Fast Color Images Clustering for Real-Time Computer Vision and AI System

Victoria Vysotska[1], Kirill Smelyakov[2], Nataliia Sharonova[3], Eugen Vakulik[2], Oleksii Filipov[2], Ruslan Kotelnykov[2]

[1] Lviv Polytechnic National University, Stepan Bandera Street, 12, Lviv, 79013, Ukraine
[2] Kharkiv National University of Radio Electronics, Nauky Ave. 14, Kharkiv, 61166, Ukraine
[3] National Technical University "KhPI", Kyrpychova str. 2, Kharkiv, 61002, Ukraine

Abstract

The article describes the development of a color image clustering algorithm for a real-time computer vision and AI system. An important feature of the algorithm is the preliminary use of a large number of fast image preprocessing algorithms to optimize the preparation of data for efficient clustering in the color space. The proposed clustering algorithm is focused on processing clusters of arbitrary shape. The paper presents the results of experiments on the processing of monochromatic and non-monochromatic color images. These clustering results are shown as images and as clusters in the Unity environment. Recommendations for practical use, conclusions and links to repositories of experiment results are given. The analysis of the results of a large number of experiments shows that the proposed adaptive clustering algorithm can be effectively used both in powerful computer vision and artificial intelligence systems and in relatively low-power embedded systems by adjusting the parameters to the specifics of the problem

Keywords

Fast Clustering, Color Image, Preprocessing, Grid Model, Adaptation, Efficiency Estimation

1. Introduction

The development of innovative tools [1-3] and modern technologies for pattern recognition [4, 5] in our time is increasingly focused on the use of new models and algorithms of artificial intelligence (AI) [6], primarily clustering, artificial neural networks (NN) and classification. Although, pre-processing models [1, 6], solutions in the fields of ICT [7], in robotics with use of embedded systems [8-10] and security systems [11-13] are also of great interest in some aspects.

Such a great interest in artificial intelligence systems is associated with their effective application in pattern recognition systems, which are enjoying commercial success. Currently, such (data-centric business) systems and applications are successfully used to search for goods in online stores using Image Based Search technology. In car and biometric identification systems, contactless payment systems in supermarkets based on face recognition technology, in many other applications.

Concerning clustering, a large number of new models and algorithms have been proposed in recent years. At the same time, not all of them meet modern requirements in terms of computational efficiency, especially when used in embedded systems. What is the reason for this?

Most of the algorithms iteratively recalculate the position of the centers over all observations of the cluster. The problem is that the number of observations may be too large. Taking into

COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine

victoria.a.vysotska@lpnu.ua (V. Vysotska); kyrylo.smelyakov@nure.ua (K. Smelyakov); yevhen.vakulik@nure.ua (E. Vakulik); nvsharonova@ukr.net (N. Sharonova); ruslan.kotelnykov@nure.ua (R. Kotelnykov); oleksii.filipov@nure.ua (O. Filipov)

0000-0001-6417-3689 (V. Vysotska); 0000-0001-9938-5489 (K. Smelyakov); 0000-0002-4940-0529 (E. Vakulik); 0000-0002-8161-552X (N. Sharonova); 0000-0003-2413-1809 (R. Kotelnykov); 0000-0001-6181-3871 (O. Filipov)

account the size of modern images (from 8 to 64 MP), the number of observations in dense clusters and their surroundings can be estimated in hundreds of thousands [1].

At the same time, classic algorithms such as k-means, mean shift, and their analogs and modifications are relatively efficient at the construction of spherical clusters because of the usage of the distance function. However, they have problems with building arbitrary shaped clusters.

As a result, clustering becomes a bottleneck in modern computer vision algorithms. Therefore, despite all the advantages of clustering, it is often abandoned due to the unacceptable computational complexity and/or inadequacy of processing clusters of arbitrary shape.

To solve these problems, in this work we propose a fast algorithm for constructing clusters of arbitrary shape. The parameterization of the algorithm makes it possible to effectively adapt it both for relatively powerful server systems and relatively less powerful embedded systems. Unless otherwise stated, the input data is a color image in RGB format.

2. Related Works

In recent decades, the k-means clustering method has gained wide popularity in image analysis, serving as an effective and powerful tool for data grouping. Clustering color images using k-means is an active area of research that attracts the attention of researchers in the fields of computer vision and image processing.

In their works, the authors explore the application of the k-means algorithm in various domains of human activity, such as color image quantization [14], sky image segmentation [15] and medicine[16].

The authors introduces a clustering algorithm to address cluster size bias, a common issue in conventional methods like K-means [17]. While balanced K-means provides equal-sized clusters, it is slow. The proposed heuristic algorithm offers a faster alternative with reduced bias. It successively divides larger clusters and optimizes centroids when the desired number is reached, allowing for bias and error adjustment.

The accurate estimation of the number of materials in a hyperspectral image is crucial for various hyperspectral image-processing tasks, such as classification and unmixing. In a work [18], authors introduced an algorithm utilizing clustering principles for estimating the number of materials in the image.

An important direction is the processing of low-quality images with noise or pixel losses.

In paper [19], the authors propose to reduce the computational complexity of spectral image clustering methods by introducing a sub-sampling procedure that cuts the number of pixels for classification in half. This preserves the spatial structure of the image.

One of the promising directions in the field of clustering is methods based on the principles of sparse subspace clustering, which group spectral signatures, ensuring their sparse representation. In recent years, there has been a growing number of works [20-22] on the topic of spectral clustering.

In the research addressed by this article [20], the authors tackle the issue of clustering spectral images, aiming to identify groups and distributions within spectral signatures without the need for a prior training stage. Methods based on sparse subspace clustering group spectral signatures into different subspaces, seeking the least dense representation for each pixel and ensuring their affiliation with the same class. Despite their high accuracy, these methods encounter the challenge of increasing computational complexity as the number of pixels grows. This work proposes an approach to reduce the number of pixels for the classification of spectral images by half through a subsampling procedure that eliminates every second adjacent pixel while preserving the spatial structure of the image.

In the field of hyperspectral image (HSI) clustering, the extraction of valuable clustering information can be utilized for the classification of real objects, environmental monitoring, and other tasks. Spectral clustering stands out as one of the most popular clustering methods and has been successfully applied in hyperspectral image clustering, garnering significant attention.

However, the majority of these methods do not take advantage of the spatial information in HSI, which could enhance pixel correlation and improve accuracy. In this paper [21], based on the physical characteristics of HSI, a new approach is proposed, named hyperspectral image clustering based on spatial information and spectral clustering (SISC). By combining spatial window and spectral factors, the algorithm utilizes joint spatial-spectral information, reconstructs the central point, and reveals local spatial structure using nearby spatial points.

The field of compressive spectral imaging (CSI) involves acquiring random projections of a spectral scene. Conventionally, a computationally expensive reconstruction of the underlying 3D scene is required before applying any post-processing tasks such as clustering. To enhance the quality of reconstruction and subsequent post-processing results, prior works have focused on adaptively designing sensing matrices. In contrast, this paper [22] introduces a novel hierarchical adaptive approach for the design of a sensing matrix in the single-pixel camera. The proposed approach enables pixel clustering directly in the compressed domain. At each step of the hierarchical model, a sensing matrix is tailored to facilitate the extraction of clustering features directly from the compressed measurements. The final segmentation map is obtained through majority voting from partial clustering results at each hierarchy step.

Neural networks provide a powerful tool for extracting high-level features from images, making them promising in the field of image clustering. Various architectures of neural networks, including deep convolutional neural networks (CNN), autoencoders, and generative adversarial networks (GAN), are actively explored to enhance the clustering process and improve accuracy in identifying similar patterns. Works [23-27] present key trends and outcomes of applying neural networks in image clustering tasks, while also identifying challenges and prospects in this research area.

3. Methods and Materials

3.1. Image Preprocessing

Considering the architecture and algorithms of modern computer vision systems and, above all, neural networks, the stage of image preprocessing is usually distinguished. This stage is designed for the rapid preparation of the image for the most efficient processing.

In this regard, the first step in the preprocessing stage is to downscale the image.

The downscaling factor and the anti-aliasing filter are usually selected experimentally, according to efficiency requirements, taking into account the characteristics of the computer vision system equipment. The authors of this paper carried out a series of experiments, and the following conclusions were made.

When the scale is reduced once, a square window with linear size being an integer greater than 1 (2, 3, …) is used. If you plan to reduce the scale iteratively and work with the pyramid of images, then the linear size of the window, as when working with SNS [1], should be a multiple of a power of an integer greater than 1. To quickly and smoothly change the scale without losing significant details of the scene, powers of 2 (2, 4, 8, ...) are usually considered.

With regard to smoothing filters, the authors have tested thinning, averaging, and a series of adaptive filters [8]. It was found that thinning is the best in terms of time efficiency, and in most cases is only slightly inferior in quality in comparison with other filters. With high input image quality, thinning is not inferior in quality with respect to other filters at all. Therefore, it is recommended to use the thinning filter by default.

The following averaging filter (1) is optimal for the most realistic transmission of small details if enough computational resources are present and when processing small images. Also, in such conditions, this filter is optimal for noise smoothing in terms of the balance of time and quality

$$r = \frac{1}{W}\sum_{i=1}^{n} r_i \cdot w_i\,,\ W = \sum_{i=1}^{n} w_i, \qquad (1)$$

where r_i is the brightness level for the component of the color model in the neighborhood (each component is processed separately from each other), and w_i are the weight coefficients [1, 8]. In practice, the mean filter is most often used with $w_i = 1$.

Adaptive filters, especially with large window sizes, unacceptably smooth small details, lines and fragments of object boundaries [1, 8]. They are also characterized by a nonlinear (most often quadratic) estimate of the complexity. Therefore, their usage is unacceptable without special justification.

To reduce the complexity of clustering, in addition to reducing the scale, it is proposed to optimize the partitioning grid of the RGBH space, where H(r, g, b) is the frequency of values (r, g, b) in the RGB space.

When preparing data for clustering, an image is scanned and a matrix H(r, g, b) is formed. The cells of this matrix store data on the frequency of distribution of colors (r, g, b) in the pixels of the image. Then the clustering of color areas in RGB space is performed using this matrix.

For a standard color image (type 24 bit) in the RGB model, $2^8 \cdot 2^8 \cdot 2^8 = 16\,777\,216$ possible colors are defined. For many applications, given the iterative nature of clustering, processing such a large number of cells is not acceptable in terms of computational complexity.

This is also unnecessary because the analysis of the experiment results shows that the contents of many cells are not informative.

It is advisable to reduce the number of cells in RGB space in such a situation. To do this, we enlarge the RGB space partition grid using the factor $d = 2^n$ (Fig. 1).

Numerous experiments have shown that the best results are obtained for steps 4 and 8. In most experiments, it was possible to reduce the complexity of clustering by about 64 (512) times without reducing the quality by using enlarged cells $4 \cdot 4 \cdot 4$ (or $8 \cdot 8 \cdot 8$).

Under such conditions, the RGB space is first divided into enlarged cells with a step d, after which the frequency is calculated for these cells by adding the frequencies of the unit cells (r, g, b).

For an adequate construction of clusters of arbitrary shape, it is planned (this will be described below) to analyze the level lines of the function H(r, g, b).

Analysis of typical distributions H(r, g, b) showed the following.

Almost any image has one or several dominant objects.

The frequency of the chromaticity region of the dominant object, as a rule, is several orders of magnitude higher than the frequencies of other objects in the image. Visually, this frequency looks like a separate peak on the histogram. In this situation, consideration of all lines of the distribution level H(r, g, b) is redundant and unnecessarily laborious.

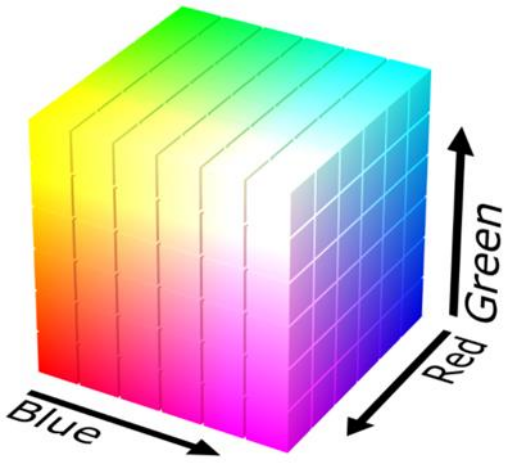


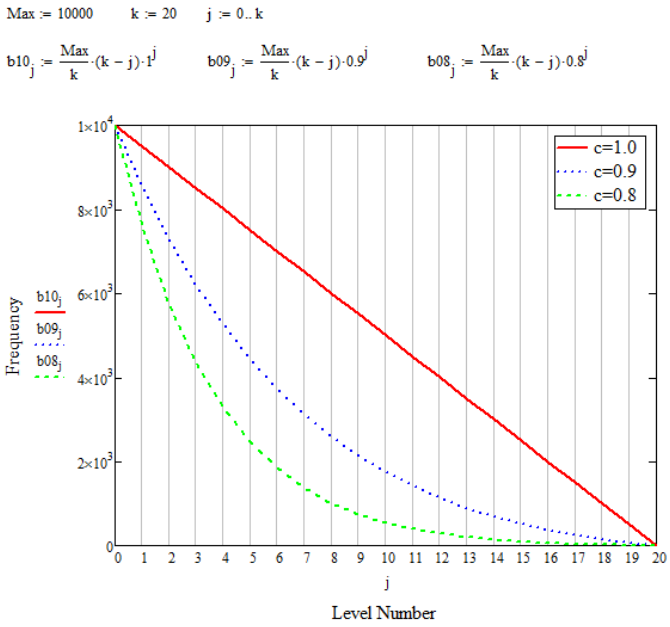
Figure 1: Enlarging the RGB space partition grid [28]

It is proposed to quantize the frequency into a predetermined (optimized at the training stage) number of levels k to find a balance between computational complexity and quality. For this, a non-linear quantization law (analog of log scale) is used to neutralize the influence of dominant

objects in the image. For these purposes, a flexible function of forming the boundaries of k frequency quantization levels is proposed

$$b_j = \frac{Max}{k}(k-j)\cdot c^j, \qquad (2)$$

where b is the quantization boundary of the level j, $j = 0, \dots, k$; Max is the maximum frequency, c is the peak frequency suppression coefficient of dominant objects (Fig. 2).



a)

b10j =	b09j =	b08j =
10000	10000	10000
9500	8550	7600
9000	7290	5760
8500	6196.5	4352
8000	5248.8	3276.8
7500	4428.675	2457.6
7000	3720.087	1835.008
6500	3108.93	1363.149
6000	2582.803	1006.633
5500	2130.813	738.198
5000	1743.392	536.871
4500	1412.148	386.547
4000	1129.718	274.878
3500	889.653	192.415
3000	686.304	131.941
2500	514.728	87.961
2000	370.604	56.295
1500	250.158	33.777
1000	150.095	18.014
500	67.543	7.206
0	0	0

b)

Figure 2: Listing, where: a) family of functions for frequency quantization; b) boundaries of quantization levels in numerical form

After all the parameters of the models are determined, the image preprocessing is performed according to the following algorithm:

1) the image is resized to the required size;

2) the resulting image is scanned:

- cell frequencies are found on the enlarged grid in RGB space;

- the found frequencies are quantized according to (2). After that, the quantization levels $j = 1, \dots, k$ are used instead of frequencies.

3.2. Data Clustering

The clustering algorithm is iterative. External iterations of the algorithm are used to enumerate and sequentially consider the levels ($j = 1, \dots, k$) of quantizing the cell frequency in the RGB cube. Internal iterations are used to build clusters at the current quantization level, taking into account the clustering results at previous levels.

So, the levels of quantization are viewed from top to bottom, starting from the most significant levels, ($j = 1, \dots, k$).

For level j, clusters of adjacent cells with a frequency at level j are constructed using the wave method. In 3-dimensional RGB space, two distinct cells (r_ξ, g_ξ, b_ξ) and (r_η, g_η, b_η) are considered adjacent if the following condition is met

$$|r_\xi - r_\eta| \le 1 \; AND \; |g_\xi - g_\eta| \le 1 \; AND \; |b_\xi - b_\eta| \le 1. \qquad (3)$$

These clusters are numbered in ascending order. Clusters built from cells of the same level will be called simple. Adjacent clusters built on the current and one of the previous levels, if any, are merged. Clusters that have adjacent cells are considered adjacent. A cluster resulting from the merging of clusters is called a composite cluster. Just like a simple cluster, it gets a new number. We use continuous numbering for numbering of clusters (Fig. 3 – Fig. 5).

For the stability of the clustering method to noise and shadows of objects (so that clusters are not falsely combined into one cluster), at least the lowest level $j = k$ is not considered when constructing clusters.

In this case, we can lose small-sized objects. In this case, we can lose small-sized objects, but we will maintain an adequate nested structure of large objects. During the process of building clusters, information about them at each iteration is stored in three tables.

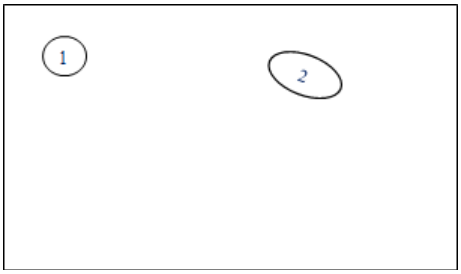


Figure 3: The result of the first iteration ($j = 1$). Two simple clusters at the highest level of significance have been formed.

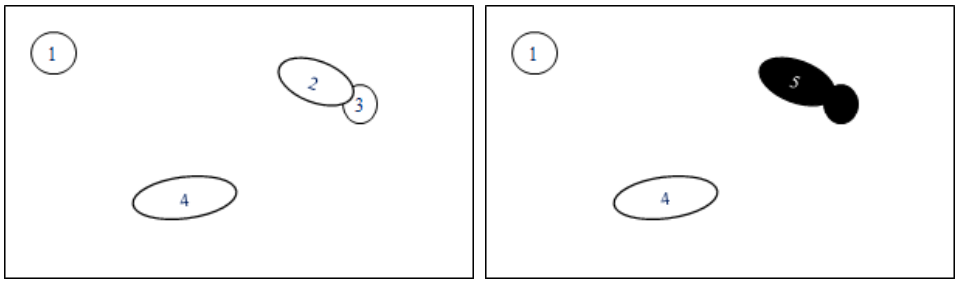


Figure 4: The result of the second iteration ($j = 2$). Two simple clusters numbered 3 and 4 have been formed. After that, a composite cluster with number 5 has been formed by combining clusters number 2 and 3.

The first table (Cluster) is used to store general information about all clusters (Table 1, Table 4, Table 7).

This table stores the cluster number (continuous numbering), cluster type (simple/composite), level number ($j = 1, \dots, k$) on which the cluster is built and the cluster number in the simple/composite clusters table.

The second table (S-Cluster) is used to store information about simple clusters (Table 2, Table 5, Table 8).

This table stores the number of a simple cluster in the continuous numbering system of simple clusters, the number of a simple cluster in the Cluster table and a reference to the array of coordinates of cells $\{ [r, g, b]_i \}_i$ from which the simple cluster is built.

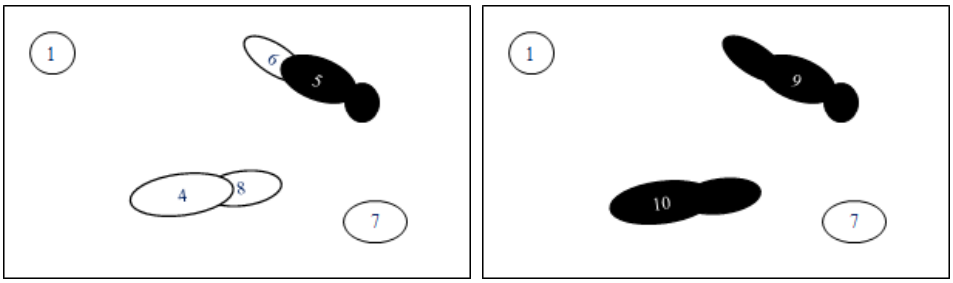


Figure 5: The result of the third iteration ($j = 3$). Three simple clusters with numbers 6, 7 and 8 have been formed. After that, two composite clusters with numbers 9 and 10 have been formed.

The third table (C-Cluster) is used to store information about composite clusters (Table 3, Table 6, Table 9).

This table stores the composite cluster number in the continuous composite cluster numbering system, the composite cluster number in the Cluster table, and the tuple of clusters from which the composite cluster is built.

Table 1
Cluster (After 1st Iteration).

Cluster Number	Cluster Type	Level Number	Cluster Number in S- / C-Cluster Table
1	S	1	1
2	S	1	2

Table 2
S-Cluster (After 1st Iteration)

S-Cluster Number	S-Cluster Number in Cluster Table	S-Cluster Cell Array Reference
1	1	SC[1]
2	2	SC[2]

Table 3
C-Cluster (After 1st Iteration).

C-Cluster Number	C-Cluster Number in Cluster Table	C-Cluster Tuple of Clusters

Table 4
Cluster (After 2nd Iteration).

Cluster Number	Cluster Type	Level Number	Cluster Number in S- / C-Cluster Table
1	S	1	1
2	S	1	2
3	S	2	3

4	S	2	4
5	C	2	1

Table 5
S-Cluster (After 2nd Iteration).

S-Cluster Number	S-Cluster Number in Cluster Table	S-Cluster Cell Array Reference
1	1	SC[1]
2	2	SC[2]
3	3	SC[3]
4	4	SC[4]

Table 6
C-Cluster (After 2nd Iteration).

C-Cluster Number	C-Cluster Number in Cluster Table	C-Cluster Tuple of Clusters
1	5	(2,3)

Table 7
Cluster (After 3rd Iteration).

Cluster Number	Cluster Type	Level Number	Cluster Number in S- / C-Cluster Table
1	S	1	1
2	S	1	2
3	S	2	3
4	S	2	4
5	C	2	1
6	S	3	5
7	S	3	6
8	S	3	7
9	C	3	2
10	C	3	3

Table 8
S-Cluster (After 3rd Iteration).

S-Cluster Number	S-Cluster Number in Cluster Table	S-Cluster Cell Array Reference
1	1	SC[1]
2	2	SC[2]
3	3	SC[3]
4	4	SC[4]
5	6	SC[5]
6	7	SC[6]
7	8	SC[7]

Table 9
C-Cluster (After 3rd Iteration).

C-Cluster Number	C-Cluster Number in Cluster Table	C-Cluster Tuple of Clusters
1	5	(2,3)

2	9	(5,6)
3	10	(4,8)

The use of the proposed data structures makes it possible to effectively organize and use a relational-hierarchical model for storing and processing information about clusters, taking into account their nested content.

Additionally, you can store the Levels table (Table 10) with the number of the first record of each clustering level ($j = 1, \dots, k$) in the Cluster table for convenience.

Table 10
Levels (After 3rd Iteration).

Level Number	First in Cluster Table
1	1
2	3
3	6

4. Experiment

During the experiment, two images of automobiles were utilized (Fig. 6). The dimensions of each image were 1 megapixel.

a)

b)

Figure 6: Images of car for experiments [29, 30]; dominant objects – sky, road surface, car

To delineate homogeneous regions during the experiment, two stages were implemented.

At the first stage, image preprocessing was conducted using the method described in Section 3.1. To mitigate clustering complexity, an optimization of the RGBH color space partitioning grid was applied, where H(r, g, b) represents the frequency of values (r, g, b) in the RGB color space.

For the second stage of the experiment, five data arrays were formed during preprocessing, each representing a one-dimensional matrix of color frequencies on the processed image in the RGB palette:

Unoptimized RGB palette - 16,777,216 possible colors (2^8·2^8·2^8)

Utilizing a 2x2x2 cell, where the frequency of consecutive, neighboring colors in the 2x2x2 cell is summed into one – 2,097,152 possible colors

Utilizing a 4x4x4 cell, where the frequency of consecutive, neighboring colors in the 4x4x4 cell is summed into one – 262,144 possible colors

Utilizing an 8x8x8 cell, where the frequency of consecutive, neighboring colors in the 8x8x8 cell is summed into one – 32,768 possible colors

Utilizing a 16x16x16 cell, where the frequency of consecutive, neighboring colors in the 16x16x16 cell is summed into one – 4,096 possible colors

The algorithm for the experiment was implemented in C# on the .NET Framework platform. Visualization of the second stage, specifically clustering, was conducted using the Unity 3D [31] Engine.

The second stage, the clustering process, was implemented using the method described in Section 3.2, based on the dataset of color frequencies prepared during preprocessing. The execution time of the clustering stage was measured for each dataset. Unity 3D Engine was employed for visualizing the clustering process.

5. Results

We will use the Unity package and the classic image representation to visualize and assess the quality of clustering results.

As the analysis of the experimental results shows, the low estimates of the complexity of clustering are fully confirmed. The time of even an unoptimized single-threaded clustering of a 1 MP image (processor - Intel Core i5 6600k 3.7 GHz, RAM - 16 Gb DDR4) for practically significant 4x4x4 cells is approximately 0.25 seconds, and for 8x8x8 cells is approximately 0.0225 seconds.

In terms of quality, the situation is as follows. In the first approximation, all objects in the image can be divided into 2 main categories: monochromatic and non-monochromatic.

The quality of clustering of monochromatic objects can be considered quite acceptable. As for non-monochromatic objects, instead of a single tone with a high frequency we get a large number of blurry tones of an object with a low frequency.

The corresponding clusters are built and combined with each other at the lower levels. In the worst conditions, they are combined at the lowest level. At the same time, they are also combined with adjacent clusters for other objects. As a result, the colors of several objects form one cluster.

Let's consider the situation with examples. As initial data, consider the cars shown in Fig. 6 (clustering parameters: number of levels $k = 20$, suppression coefficient $c = 0.75$, cell size $8 \cdot 8 \cdot 8$).

Let's start by analyzing the “good” data for the comparatively monochromatic car in Fig. 6.a. As expected, the dominant objects with a high frequency were the first to stand out relatively well. At first, a monochromatic sky was clustered by the 9th level (Fig. 7.a). Then a cluster with shades of black stood out by the 13th level (Fig. 7.b). After that, the cluster corresponding to the one-color part of the car was constructed by the 15th level (Fig. 7.c).

After that, at level 15, the unsaturated colors (located near the axis of grayscale) began to clearly combine with each other (Fig. 8).

a) b)

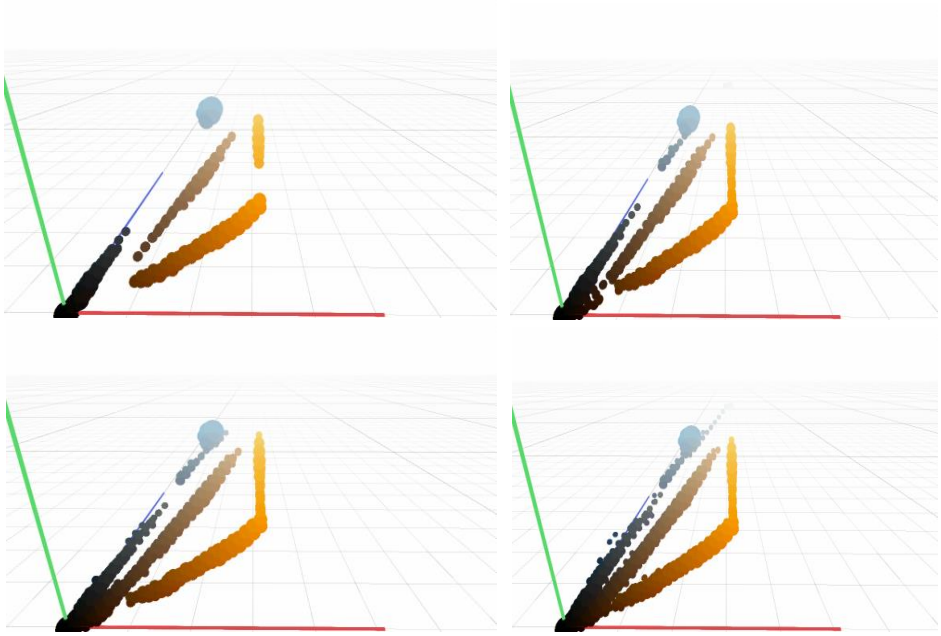
c)

Figure 7: The result of clustering of monochrome fragments of the scene in Fig. 6.a.

Why does that happen? To understand the reasons, let's look at what happens in the color space at levels 13 (top left) to 20 (bottom right) without levels 18 and 19 using Unity (Fig. 9). Starting from level 14, the clusters gradually merge in the color space, especially near the gray scale axis. This is due to the construction of clusters with colors corresponding to different objects.

a) b)

Figure 8: The result of combining unsaturated fragments of the scene in Fig. 6.a. (Level 15).



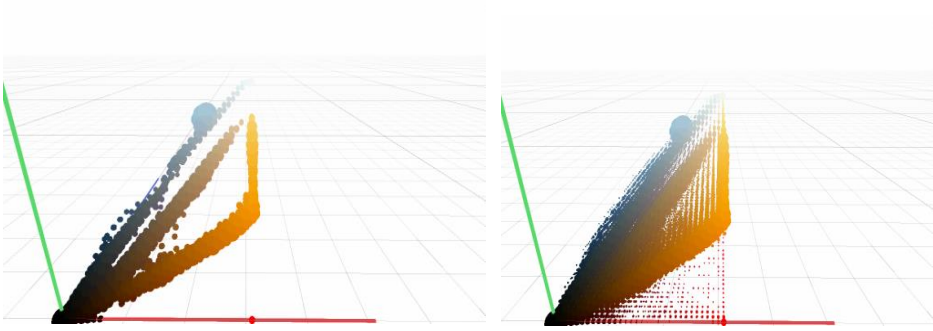
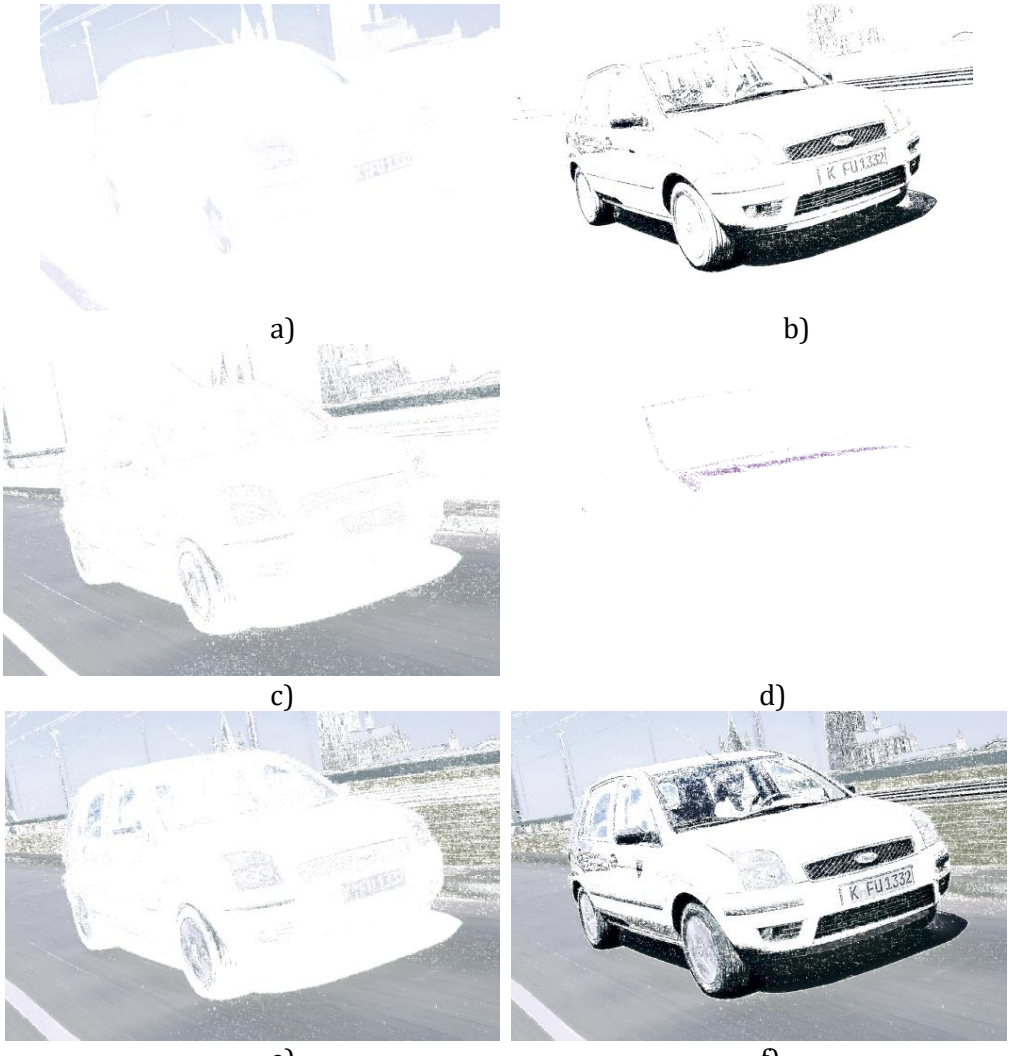


Figure 9: The result visualization in Unity (distribution of pixel color frequencies in the RGB (0-256) palette for the analyzed image, with color values on the axes (red, green, blue)).

Now, let's consider the "bad" data in the same initial conditions. Let's consider a non-monochromatic (due to uneven lighting) car in Fig. 6.b. In terms of dominant objects, blue, gray, and black objects clustered relatively well by the 10th level (Fig. 10.a-c). The colored fragment of the car appears only at the 11th level (Fig. 10.d). And at the same level, the merging of clusters begins significantly in the region of unsaturated colors (Fig. 10.e). This is even more noticeable at the 12th level (Fig. 10.f).



a) b)

c) d)

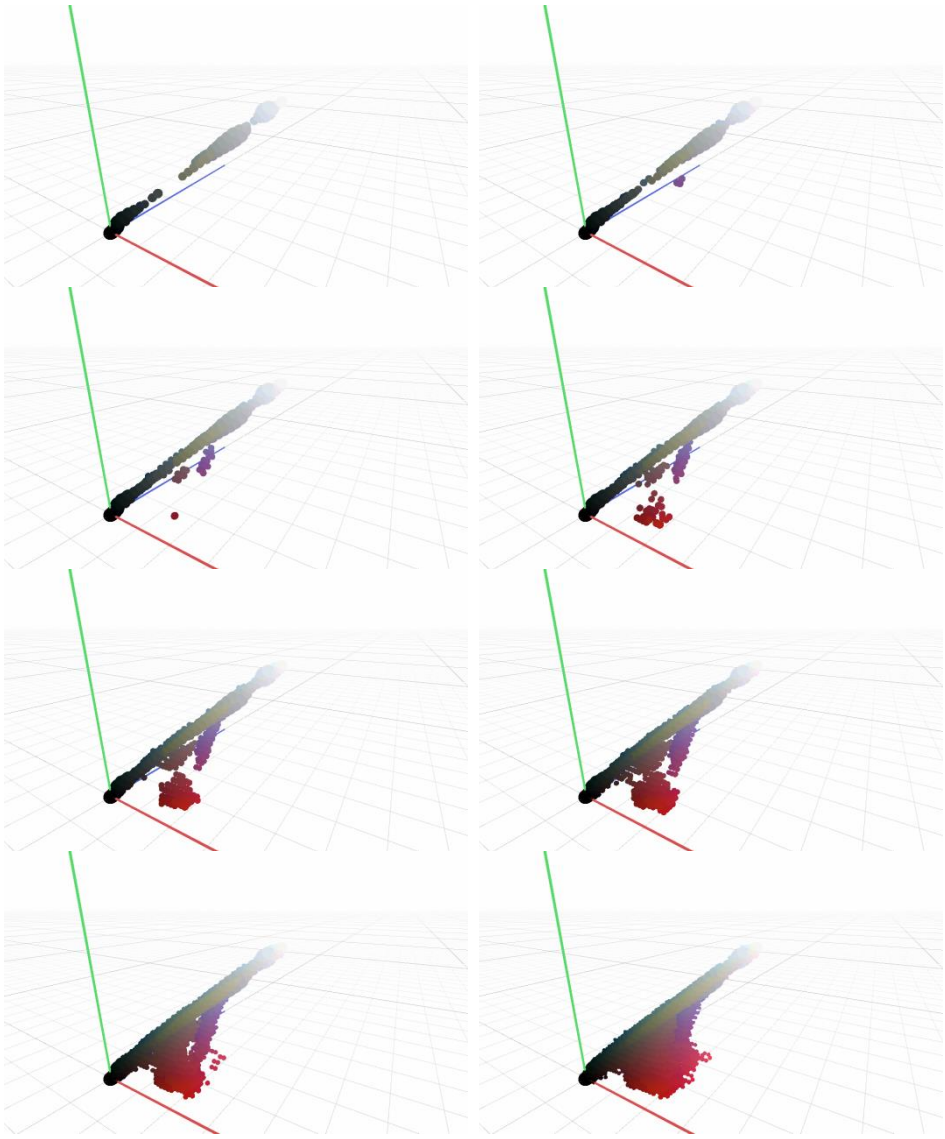
e) f)

Figure 10: The result of car clustering in Fig. 6.b.

The colored part of the car stands out more or less only at the 14th level and immediately merges into one cluster with other objects (Fig. 11). This is because of the significant range of colors of the red-violet part of the car. This is because of the significant spread of colors of the red-violet part of the car and the inseparability of this cloud in the RGB cube from adjacent colors (Fig. 12).

a) b)

Figure 11: The result of car clustering in Fig. 6.b.



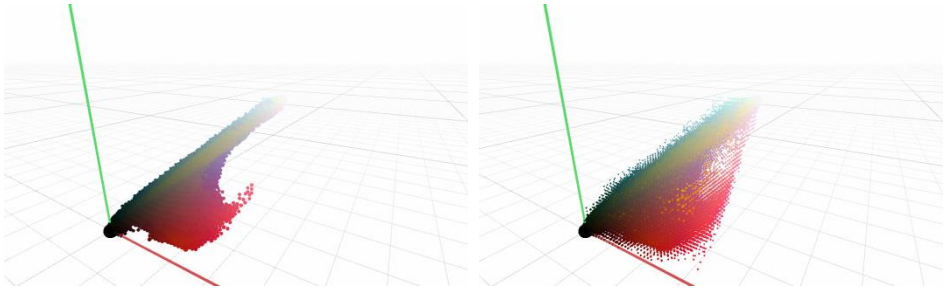


Figure 12: The result visualization in Unity (distribution of pixel color frequencies in the RGB (0-256) palette for the analyzed image, with color values on the axes (red, green, blue)) (levels 10 – 18, 20).

All clustering results (Fig. 9 and Fig. 12) show an interesting effect. Saturated colors of objects, whatever their spread in the RGB cube is, are rather compactly located in the plane along the H component of the HSV model. Further, this 3D data can be further processed and filtered.

Moreover, the saturation S of the object also changes slightly for "good" data (orange car in Fig. 9). Because of this, the shape of the object's cluster clearly repeats the triangular section formed by the color half-plane and the edges of the RGB cube.

The clustering results of the considered images at all levels can be found here [32].

6. Discussions

The obtained results of the experiments indicate the high efficiency of the proposed method for clustering color images. The clustering demonstrates the capability of effectively extracting color regions, rendering it promising in the context of real-time computer vision and artificial intelligence systems.

One of the key merits of the developed system is its high processing speed in real-time mode. As evident from the table, the utilization of merging adjacent color frequencies into 4x4x4 and 8x8x8 cells during the preprocessing stage yields minimal temporal delays in clustering while maintaining satisfactory quality. This feature renders the system effective and swift in conditions requiring prompt data processing.

Table 11
Clustering results.

Size of cell	Number of colors	Time, sec	Clustering efficiency
1x1x1	16 777 216	120-180	Excellent
2x2x2	2 097 152	10	Excellent
4x4x4	262144	0,25	Excellent
8x8x8	32768	0,0225	Good
16x16x16	4096	<0,01	Bad

The assessment of clustering quality confirms a sufficiently high accuracy and reliability in delineating various color regions in images. The algorithm's performance leads to clear and distinguishable groups, affirming its applicability across a broad spectrum of visual data.

Furthermore, the system demonstrated robustness to variations in conditions, such as changes in lighting, image resolution, and the presence of noise. Comparative analysis with other methods highlights a key advantage of the developed system in that, unlike the k-means method, it does not necessitate a predetermined number of clusters. The k-means method assumes that the number of clusters is known in advance, which can be a limitation in real-world scenarios where the number of clusters may be variable and unknown.

The developed system showcases automatic determination of the optimal number of clusters based on intrinsic data characteristics. This significantly enhances the method's flexibility and applicability in situations where the number of clusters to be delineated in an image is not pre-determined. Such an approach is particularly valuable in real-time conditions where instantaneous adaptation to changing conditions and data dynamics is required. The automatic determination of the number of clusters reduces the need for preconfiguration, making the method more convenient and versatile for various usage scenarios.

The developed clustering method has a wide range of practical applications, including real-time streaming data processing, automatic object recognition in images, and optimization of computer vision processes in the field of artificial intelligence.

Possible directions for future research include refining the method to handle different types of images, expanding functionality for multitasking scenarios, and conducting in-depth investigations into the impact of parameters on clustering outcomes.

7. Conclusions

The paper proposes a fast clustering algorithm based on the consistent use of preprocessing and clustering of image pixels in RGB space.

At the preprocessing stage (a), the scale is first reduced and (optionally) the image is smoothed, (b) the aggregated color frequencies are found on the enlarged RGB space grid, (c) the adaptive quantization of the cell frequencies of such a grid in RGBH is performed.

The proposed preprocessing algorithm allows you to quickly prepare data and reduce by several orders of magnitude the number of enumerated space cells when building clusters in RGBH. Accordingly, the complexity of data clustering in RGBH is decreased. We obtain the following conclusions based on the averaged experimental data (scaling is not taken into account as a standard procedure): 1) the grid sampling is most often done with a factor of 4 (or 8), so the number of cells in the RGBH space is reduced by 64 (or 512) times; 2) the number of quantization levels usually ranges between 16 and 32, while the frequency of some cells can be up to 10,000 or more. In this situation, the number of frequency gradations decreases from 100 to 1000 times.

As a result, the proposed algorithm makes it possible to cluster data almost instantly.

The following computing system was used for the purposes of the experiments: CPU - Intel Core i5 6600k 3.7 GHz, RAM - 16 Gb DDR4. The software was run locally in a single thread. Color images with a size of 1 MP are processed, the number of levels is $k = 20$. In these conditions, the clustering time is: a) for 1x1x1 cells - several minutes, the result is not very stable (it has no practical sense); b) for 2x2x2 cells - about 10 seconds; c) for 4x4x4 cells - about 0.25 seconds; d) for 8x8x8 cells - about 0.0225 seconds. Such a high efficiency of the proposed algorithm (taking into account the possibility of parallel data processing) makes it possible to use it for clustering and segmentation in even relatively low-powered embedded systems.

Due to the wide parameterization, preprocessing algorithms can be fine-tuned to the features of the problem that is being solved.

The proposed clustering algorithm allows to adequately build clusters of arbitrary shape for relatively monochromatic objects taking into account the position of the level lines. It also allows to create a hierarchy of nesting clusters (corresponding to the level lines), which is important for separating objects close in color in the image during subsequent processing. Unfortunately, non-solid objects (usually unevenly lit) may not be processed adequately, since their colors are highly diffused and become adjacent to the colors of other objects.

References

[1] Rafael C. Gonzalez, Richard E. Woods Digital Image Processing, 4th edition Pearson/Prentice Hall, 2018. – 1168p.P. S. Abril, R. Plant, The patent holder's dilemma: Buy, sell, or troll?, Communications of the ACM 50 (2007) 36–44. doi:10.1145/1188913.1188915.

[2] I. Ruban, H. Khtoudov, V. Lishchenko, A. Zvonko, S. Glukhov, I. Khizhnyak, V. Maliuha, Y. Polonskyi, R. Kushpeta, The Calculating Effectiveness Increasing of Detecting Air Objects by Combining Surveillance Radars into The Coherent System // International Journal of Emerging Trends in Engineering Research. – Vol. 8. (№ 4), 2020. – P. 1295-1301.
[3] V. Lishchenko, V. Chaliy, H. Khudov, and A. Zvonko. Proposals for Improving of Air Surveillance Informativity in MIMO Radar Systems Based on Two-Dimensional Radars // Intern. Scient.-Pract. Conf. Problems of Infocommunications. Science and Technology (PIC S&T), 9-12 Oct 2018, Kharkiv. – P. 153–156.
[4] S. Orozco-Arias *et al*., "SENMAP: A Convolutional Neural Network Architecture for Curation of LTR-RT Libraries from Plant Genomes," *2021 IEEE 2nd International Congress of Biomedical Engineering and Bioengineering (CI-IB&BI)*, Bogota D.C., Colombia, 2021, pp. 1-4, doi: 10.1109/CI-IBBI54220.2021.9626130.
[5] Lishchenko V., Khudov H., Tiutiunnyk V., Kuprii V., Zots F. and Misiyuk G. The Method of Increasing the Detection Range of Unmanned Aerial Vehicles In Multiradar Systems Based on Surveillance Radars // IEEE 39th International Conference on Electronics and Nanotechnology (ELNANO), 16-18 April 2019, Kyiv, Ukraine. – P. 559-562.
[6] C. Mejia-Escobar, M. Cazorla and E. Martinez-Martin, "Improving Facial Expression Recognition Through Data Preparation and Merging," in IEEE Access, vol. 11, pp. 71339-71360, 2023, doi: 10.1109/ACCESS.2023.3293728.
[7] B. Zhao, J. R. Sveinsson, M. O. Ulfarsson and J. Chanussot, "(Semi-) Supervised Mixtures of Factor Analyzers and Deep Mixtures of Factor Analyzers Dimensionality Reduction Algorithms For Hyperspectral Images Classification," *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, Yokohama, Japan, 2019, pp. 887-890, doi: 10.1109/IGARSS.2019.8898932.
[8] Ruban I., Churyumov G., Tokarev V., Tkachov V. Provision of Survivability of Reconfigurable Mobile System on Exposure to High-Power Electromagnetic Radiation // Selected Papers of the XVII International Scientific and Practical Conference on Information Technologies and Security (ITS 2017). – CEUR Workshop Processing. – Kyiv, Ukraine, 2017. – P. 105-111.
[9] Serkov A., Kravets V., Yakovenko I., Churyumov G., Tokariev V., Nannan W. Ultra Wideband Signals in Control Systems of Unmanned Aerial Vehicles // IEEE International Conference on "Dependable Systems, Services and Technologies", (DESSERT'2019). – 2019, Leeds, United Kingdom, June 5-7. – P. 26-29.
[10] Tkachov V., Tokariev V., Dukh Y., Volotka V. Method of Data Collection in Wireless Sensor Networks Using Flying Ad Hoc Network // IEEE International Scientific-Practical Conference "Problems of Infocommunications, Science and Technology", (PIC S&T). – 2018, Kharkiv, Ukraine, 9-12 October. – P. 197-201.
[11] Yevdokymenko M. Investigation of Tensor Approach for Providing Multimedia Quality in Infocommunication Networks // Ceur Workshop Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019). – 2019. – Vol. 2654. – P. 227-239.
[12] Yevdokymenko M., Mohamed E., Onwuakpa P. Ethical hacking and penetration testing using raspberry PI // 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, 2017. – P. 179-181.
[13] Kuzminykh I., Carlsson A., Yevdokymenko M., Sokolov V. Investigation of the IoT device lifetime with secure data transmission // In: Galinina O., Andreev S., Balandin S., Koucheryavy Y. (eds) Internet of Things, Smart Spaces, and Next Generation Networks and Systems. NEW2AN 2019, ruSMART 2019. Lecture Notes in Computer Science, Springer, Cham. Vol. 11660. – P. 16-27.
[14] H. Palus and M. Frackiewicz, "Deterministic vs. Random Initializations for K-Means Color Image Quantization," 2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Sorrento, Italy, 2019, pp. 50-55, doi: 10.1109/SITIS.2019.00020.

[15] S. Dinc, R. Russell and L. A. C. Parra, "Cloud Region Segmentation from All Sky Images using Double K-Means Clustering," 2022 IEEE International Symposium on Multimedia (ISM), Italy, 2022, pp. 261-262, doi: 10.1109/ISM55400.2022.00058.
[16] F. Lopez-Tiro, H. Peregrina-Barreto, J. Rangel-Magdaleno and J. C. Ramirez-San-Juan, "Localization of blood vessels in in-vitro LSCI images with K-means," 2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Glasgow, United Kingdom, 2021, pp. 1-5, doi: 10.1109/I2MTC50364.2021.9460100.
[17] A. Ito, "Successive Binary Partition K-means Method for Clustering with Less Cluster Size Bias," 2022 7th International Conference on Signal and Image Processing (ICSIP), Suzhou, China, 2022, pp. 772-776, doi: 10.1109/ICSIP55141.2022.9886452.
[18] J. Prades, A. Salazar, G. Safont and L. Vergara, "Experimental Study of Hierarchical Clustering for Unmixing of Hyperspectral Images," 2021 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES), Bali, Indonesia, 2021, pp. 1-5, doi: 10.1109/ICARES53960.2021.9665201.
[19] J. Bacca, K. Sanchez and H. Arguello, "Sparse Subspace Clustering for Hyperspectral Images with Missing Pixels," 2019 XXII Symposium on Image, Signal Processing and Artificial Vision (STSIVA), Bucaramanga, Colombia, 2019, pp. 1-5, doi: 10.1109/STSIVA.2019.8730242.
[20] Y. Wei, C. Niu, H. Wang and D. Liu, "The Hyperspectral Image Clustering Based on Spatial Information and Spectral Clustering," *2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP)*, Wuxi, China, 2019, pp. 127-131, doi: 10.1109/SIPROCESS.2019.8868487.
[21] C. Hinojosa, J. Bacca, E. Vargas, S. Castillo and H. Arguello, "Single-Pixel Camera Sensing Matrix Design for Hierarchical Compressed Spectral Clustering," 2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP), Pittsburgh, PA, USA, 2019, pp. 1-6, doi: 10.1109/MLSP.2019.8918856.
[22] J. Lopez, C. Hinojosa and H. Arguello, "Fast Subspace Clustering Algorithm with Efficient Similarity-Constrained Sampling for Hyperspectral Images," 2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP), Gold Coast, Australia, 2021, pp. 1-6, doi: 10.1109/MLSP52302.2021.9596507.
[23] K. Smelyakov, A. Chupryna, O. Bohomolov and I. Ruban, "The Neural Network Technologies Effectiveness for Face Detection," 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP), 2020, pp. 201-205, doi: 10.1109/DSMP47368.2020.9204049.
[24] K. Smelyakov, A. Chupryna, O. Bohomolov and N. Hunko, "The Neural Network Models Effectiveness for Face Detection and Face Recognition," 2021 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), 2021, pp. 1-7, doi: 10.1109/eStream53087.2021.9431476.
[25] K. Smelyakov, M. Shupyliuk, V. Martovytskyi, D. Tovchyrechko and O. Ponomarenko, "Efficiency of image convolution," 2019 IEEE 8th International Conference on Advanced Optoelectronics and Lasers (CAOL), 2019, pp. 578-583, doi: 10.1109/CAOL46282.2019.9019450.
[26] M. Tonin and R. L. de Queiroz, "On Quantization of Image Classification Neural Networks for Compression Without Retraining," 2022 IEEE International Conference on Image Processing (ICIP), Bordeaux, France, 2022, pp. 916-920, doi: 10.1109/ICIP46576.2022.9897466.
[27] Kyrychenko, I., Nazarov, O., Huliiev, N., Avdieiev, O. “Selection of Artificial Neural Networks for Disease Prediction”, 2023 7th International Conference on Computational Linguistics and Intelligent Systems (COLINS-2023), 2023. – CEUR-WS, 2023, ISSN 16130073. - Volume 3387, PP. 236-248.
[28] Color Space: https://cdn.firespring.com/images/36580bd4-0617-45db-9623-941537eab10d.png
[29] Car Image: https://www.autocentre.ua/wp-content/uploads/2018/05/Die-Besten-bis-5000-Euro-TueV-Report-2018-1200x800-17fa1ff16f0bfc7a.jpg
[30] Car Image: https://auto.ironhorse.ru/wp-content/uploads/2018/06/Q8-side.jpg
[31] https://unity.com
[32] Results: https://drive.google.com/drive/folders/1MHdinKBlqIGuwz0Irtm4zYV10Ca-XLAK