane and ends at the origin.

Fig. 2 shows how the quality graphs of three agglomerative clustering algorithms with different objective functions for decision making behave. The abscissa axis Ox represents the number of clusters N, and the ordinate axis Oy represents the total squared error E. The graphs are considered from right to left, from a larger number of clusters N to a smaller number. At the beginning, the graphs are bundled together, but they diverge towards the end. This behavior is explained by the initial merging of small clusters. After a certain point, only pairs of large clusters remain. The worst performance is observed in the quality graph for the algorithm with the objective entropy function H. This is due to the calculation of H, which considers only the areas of the merging clusters without considering their brightness (see Tab. 5). Some compensation is achieved by the histogram ordering condition, but it is insufficient.

Tab. 5. Dependencies of objective functions on parameters

Objective function	**Parameters**
$\Delta E \rightarrow$ min	Intensity, cluster area
$H \rightarrow$ min	Cluster area
$\sigma^2_I\sigma^2_A \rightarrow$ min	Intensity, cluster area

The quality graphs for algorithms using the increment of the total squared error ΔE and the product of variances $\sigma^2_I\sigma^2_A$ are close (see Fig. 2), but they have a significant difference. The quality graph for the product of variances $\sigma^2_I\sigma^2_A$ has at least one inflection point, whereas the quality graph for the algorithm using the increment of the total squared error ΔE is convex and monotonically decreasing. This suggests the hypothesis that the first modification can generate an optimal or nearly optimal sequence of partitions, which must be tested on real images.

Example 2. To confirm the hypothesis about the optimality or proximity to the optimum of the partitions, we apply agglomerative multithreshold processing in the first modified version, using the increment of the total squared error ΔE as the distance function *Dist* to standard test images from the open international database of the Signal and Image Processing Institute of the University of Southern California (SIPI USC) [25]. The characteristics of the images used are shown in Table 6. Since the multithreshold processing algorithm operates with shades of gray, the original images are converted to grayscale (see Fig. 3 – 6).

Tab. 6. Characteristics of the original standard test images

File Name	Description	Dimensions	Image Type
2.2.05	San Diego (Miramar NAS)	1024×1024 pixels, 3072 kilobit	color (24 bit/pixel)
3.2.25	Pentagon	1024×1024 pixels, 1024 kilobit	grayscale (8 bit/pixel)
wash-ir	Washington, D.C. (infra-red)	2250×2250 pixels, 14831 kilobit	color (24 bit/pixel)
2.2.06	San Francisco (Bay Bridge)	1024×1024 pixels, 3072 kilobit	color (24 bit/pixel)

Fig. 3.Grayscale image “2.2.05”

Fig. 4. Grayscale image “3.2.25”, 24 bit/pixel

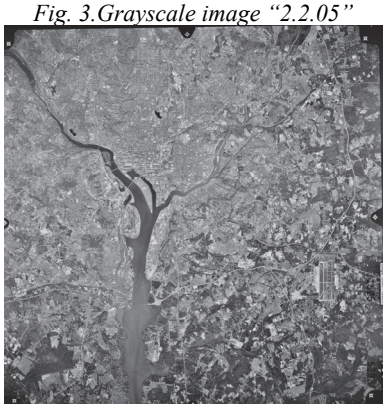

Fig. 5. Grayscale image “wash-ir”

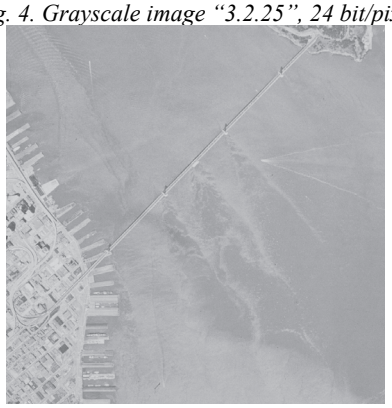

Fig. 6. Grayscale image ”2.2.06”

Let's consider the Fig. 3 grayscale image “2.2.05”, which contains 210 shades of gray, forming 210 initial clusters. This means there will be a total of 209 cycles through the brightness histogram since the multithresholding algorithm operates based on the initial number of clusters. The number of cycles is less than the number of partitions by 1.

The results of agglomerative multithreshold processing of images are shown in Fig. 7–10. Each figure shows the final parts of the sequences of partitions into clusters. The 20th, 6th, 5th, 4th, 3rd, 2nd, and 1st partitions are shown. Each subsequent partition is obtained from the previous one by combining a pair of adjacent clusters in the histogram. Each piecewise-constant partition is labeled with its ordinal number equal to the number of clusters N in the partition and the value of the partition quality functional – the root mean square deviation σ. Each figure indicates the total number of partitions in the sequence.

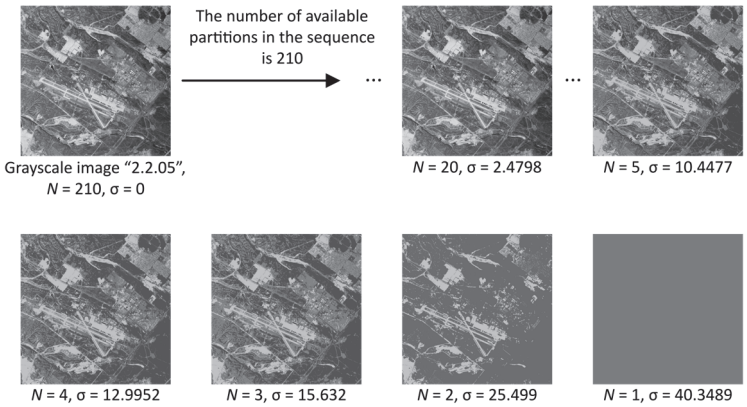

Fig. 7. The final part of a series of piecewise- constant approximations for the grayscale image "2.2.05" (San Diego, Miramar NAS)

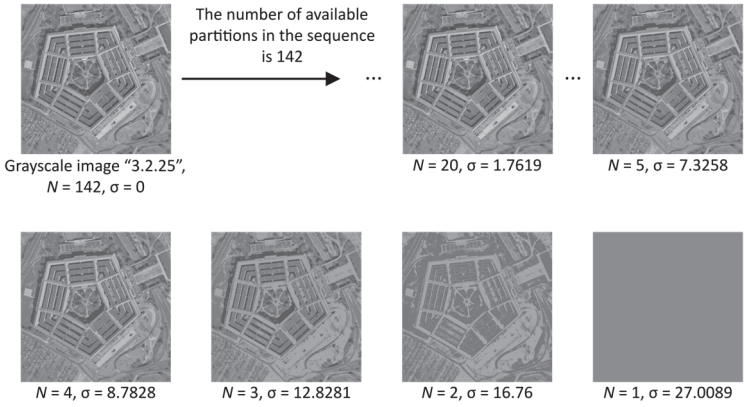

Fig. 8. The final part of a series of piecewise-constant approximations for the grayscale image "3.2.25" (Pentagon)

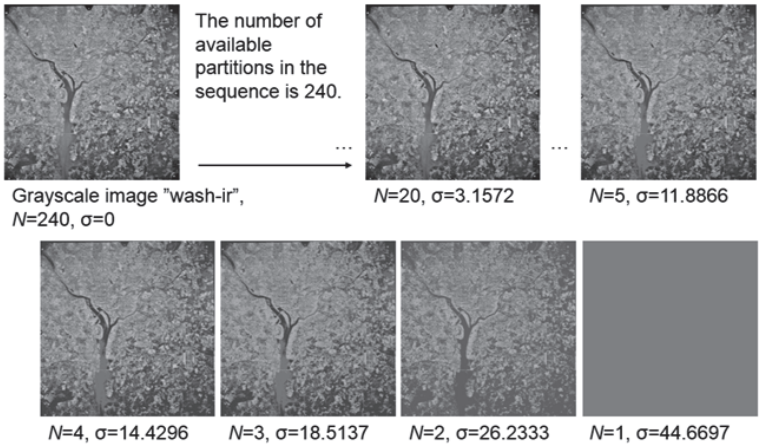

Fig. 9. The final part of a series of piecewise constant approximations for a grayscale “wash-ir” image (Washington, D.C., infra-red)

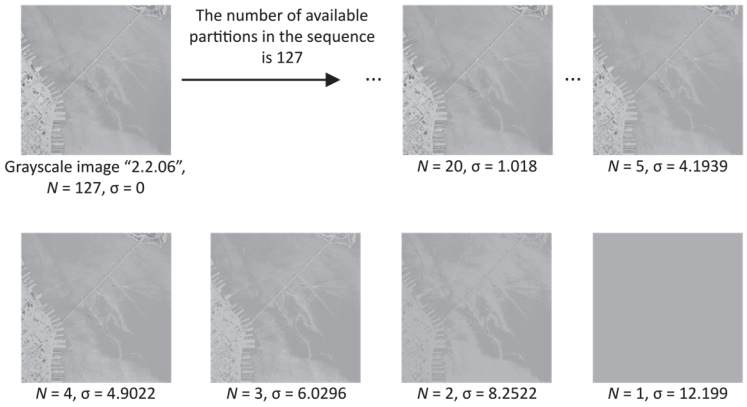

Fig. 10. The final part of a series of piecewise-constant approximations for the grayscale image “2.2.06” (San Francisco, Bay Bridge)

Fig. 11 shows four quality graphs for four sequences of partitions, formed from the σ values characterizing the piecewise-constant partitions. The green curve, starting at the point $\sigma_{(N=240)}=0$ and ending at the point $\sigma_{(N=1)}=44.6697$, belongs to the sequence of piecewise-constant partitions of the grayscale image "wash-ir" (Washington, D.C., infra-red) (see Fig. 9).

The dark blue curve in Fig. 11, starting at $\sigma_{(N=210)}=0$ and ending at $\sigma_{(N=1)}=40.3989$, belongs to the sequence of piecewise-constant partitions of the grayscale image "2.2.05" (San Diego, Miramar NAS) (see Fig. 7).

The red curve in Fig. 11, starting at $\sigma_{(N=142)}=0$ and ending at $\sigma_{(N=1)}=27.0089$, belongs to the sequence of piecewise-constant partitions of the grayscale image "3.2.25" (Pentagon) (see Fig. 8).

The light blue curve in Fig. 11, starting at $\sigma_{(N=127)}=0$ and ending at $\sigma_{(N=1)}=12.199$, belongs to the sequence of piecewise-constant partitions of the grayscale image "2.2.06" (San Francisco, Bay Bridge) (see Fig. 10).

The examination showed that the quality graphs for the total squared errors E (curves in Fig. 11) are convex and monotonically decreasing throughout:

$$E = S\sigma^2, \frac{(E_{i-1}+E_{i+1})}{2} < E_i, i \in [2;255], \tag{16}$$

where S – total number of pixels in image.

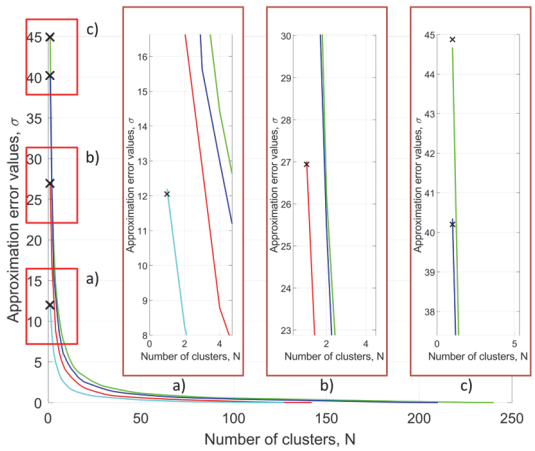

Fig. 11. Graphs of standard deviations for four sequences of piecewise-constant partitions that characterize quality of series of partitions into clusters

The hypothesis proposed at the end of the first example is confirmed: the modification using the increment of the total squared error ΔE as the distance function *Dist* between pairs of adjacent clusters allows generating suboptimal piecewise-constant partitions. The degree of proximity to the optimum is a subject for separate study.

Fig. 7 – 10 present two parameters under each partition. The first parameter, N, represents the number of clusters in the partition and serves as the serial number in the sequence. Its values range from N_{max} to 1, updating iteratively during software calculations. The constant N_{max} is defined by the number of gray shades in the original image, taking a value within the range [1; 256]. The second parameter, σ, indicates the standard deviation, representing the approximation error and characterizing the quality of each partition. Lower values of σ or E for a given N signify higher clustering quality.

By using the increment of the total squared error, ΔE as a quality functional, the total squared error E for a partition can be computed using an additive formula (9), where values are accumulated. This approach streamlines programming by eliminating the need to generate a separate table for quality calculation. If alternative functionals from Tab. 5 are employed to identify the "minimum pair," additional program arrays must be generated and combined into a separate table to assess the quality for each partition using the difference formulas (17) and (18):

$$E = \sum_{g=1}^{N_{max}\Sigma} E_g, \tag{17}$$

$$E_g = \sum_{i=1}^{j} z_i^2 h(z_i) - \frac{\left(\sum_{i=1}^{j} z_i h(z_i)\right)^2}{w}, \tag{18}$$

where E – the total squared error for the entire partition, E_g – the total squared error for individual cluster g in the N^{th} partition (N^{th} partition contain N individual clusters), j – the number of intensities in cluster g, z – the intensity levels within cluster g, $h(z)$ – the frequency of intensity z in cluster g, w – the number of pixels in cluster g.

The number of pixels in cluster g is calculated by formula (19):

$$w = \sum_{i=1}^{j} h(z_i), \tag{19}$$

where initial data $h(z)$ obtained from original grayscale image.

Formula (18) requires two terms to be computed for each cluster at each iteration of the partitioning cycle, followed by calculating their difference.

6. Result analysis and discussion

Tab. 7 presents the control and achieved values obtained through the first modification of Arifin's method, which are used to assess the convergence of the results of this modification. The first column lists the names of the standard test images. The second column contains the control values, calculated for each test image by merging all pixels into a single cluster and subsequently determining the approximation error value. The third column provides the approximation error values for partitions into a single cluster. These values result from an agglomerative sequence of piecewise-constant partitions, where each subsequent partition is generated from the previous one by merging a pair of adjacent clusters chosen based on the optimal value of the functional. The fourth column shows the difference between the control and achieved values, along with the relative error in approximating the control value of σ.

Tab. 7. Control and achieved σ values for first partitions N = 1

Image File Name	Control values	Achieved values	±Δ, (±%)
2.2.05	40.2003	40.3489	+0.1486, (+0.370%)
3.2.25	26.9396	27.0089	+0.0693, (+0.257%)
wash-ir	44.8755	44.6697	–0.2058, (–0.495%)
2.2.06	12.0482	12.1990	+0.1508, (+1.252%)

Using the minimum increment of the total squared error as the distance function *Dist* in Arifin's agglomerative method for cluster enlargement simplifies development and reduces programming complexity for software implementation. However, practical data obtained from processing the standard test images 2.2.05, 3.2.25, and 2.2.06 reveal that this modification accumulates the approximation error σ, as demonstrated by the values in the fourth column of Tab. 7. The curves in Fig. 11 are monotonically convex; however, the control values for $N = 1$ do not converge, confirming the hypothesis from Example 1 in Section 5 regarding the generation of suboptimal solutions by this modification.

Special attention should be given to the "wash-ir" image, which contains 4075 "broken" (or "empty") pixels (0.097 % of the total pixel count) with values outside the range [1; 256]. To prevent programming errors, these pixel values were replaced with the nearest boundary value, 1, in the control method. This adjustment impacts both control and achieved results.

Formula (8) for calculating the increment of the total squared error in the first modification resembles the formula used to compute intraclass variance (2) in the original method. In the first factors of formulas (8) and (2), the pixel count n_i corresponds to the cluster probability $P(i)$. The similarity between the second factors in formulas (8) and (2) lies in calculating the difference in average brightness. However, while formula (8) uses the absolute value of the brightness difference, formula (2) squares this difference. This structured approach allows for the use of a simple difference in programming, as the second brightness value is expected to be greater than the first; otherwise, an error may occur.

The original method includes a correction coefficient, the interclass dispersion, which accounts for "long-distance relations" when searching for a "minimal pair." This coefficient is calculated in a more complex manner than the main function – intraclass variance (formula (2)) – yet it enables a precise solution. The approximation error graphs reach the control values but exhibit bends along their length (see Fig. 2). Excluding the correction coefficient from the calculations reduces programming efforts by 25 – 30 %, while only slightly compromising the final result by approximately 1.5 %. The Matlab program for implementing the original Arifin method contains about 1200 lines of code, whereas the first modification is under 900 lines. Identifying a replacement for the correction coefficient (formula (6)) could help achieve control values with reduced programming effort.

7. Implementation details

Matlab code implementing the modified Arifin's method for agglomerative multilevel thresholding of grayscale image pixels is available on GitHub [26]. The code comprises fewer than 900 lines. Upon receiving an image as input, the program checks whether it is 8-bit or 24-bit and converts it to grayscale. A folder is then created, named after the input image, to store the output data. The program operates without requiring any control parameters. Its main loop is linear, generating one image partition per iteration and calculating its approximation error. Typically, over 256 cycles, all 256 image partitions are generated. Upon completion, the program displays a graph of approximation error values in both linear and logarithmic scales. Each output file name contains the partition order number and the corresponding approximation error value.

8. Conclusion

In this paper:

Firstly, the idea of agglomerative multithreshold processing of grayscale images was presented, which involves the sequential merging of "minimal pairs" of adjacent clusters calculated at each step of the process (see Section 2)).

Secondly, the main stages of the agglomerative pixel clustering program were outlined, consisting of constructing a cycle through the brightness array for the sequential merging of "minimal pairs" (see Section 3).

Thirdly, an original method for calculating the distance *Dist* between pairs of adjacent clusters was presented, which involves calculating the product of intra-class and interclass variances (see Method 1 in Section 4).

Fourthly, the disadvantages of the original method of calculating the distance *Dist* were listed:

- it is characterized by cumbersome calculation formulas, which are easy to make mistakes in;
- it lacks built-in functionality for assessing the quality of the current clustering.

Fifthly, the significance of the simplifying modification based on the use of the increment of the total squared error ΔE (see Method 2 in Section 4) was substantiated:

- modification reduces the number of operations per cycle of merging clusters by half while maintaining linear computational complexity [16];
- the accumulated value of the increment of the total squared error ΔE (formula (9)) serves as an indicator of the quality of clustering (see formula (9));
- series of piecewise-constant partitions generated by the first modification (Method 2 in Section 4) form a convex monotonic sequence according to their values of total squared errors (see Figs. 2, 11).

Sixthly, an experimental study was conducted (see Example 2 of Section 5) of the modification using the increment of the total squared error ΔE as the objective function of decision-making, using examples of processing images:

1)«2.2.05» (San Diego, Miramar NAS),
2)«3.2.25» (Pentagon),
3)«wash-ir» (Washington, D.C., infra-red),
4)«2.2.06» (San Francisco, Bay Bridge)

taken from an open international database of standard test images [7]. The characteristics of the original aerospace images are given in Tab. 6. The images used were converted to grayscale format (see Fig. 3–6) according to the requirements of the agglomerative multithreshold processing algorithm (see Section 3).

Seventhly, the study further confirms, using the agglomerative processing example of a 50-pixel set (Example 1 in Section 5), that relying solely on adjacent pairs in searching for the "minimal pair" contributes to approximation error accumulation in Arifin's agglomerative clustering method. For example, Fig. 2 shows that the curves of the quality graphs tend to the vanishing point E_1 but do not reach it, lagging behind each other with a certain error. Therefore, the quality curves of partitioning sequences for converted test images obtained in Fig. 11, despite the monotonic convexity throughout, are suboptimal due to the fact that Arifin's agglomerative method limits comparisons to adjacent cluster pairs, thereby amplifying approximation error. Their actual value at the end of the path, especially in the range of values of the number of clusters N equal to [1; 50], should be slightly smaller than shown in the graph in Fig. 11. How much less and how close the curves are to the optimal values is the subject of the next study.

Eighthly, an analysis of the obtained partition results was conducted, confirming the hypothesis that the first modification of Arifin's method generates a suboptimal partition sequence. This conclusion is supported by the monotonic curve patterns in Fig. 11 and the data in Tab. 7, which reveal discrepancies from the control values at $N = 1$ (see Section 6).

Ninthly, a computer code implementation of the modified Arifin's method for agglomerative multilevel threshold clustering of grayscale image pixels has been developed and is available online via GitHub [25] (see Section 7).

Conclusions:

1) examples of processing standard test images showed that agglomerative pixel clustering is applicable for "blind" image segmentation;
2) application of the minimum increment of the total squared error ΔE as the objective function of decision-making simplifies the procedure for calculating the distance *Dist* between pairs of adjacent clusters by a factor of two;
3) removing the correction factor leads to a little loss of accuracy of the results;
4) it is necessary to find a way to simply the calculation of the correction factor to achieve control values with minimal programming effort.

Acknowledgements

The paper was prepared with the financial support of the Ministry of Science and Higher Education of the Russian Federation, grant agreement No. FSRF-2023-0003, "Fundamental principles of building of noise-immune systems for space and satellite communications, relative navigation, technical vision and aerospace monitoring".

References

[1] Jain J, Li J, Chiu MT, Hassani A, Orlov N, Shi H. OneFormer: One Transformer to Rule Universal Image Segmentation. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2023; 2989-2998.
[2] Hoeser T, Bachofer F, Kuenze C. Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review—Part II: Applications. Remote Sens 2020; 12(18): 3053. DOI: 10.3390/rs12183053.
[3] Otsu N. A Threshold Selection Method from Gray-level Histograms. IEEE Transactions on Systems, Man, and Cybernetics. 1979; 9(1): 62-66. DOI: 10.1109/TSMC.1979.4310076
[4] Kapur JN, Sachoo PK, Wong AKC. A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram. Computer Vision, Graphics and Image Processing. 1985; 29(1): 273-285. DOI: 10.1016/0734-189X(85)90125-2
[5] Kittler J, Illingworth J. Minimum Error Thresholding. Pattern Recognition. 1986; 19: 41-47. DOI: 10.1016/0031-3203(86)90030-0
[6] Ward JHJ. Hierarchical Grouping to Optimize an Objective Function. Journal of the American Statistical Association. 1963; 58: 236-244. DOI: 10.1080/01621459.1963.10500845

[7] Arifin AZ, Asosno A. Image segmentation by histogram thresholding using hierarchical cluster analysis. Pattern Recognition Letters. 2006; 27(13): 1515-1521. DOI: 10.1016/j.patrec.2006.02.022.
[8] Tong H, Tong F, Zhou W, Zhang Y. Purifying SLIC Superpixels to Optimize Superpixel-Based Classification of High Spatial Resolution Remote Sensing Image. Remote Sens. 2019; 11: 2627. DOI: 10.3390/rs11222627.
[9] Berkeley Segmentation Dataset 500 (BSDS500). Source: <https://paperswithcode.com/dataset/bsds500> (accessed on 2024.04.18)
[10] Rahkar Farshi T, Demirci R, Feizi-Derakhshi MR. Image Clustering with Optimization Algorithms and Color Space. Entropy 2018; 20: 296. DOI: 10.3390/e20040296.
[11] Cardone B, Di Martino F, Miraglia V. A Novel Fuzzy-Based Remote Sensing Image Segmentation Method. Sensors 2023; 23: 9641. DOI: 10.3390/s23249641.
[12] Huang S, Zhang O, Chen Q. Ship Target Detection Method in Synthetic Aperture Radar Images Based on Block Thumbnail Particle Swarm Optimization Clustering. Remote Sens. 2023; 15: 4972. DOI: 10.3390/rs15204972.
[13] Wang Z. Unsupervised Wavelet-Feature Correlation Ratio Markov Clustering Algorithm for Remotely Sensed Images. Appl. Sci. 2024; 14: 767. DOI: 10.3390/app14020767.
[14] Olkina DS. Algorithm of Semantic Image Segmentation for Solving the Problem of Positioning an Aircraft on the Earth`s Surface. Trudy MAI. 2023. № 130. DOI: 10.34759/trd-2023-130-18 (In Russian)
[15] Bukirev AS. Method for Diagnosing an Aircraft on-board Equipment Complex Based on Machine Learning. Trudy MAI. 2023; 133. Source: <https://trudymai.ru/published.php?ID=177672> (In Russian)
[16] Khanykov IG. Technology modification for distance calculation between pairs of adjacent clusters in multilevel thresholding. Trudy MAI. 2021; 118: 1-34. DOI: 10.34759/trd-2021-118-13.
[17] Koshkarov AS, Gulii DD, Bariaksheva VP. Multi-Rotor Unmanned Aerial Vehicle Emergency Landing Algorithm Based on Underlying Surface Image Analysis. Trudy MAI. 2023; 132. Source: <https://trudymai.ru/published.php?ID=176854> (In Russian)
[18] Lukin VN, Nikitin IK. Algoritm poiska sobytii v videoposledovatel'nostiakh. Trudy MAI. 2017; 96. (In Russian)
[19] Molotkov AA, Tretiakova ON. Application of Machine Vision in Laser Technologies. Trudy MAI. 2022; 127. DOI: 10.34759/trd-2022-127-25. (In Russian)
[20] Nenashev VA, Khanykov IG. Formation of the complex image of the Earth surface based on the pixel clustering of the location images in the multi-position airborne system. Informatics and Automation 2021; 20(2): 302-340. DOI: 10.15622/ia.2021.20.2.3.
[21] Nenashev VA, Shepeta AP, Kryachko AF. Fusion radar and optical information in multi-position on-board location systems. In Proceedings of the 2020 Wave Electronics and Its Application in Information and Telecommunication Systems (WECONF), St. Petersburg, Russia, 1–5 June 2020; pp. 1-5. – DOI: 10.1109/WECONF48837.2020.9131451.
[22] Schwab M, Mayr1 A, Haltmeie M. Deep Gaussian mixture model for unsupervised image segmentation. 2024; arXiv:2404.12252v1. DOI: 10.48550/arXiv.2404.12252
[23] Andriyanov NA, Dementiev VE. Developing and Studying the Algorithm for Segmentation of Simple Images Using Detectors Based on Doubly Stochastic Random Fields. Pattern Recognition and Image Analysis 2019; 29(1): 1–9. DOI:10.1134/s105466181901005x.
[24] Krasheninnikov V, Kuvayskova Yu, Subbotin A. Pseudo-gradient algorithm for identification of doubly stochastic cylindrical image model. 24th International Conference on Knowledge Based and Intelligent Information & Engineering Systems 2020; 1858–1867. DOI: 10.1016/j.procs.2020.09.225.
[25] Signal and Image Processing Institute open dataset, Volume 2: Aerials. Source: <https://sipi.usc.edu/database/database.php?volume=aerials&image=38#top >
[26] Modified Arifin's method. Source: <https://github.com/IgorKhanykov/ModifOfAACM>

Authors' information

Igor Georgievich Khanykov, (b. 1987) holds a Bachelor's degree (2010) and Master's degree (2012) from Saint Petersburg State University of Aerospace Instrumentation (SUAI). In 2018, he completed the postgraduate program of the St. Petersburg Institute of Informatics and Automation of the Russian Academy of Sciences with graduate qualification work focusing on System Analysis and Automation (05.13.01). He currently works as a junior researcher and programmer at the SUAI Engineering School. Research interests: digital image processing, image segmentation, clustering methods, and big data in cluster analysis. E-mail: *igk@iias.spb.su*

Vadim Aleksandrovich Nenashev, (b. 1989) studied at Saint Petersburg State University of Aerospace Instrumentation (SUAI). He currently works as an associate professor in the Department of Design and Technology of Electronic and Laser Means in SUAI, heads the Machine Learning Research Laboratory at SUAI Engineering School, and is a leading researcher in SUAI. Research interests: computer modeling, navigation and radar systems, statistical analysis, and information integration. E-mail: *nenashev.va@yandex.ru*

Mikhail Vyacheslavovich Kharinov, (b. 1954) studied at Leningrad State University named after Zhdanov (Currently – Saint Petersburg State University). He currently works at the St. Petersburg Federal Research Center of the Russian Academy of Sciences (SPC RAS) as a senior researcher in the Laboratory of Big Data of Sociocyberphysical Systems. Research interests: dynamic trees, steganography, image segmentation, and object recognition. E-mail: *khar@iias.spb.su*

Received July 28, 2024. The final version – March 24, 2025.