technologies

MDPI

Article

Application of a Linear Hash Function in Adaptive Image Steganography

Elmira Daiyrbayeva [1,2], Ekaterina Merzlyakova [3], Aigerim Yerimbetova [2,4,*], Lyailya Cherikbayeva [5,6], Bekturgan Akhmetov [7,*], Nurzhigit Smailov [1,*] and Gulmira Shangytbayeva [8]

1 Institute of Automation and Information Technologies, Satbayev University, Almaty 050010, Kazakhstan; e.daiyrbayeva@satbayev.university
2 Institute of Information and Computational Technologies CS MSHE RK, Almaty 050000, Kazakhstan;
3 Faculty of Information Science and Computer Engineering, Siberian State University of Telecommunications and Information Science, Novosibirsk 630102, Russia; katerina.artist@yandex.ru
4 School of Engineering and Information Technology, META University, Almaty 050012, Kazakhstan
5 Department of Computer Science, Al-Farabi Kazakh National University, Almaty 050040, Kazakhstan; cherikbayeva.lyailya@gmail.com
6 Department of Computer Engineering, Almaty Technological University, Almaty 050012, Kazakhstan
7 Construction and Technical College, Almaty 050062, Kazakhstan
8 Department of Computer Science and Information Technology, K. Zhubanov Aktobe Regional University, Aktobe 030000, Kazakhstan; g.shangytbayeva@leeds.ac.uk
* Correspondence: aigerian8888@gmail.com (A.Y.); abs83bek83@mail.ru (B.A.); n.smailov@satbayev.university (N.S.)

Abstract

This paper discusses an adaptive method of image steganography issues based on the application of a linear hash function over the GF (2) field to control the embedding process. The method uses staggered splitting of an image into 8 × 8-pixel blocks to provide blind steganography. Classification thresholds are defined as the percentiles of the distribution of gradients throughout the image, allowing for efficient load distribution between textured and smooth areas. Experiments on the BOSSBase, SIPI and Kaggle kits show that the method provides an actual capacity of up to 0.7 bpp at PSNR 47–50 dB and is resistant to statistical tests and RS analysis. At the same time, like other approaches based on modification of pixel differences, it remains vulnerable to modern stegoanalysis based on spatial rich models (SRMs). However, thanks to the modular structure of embedding control based on linear hash function, the proposed architecture allows direct integration with many modern adaptive strategies aimed at minimizing statistical anomalies.

Keywords: steganography; adaptive embedding; linear hash function; chess partition; gradient; difference container; PSNR

Academic Editor: Hongying Meng

Received: 28 February 2026
Revised: 10 April 2026
Accepted: 17 April 2026
Published: 21 April 2026

1. Introduction

The modern digital environment is characterized by huge volumes of transmitted data, which makes information security especially relevant. In addition to traditional methods of protection, such as encryption, technologies aimed at hiding the very fact of transmission of a message are becoming increasingly important. One of the effective solutions in this direction is steganography, dedicated to the hidden introduction of information into ordinary digital objects, which are most often images due to their wide distribution, high redundancy and the ability to mask minor changes without visual loss of quality.

In this regard, the development of adaptive steganographic methods that seek to minimize statistical anomalies due to the local features of the image. Such algorithms

analyze the texture complexity of each site and decide on the amount and method of data to be implemented based on this complexity. This allows data to be placed predominantly in areas with high texture complexity, where natural visual noise effectively masks the introduced distortions, thereby increasing stealth and stability of the stegosystem [1,2].

Adaptive strategies are used both in reversible data hiding (RDH) [3] tasks, where the ability to restore the original image is critical, and in irreversible blind steganography, focused on maximizing stealth at a given capacity.

However, analysis of the evolution of methods shows the existence of two main directions in ensuring stealth. On the one hand, there are highly efficient algorithms that use machine learning and complex distortion functions whose practical applicability is limited by computing power requirements. On the other hand, the development of lightweight solutions that can operate in the conditions of limited resources characteristic of real-time systems, mobile and IoT devices remains relevant.

Within the framework of this work, an original approach to adaptive steganographic embedding is proposed, in which the order of modification of pixels is controlled by a linear hash function over the GF (2) field. Unlike modern resource-intensive methods that use deep learning or complex cost functions, our approach is focused on simplicity, transparency and modularity. Thanks to the use of simple bit-to-bit operations (XOR, shifts, polynomial division), the algorithm does not require the connection of heavy libraries, GPU acceleration or the learning stage, which makes it suitable for deployment on devices with limited resources. The algorithm can work effectively on mobile devices, video surveillance systems and IoT platforms, where energy efficiency and processing speed are important.

The structure of the work is as follows: In the second section the review of related works is presented, in the third the materials and methods of the proposed algorithm are described, in the fourth the results of experiments are presented, in the fifth the comparative analysis is performed, and in the last section the main conclusions are formulated.

A thorough examination of contemporary adaptive steganographic methodologies has exposed a discernible disparity between two distinct approaches. On one hand, there are methods that are remarkably robust but are computationally exorbitant, necessitating the utilization of GPUs, deep learning, or an optimization step. On the other hand, there are lightweight algorithms that, despite their statistical robustness, are susceptible to vulnerability. In order to address this gap, the present paper presents a study of a lightweight adaptive information embedding algorithm with deterministic $\mathcal{O}(N \log_{10} N)$ computational complexity, no training step, and the ability to run efficiently on a central processing unit (CPU) without the use of GPUs.

2. Related Works

The fundamental principle of modern adaptive methods is the classification of image blocks based on the analysis of their properties. At the same time, various methods offer alternative solutions for selecting a classifying feature and algorithms for determining thresholds. For example, in the field of reversible data hiding (RDH), where the recovery of the original container requires strict control of distortion, adaptability becomes a determining factor in the effectiveness of methods. For example, in [4], an adaptive RDH method based on the combined use of paired PVO and PEE is proposed; in another study [5], an adaptive method is presented that uses for classification the noise level calculated on the perimeter of the block. Based on this value, blocks are classified into several categories and each category uses its own optimal number of pixels processed in each pass. The method additionally uses pairwise PEE for the most "favorable" pairs of errors, which further reduces distortion and increases PSNR compared to traditional PVO approaches. A promising approach is presented in paper [6], where an adaptive PVO method is pro-

posed, using a global adaptive classification based on the analysis of the histogram of the entire image. Despite the fact that these methods are developed within the framework of reversible data concealment, the principles of adaptive block classification and distortion distribution proposed in them are of considerable interest for the development of persistent blind stegosystems to which the method proposed in our work belongs.

Another approach to adaptability is implemented in [7], where the strategy of Adaptive multiple Histograms Modification (AMHM) is proposed. Unlike methods based on the classification of blocks by complexity, the authors use the absolute distance between correlated pixels within the block to consistently generate error sub histograms. This approach allows to adapt the embedding procedure to the local structure of the image. Although the method is developed in the context of reversible concealment, its adaptive idea is also of interest for blind steganography, where it is necessary to minimize statistical anomalies without knowing the key.

In the field of irreversible adaptive steganography, methods that minimize distortions with the help of cost functions, such as HIgh-pass, Low-pass, Low-pass (HILL) [8] and Wavelet observed Weights (WOWs) [9], have become common. Both approaches implement the principle of content-adaptive embedding, directing changes to textured areas of the image, where natural statistical heterogeneities most effectively mask the implementation. The HILL method determines the cost of changing each pixel based on the high-frequency residues obtained with three directional high-frequency filters. The higher the local complexity, the lower the cost of modification. The WOW method determines the cost of changing each pixel based on the residues in the wavelet domain. To achieve this, the image is decomposed using a bank of Dobeshi directional filters. The obtained coefficients of the high-frequency subranges are interpreted as measures of local complexity: The higher the absolute value of the coefficient, the less noticeable the change in the pixel. UNIWARD's universal approach [10] generalizes this concept by evaluating distortions in the wavelet view regardless of the domain in which the embedding is performed. The cost of changing an individual element is defined as the total relative change in the wavelet coefficients that are affected by this change. Since wavelet coefficients are small in smooth regions and along clear boundaries, even a small perturbation causes significant relative distortion, so the algorithm does not embed into this area. In contrast, in texture and noise-like areas, the coefficients are large, so the same changes lead to little relative distortion, hence such areas are used to hide data.

To increase stealth, methods based on the difference in pixel values (PVD, Pixel-value Differencing) are also widely used [11]. These methods demonstrate resistance to statistical attacks, since they adapt the capacity depending on local contrast. In subsequent works, modifications of PVD are proposed, aimed at eliminating statistical artifacts arising at the boundaries of the range of differences. In particular, in [12], an adaptive PVD method is proposed, in which embedding is performed in three-pixel blocks while maintaining the relative order of brightness and using a dynamic threshold to select only the "safest" textured areas. In [13], a method is proposed in which the number of embedded bits is determined not by the difference of two pixels but by the deviation of the target pixel from the average value of its 5–8 neighbors. This approach provides a more accurate estimate of local contrast and avoids embedding in homogeneous areas where even small changes are easily detected. This approach is further developed in [14], where an adaptive block scheme is introduced based on the partition of the image into non-overlapping blocks of 4×4. However, despite these improvements, PVD-based methods are not aimed at countering modern stegoanalyzers using multidimensional feature spaces, such as the spatial rich model (SRM).

In the context of this work, the method proposed in article [15] is also of interest, which implements the concealment of data not in the image itself, but in the code, book used in the compression of images by the method of vector quantization. The adaptability of the method is provided by an adjustable threshold value, which controls the amount of information to be introduced depending on the properties of the code words. Work [16] is interesting because the authors propose a method that uses fuzzy logic to adapt data into images based on the brightness of each pixel. The entire intensity space (0–255) is divided into five levels on which the number of bits to be inserted depends. Up to three bits can be safely embedded in dark pixels, since distortion in these areas is almost invisible. In medium-dark there can be two bits, and in medium and medium-light pixels one bit each can be enbedded. Light pixels remain unchanged, since even minimal modifications in them easily give out the fact of hiding data. This approach takes into account the peculiarities of human vision and provides high-quality steg images with significant capacity. In experiments, the average values of PSNR $\approx$ 58.6 dB and SSIM $\approx$ 1.00 were achieved, and the real capacity of the method is from 1.0 to 2.0 bits per pixel with a maximum theoretical capacity of 3 bits per pixel. Also, this method demonstrates resistance to stegoanalysis, and the detection accuracy does not exceed 29.3% even with large volumes of hidden data.

Unlike traditional methods, the approach in work [17] combines gradient contrast analysis with the application of threshold values in RGB space, offering embedding in medium-contrast (MCCE) and extreme-contrast (ECCE) channels. The authors conduct a comprehensive stegoanalysis of their proposed methods and demonstrate a statistically significant improvement in quality indicators even with respect to methods based on GAN and other deep learning architectures.

At the same time, the last decade has been actively developing approaches to steganography using machine learning and artificial intelligence methods, such as neural network models for generating stegcontainers (for example, based on GAN—Generative Adversarial Networks). Special attention is drawn to adaptive methods that automatically determine the optimal positions for the introduction of secret data with minimal image distortion. Among the most significant developments is the ASDL-GAN model [18], which implements automatic training of the cost of embedding using generative-adversarial networks. This architecture includes a generator that creates a map of the probability of changing pixels, a data embedding emulator, and a discriminator that evaluates the detectability of hidden information. Then, an upgraded version of UT-SCA-GAN [19] was proposed, replacing the emulator with a differentiable activation function Tanh-simulator, which significantly accelerated network training and improved the quality of the resulting stegcontainers.

Another promising direction is methods based on opposing examples (adversarial examples). Models such as ADV-EMB [20] use gradients obtained from stegoanalyzers to change the cost of embedding in individual pixels, thereby reducing the likelihood of detecting hidden data. At the same time, modern AI-oriented methods, although they demonstrate high resistance to stegoanalysis, require large computational resources, long learning time, and also depend on the quality and size of the training sample. For example, a significant contribution to image steganography is made in work [21], in which the authors implement an adaptive approach to steganographic implementation based on a deep neural network architecture of the "encoder–decoder" type. Modern AI-oriented methods demonstrate high resistance to stegoanalysis, but at the same time require large computational resources, long learning time and depend on the quality of the training sample. In work [22], the scheme of encryption-authentication of images based on pixel adaptive diffusion and singular value decomposition ghost imaging (SVDGI) is proposed. Although this work is in the field of cryptography rather than steganography, it demonstrates the idea of adaptive distortion distribution based on the analysis of the local

structure of the container. In particular, the pixel adaptive diffusion mechanism, in which the modification of each pixel depends on the values of previously processed neighbors, provides a natural distribution of changes according to the local texture of the image. This principle of adaptation to the local structure can be used in steganographic methods to minimize statistical anomalies when embedding data.

The analysis of the evolution of adaptive steganographic methods allows us to state that, despite the high efficiency of algorithms based on machine learning, their practical applicability is limited. This confirms the relevance of the development of lightweight solutions that can function in conditions of limited resources, which determines the purpose of this study.

3. Materials and Methods

In work [23], a method was proposed that allows changing only one bit of the built-in sequence of an empty container so that another longer secret message is inserted into it. This approach is based on the use of linear hash functions, which allows transmitting a secret message with minimal possible distortions of the container statistics.

In our work, we apply this approach in an adaptive steganographic method for embedding data into images. First, we consider in more detail the essence of the idea of using linear hash functions that allow evenly distributing changes in the container, providing maximum capacity with minimal detectability. Such a function should have two properties: mixing property and linearity.

Let λ be a function defined on binary words of length N and taking values from the set of binary words of length m, and $N \geq m \geq 1$. If for any $v \in \{0,1\}^m$, $\Pr\{\lambda(w) = v\} = 2^{-m}$, where w are selected from the set of words $\{0,1\}^N$ independently and with equal probabilities. Then, this function is called a stringing function. Also, the function λ has the linearity property if for any $c, d \in \{0,1\}^N$ the identity $\lambda(c \oplus d) = \lambda(c) \oplus \lambda(d)$ holds.

The idea of our steganographic method is to apply this approach for adaptive block-by-block embedding of information depending on the difference characteristics of local blocks. We propose to divide the image into 8×8 blocks, since this size is small enough to accurately reflect local features of the image, but at the same time it is large enough to provide sufficient data for analysis, such as variance estimation. Most often, 8×8 blocks contain statistically homogeneous areas of the image, which allows us to more accurately classify the block and apply the most appropriate implementation strategy. When processing blocks, we divide the actions for "white" and "black" cells in staggered order. A chess grid is defined as the division of a block into two subsets.

$\beta_{\text{black}} = \{(x,y) \mid (x+y) \bmod 2 = 0\}$—"black" positions,

$\beta_{\text{white}} = \{(x,y) \mid (x+y) \bmod 2 = 1\}$—"white" positions.

First, a pseudo-random seed number is generated to determine the order of block processing. Then, the classification thresholds are calculated. For the source image $I \subset \mathbb{Z}^{H \times W}$, $H = W = 512$, gradients of local blocks $\beta^{16}_{p,q}$ of size 16×16, where p, q are coordinates of blocks, are calculated. This block size minimizes computational complexity while maintaining statistical significance.

$$\Delta(\beta^{16}_{p,q}) = \frac{2}{|M|(|M|-1)} \sum_{\substack{p_i, p_j \in M \\ i<j}} |p_i - p_j|.$$

where M is the set of pixels on the "white" positions of the chess grid.

The calculation of the mean absolute gradient requires only addition, subtraction, and modulus taking operations, which provides linear complexity with respect to the number of pixels analyzed and, unlike the variance, does not require squares calculation and root

extraction. And the mean absolute deviation is less sensitive to single extreme luminance values than the variance, which quadratically amplifies the effects of emissions.

Thresholds t_1, t_2 are defined as the 20th and 60th percentiles of an ordered sample $\{\Delta(\beta_{p,q}^{16})\}$. Then follows the adaptive embedding process, which consists of several stages, which we now look at.

3.1. Classification of Blocks

At this stage, the image is divided into non-intersecting blocks $\beta_{i,j}^{(8)}$ of size 8×8, where i is the row number of the block and j is the column number of the block. To classify individual blocks, a gradient is calculated:

$$\Delta(\beta) = \frac{2}{|M|(|M|-1)} \sum_{i<j,\, p_i, p_j \in M} |p_i - p_j|,$$

where M is the set of white pixels of the 8×8 block. For each block, the type $\tau \in \{1, 2, 3, 4\}$ is calculated:

$$\tau(\beta) = \begin{cases} 1, & \nabla(\beta) < t_1 \text{ and the block is not rejected by condition } \varphi_1, \\ 2, & t_1 \le \nabla(\beta) < t_2 \text{ and the block is not rejected by conditions } \varphi_1(0) \text{ and } \varphi_1(1), \\ 3, & \nabla(\beta) \ge t_2 \text{ and the block is not rejected by conditions } \varphi_1(0), \varphi_1(1), \text{ and } \varphi_1(2), \\ 4, & \text{otherwise (the block is rejected by } \varphi_1(r) \text{ for any } r \in \{0,1,2\} \text{ or by } \varphi_2). \end{cases}$$

where $\varphi_1(r)$ is the uniformity condition for the rth least significant bit ($r \in \{0, 1, 2\}$) among all white pixels of the block. A block is rejected by φ_1 if there exists an r such that the value of the rth bit is the same for all white pixels in the block (all 0 or all 1):

$$\varphi_1(r) = \forall p \in P_{\text{white}} : (p \gg r) \,\&\, 1 = c, \quad c \in \{0, 1\}.$$

A block is considered rejected by φ_1 if $\varphi_1(r)$ is true for at least one $r \in \{0, 1, 2\}$.

φ_2 is the condition for "black" pixels. A block is rejected by φ_2 if more than half of the "black" pixels have extreme values (0 or 255):

$$\varphi_2 = \frac{|\{p \in P_{\text{black}} : p = 0 \text{ or } p = 255\}|}{|P_{\text{black}}|} > 0.5.$$

3.2. Formation of a Differential Container

Next, a container C_τ needs to be formed with type $\tau \in \{1, 2, 3\}$ based on local difference characteristics invariant to uniform shifts of brightness. For each pixel $(x, y) \in \beta_{\text{black}}$, a canonical neighbor is defined $(x, y + \Delta_y) \in \beta_{\text{white}}$ according to the rule:

$$\Delta_y = \begin{cases} +1, & \text{if } x \bmod 2 = 0, \\ -1, & \text{if } x \bmod 2 = 1. \end{cases}$$

This selection ensures topological connectivity and minimizes boundary effects.

The absolute difference in brightness is defined as follows:

$$d_{x,y} = \left| I_{i+x, j+y} - I_{i+x, j+y+\Delta_y} \right|, \qquad (x, y) \in \beta_{\text{black}}^{int},$$

where $\beta_{\text{black}}^{int}$ are the internal "black" pixels having a canonical neighbor within the block. Thus, the number of internal black pixels $|\beta_{\text{black}}^{int}| = 32$ for the 8×8 block.

A container C_τ is a bit sequence formed as follows:

1. For each difference $d_{x,y}$, the lower bits are retrieved:

$$\delta_{x,y} = (d_{x,y} \bmod 2^{\tau})_2.$$

For example, at $\tau = 3$ and $d_{x,y} = 5$ (101_2): $\delta_{x,y} = [1, 0, 1]$ (the senior bit is written first).

2. Pixels from internal "black" pixels $\beta^{int}_{\text{black}}$ are processed in raster scan order, that is, from top to bottom and left to right. Bit groups $\delta_{x,y}$ are concatenated in bypass order, resulting in a sequence:

$$C_{\tau} = \delta_{x_1,y_1} \| \delta_{x_2,y_2} \| \ldots \| \delta_{x_k,y_k},$$

where $k = |\beta^{int}_{\text{black}}|$ is the number of internal black pixels.

As a result, the length of the container is fixed depending on the type of block:

- For Type 1 blocks: 32 bits.
- For Type 2 blocks: 64 bits.
- For Type 3 blocks: 96 bits.

Thus, the modification $d'_{x,y} = d_{x,y} + \Delta$, where $|\Delta| < 2^{\tau}$, results in a change in the brightness of the pixel by no more than $2^{\tau} - 1$. For $\tau = 3$, this ensures $\Delta I \leq 7$, which lies within the threshold of vision sensitivity. In this case, the theoretical upper limit of the block capacity is determined by the entropy of differences:

$$H_{\max} = \sum_{(x,y)} \log_2\Big(\min\Big(2d_{x,y} + 1,\ 2^{\tau+1} - 1\Big)\Big),$$

Using pairwise differences instead of direct pixel values reduces correlation between the container bits and normalizes the distribution of the lower bits.

3.3. Embedding a Message Using a Linear Hash Function

After forming a differential container C_{τ} with a length of $\tau = 32, 64$ or 96 bits (depending on the type of block τ), the algorithm begins to embed a secret message using a linear hash function implemented in the StegoContainer class [24]. The container is divided into non-overlapping groups of $n = 3$ bits. For each such group, an embedding scheme is applied, in which 2 bits of information are implemented by modifying no more than one bit in the group. The linear hash function is denoted $\lambda(C)$ and is the calculation of the remainder of the division $\lambda(C) = C \bmod g$, where C is in this case the sequence of an empty container and g is a primitive polynomial in the field $\mathrm{GF}(2)$. The primitive polynomial is chosen so that the size of the secret message is one less than the length of the primitive polynomial in binary form. For most practical applications, there are ready-made tables of primitive polynomials [25]. Let s be a secret message; then, to obtain a new embedded sequence c' in which the secret message s is encoded, u needs to be calculated, which is the value of the hash function, and the intermediate variable v:

$$u = C \bmod g;$$

$$v = u \oplus s;$$

The permissible distortions of the container that can be made to secretly transmit the message in the method under consideration are denoted as $\hat{D} = \{s_1, \ldots, s_N\}$, each of which is a set of zero bits and a single bit with a value of 1. A single bit inherently determines the position of the changeable bit of the container in the embedded sequence (analog of the mask). That is, in essence, D is a set of all the permissible modifications of the original sequence in the container that the system can make while remaining invisible.

We find one $d \in \hat{D}$ for which $\lambda_G(d) = v$, that is,

$$d = \lambda^{-1}(v).$$

Then the embedded sequence $c' = d \oplus c$ is written into the container.

Next, the resulting sequence has to be mapped back into the space of brightness pixels. For each black–white pixel pair $(p_{\text{black}}, p_{\text{white}})$, the algorithm calculates the required absolute difference $d'_{x,y}$ in which the τ lower bits coincide with the corresponding substring C_τ. Since the brightness of the white pixel is immutable according to the algorithm, the brightness of the black pixel has to be selected so that:

$$|p_{\text{black}} - p_{\text{white}}| \bmod 2^\tau = d'_{x,y} \bmod 2^\tau.$$

Selection is carried out by searching all integer values in the neighborhood up to ± 20 of the original black pixel value. Among the values that give the required difference bits (the τ lower bits), the one that deviates minimally from the original value is selected:

$$\min_{p'_{\text{black}}} |p'_{\text{black}} - p_{\text{black}}|,$$

under the conditions:

$$0 \le p'_{\text{black}} \le 255,$$

$$|p'_{\text{black}} - p_{\text{white}}| \le 20.$$

If a suitable value is not found in the specified range, then the pixel remains unchanged; however, in the current implementation on a large sample of containers such situations do not occur, and their probability tends to zero. This process is repeated for all blocks in a pseudo-random order specified by the shared secret key. Thus, the embedded information is contained in the absolute differences between the brightness of "black" cells and their canonically matched white neighbors.

3.4. Retrieving the Embedded Sequence

The process of extracting the embedded sequence is fully symmetrical and corresponds to the blind steganographic scheme. Using the shared secret key, the order of processing the blocks is determined. For each image block, the same thresholds t_1 and t_2 as in the embedding step are calculated based on the 20th and 60th percentiles of the distribution of the mean gradient modules calculated from 16×16 blocks. Then each block 8×8 is classified according to the same criteria as in the embedding process. The unambiguity of the extraction process is provided by the staggered order of processing pixels, when all calculations are carried out on immutable "white" cells, and the built-in information is contained in "black" cells (Figure 1).

For each classified block, the procedure for constructing a difference container is repeated, which in turn determines the sequence C' containing embedded information. Three-bit groups of this sequence are entered into the procedure for calculating the hash function, in which each group is divided by a primitive polynomial to obtain the embedded bits of the message. All extraction steps are based solely on the analysis of the stego image and do not require storage or transfer of additional parameters other than the key. An example implementation in accordance with the proposed method is provided at https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 8 January 2026).

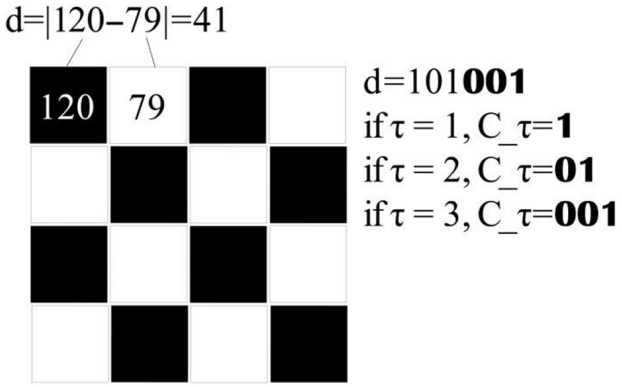


Figure 1. Block 8 × 8, example of forming container C_τ.

4. Results

This algorithm was implemented and tested on a sample of 500 files from a set of 8-bit grayscale images of the dataset [26] database with a size of 512 × 512 pixels.

Adaptive selection of thresholds t_1, t_2 used to classify image blocks allows automatic adjustment of the embedding strategy to the local characteristics of the container. The value t_1 was chosen as the 20th percentile of the ordered sample $\left(\beta(p_{p,q}^{(16)})\right)$ and provided a cut-off of 20% of the smoothest blocks. The value t_2 was chosen as the 60th percentile, which provides a selection of 40% of the most textured blocks that can hide the maximum amount of data with minimal visual distortion. This range allows flexible distribution of the load, avoiding both excessive embedding in smooth areas and underestimating the capacity of textured zones. For experimental verification of the validity of the selected parameters, a sensitivity analysis was carried out. Average capacity and visual quality were used as performance metrics. The experiments were conducted on a subset of 100 images of the database.

As can be seen from Table 1, the proposed configuration (20, 60) provides the best compromise between capacity and quality.

Table 1. Analysis of sensitivity of threshold parameters t_1, t_2.

Percentile	Capacity, bpp	PSNR, DB
5, 45	0.735747	46.8952
10, 50	0.696225	47.3276
15, 55	0.660577	47.7659
20, 60	0.626629	48.2362
25, 65	0.59286	48.7442
30, 70	0.562006	49.2778
35, 75	0.532791	49.8412

A comparison of texture metrics was also conducted to justify the gradient selection. The results of the evaluation of the four alternative metrics are shown in Table 2:

Table 2. Comparison of texture complexity metrics.

Metric	Capacity, bpp	PSNR, dB
Mean absolute gradient	0.626629	48.2362
Brightness dispersion	0.618162	48.3835
RMS difference	0.621156	48.3312
Entropy	0.607898	48.374

The experiment was conducted on a subset of 100 images of the database [26]. The results presented in Table 2 show that the proposed metric provides the best measure of average capacity at comparable visual quality. Alternative metrics show a capacity reduction of 0.9–3.0% without statistically significant improvement in quality.

The capacity of the developed method is expressed as a function of the block type τ, which takes values 1, 2 or 3 depending on the local texture complexity. Each difference $d_{x,y}$ for the "black" pixel holds $\tau - 1$ bits. Thus, the capacity per pixel is:

$$C_{\text{pixel}}(\tau) = \frac{N_{\text{black}} \cdot (\tau - 1)}{N}, \text{ bit}$$

where N is the total number of pixels in a block and N_{black} is the number of internal "black" pixels used for embedding. For an 8×8 block, $N = 64$ and $N_{\text{black}} = 32$:

$$C_{\text{pixel}}(\tau) = \frac{32 \cdot (\tau - 1)}{64} = \frac{\tau - 1}{2}, \text{ bit}$$

The theoretical upper limit of the method capacity is reached at $\tau = 3$. When using the linear hash function method, 2 bits of the message are embedded. Thus,

$$C_{\text{pixel}}(\max) = 1 \text{ bit}.$$

The modification of the pixel is carried out by selecting the value p' in the vicinity of ± 20 from the original p so that τ lower bits of the difference coincide with the target bits. The expected value of the modification $p' - p$ depends on τ:

For $\tau = 1$, the modification is not more than 1 brightness unit.

For $\tau = 2$, the modification is not more than 3 luminance units.

For $\tau = 3$, the variation is not more than 7 luminance units.

MSE upper bound for block type τ:

$$MSE_{\tau} \leq \frac{(2^{\tau} - 1)^2}{12}.$$

For a mixed image with slices of blocks f_1, f_2, f_3, where $f_1 + f_2 + f_3 \leq 1$:

$$MSE_{\text{total}} \leq \sum_{\tau=1}^{3} f_{\tau} \cdot \frac{(2^{\tau} - 1)^2}{12}.$$

For uniform type distribution $f_1 \approx f_2 \approx f_3 \approx 0.3$:

$$MSE_{\text{total}} \leq 0.3 \cdot \left(\frac{0}{12} + \frac{1}{12} + \frac{36}{12} \right) \approx 0.92,$$

which corresponds to $PSNR \geq 10 \cdot \log_{10}\left(\frac{255^2}{0.92}\right) \approx 48.3$ dB.

The average practical capacity, as well as the minimum and maximum actual capacity, was calculated experimentally by embedding the maximum possible number of bits according to the algorithm for the given containers and the block size of 8×8 pixels. The results of this study are shown in Table 3.

Table 3. Quality indicators for 500 BOSS containers.

Values for the Entire Dataset	Minimum	Average	Maximum
Capacity, bpp	0.502602	0.626558	0.692841
PSNR, dB	47.3713	48.2392	50.5584
MSE	0.571793	0.982471	1.1911

The maximum capacity in the given dataset was ≈0.7 bpp and was achieved for this container, represented in Figure 2, with the actual calculation of $t_1 = 19.6567$, $t_2 = 38.062$ and the obtained values PSNR = 47.4305, MSE = 1.17498.

(**a**) (**b**)

Figure 2. Cover (**a**) and stego (**b**) at this value, 0.7 bpp, for boss base.

The minimal capacity in the given dataset was ≈0.5 bpp and was obtained for this container, shown in Figure 3, with the actual calculated values $t_1 = 0.307087$, $t_2 = 1.0812$, PSNR = 48.4925, and MSE = 0.920082.

(**a**) (**b**)

Figure 3. Cover (**a**) and stego (**b**) at this value, 0.5 bpp, for boss base.

The obtained data confirm the peculiarity of the application of this algorithm for noisy images, which achieve maximum capacity. Images with multiple homogeneous blocks classified as Type 4, in contrast, have minimum actual capacity.

Figure 4 below shows a graph of the implementation capacity versus PSNR for 500 test images. The color scale reflects the value of the RMS Error (MSE) so that darker dots

correspond to less distortion. As can be seen from the graph, there is an inverse correlation between the capacity and the quality of the stegimage: as bpp increases, the PSNR values decrease. But even at maximum capacity, the PSNR value remains above 47 dB, and the MSE does not exceed 1.2, which indicates the high quality of the received stegcontainers.

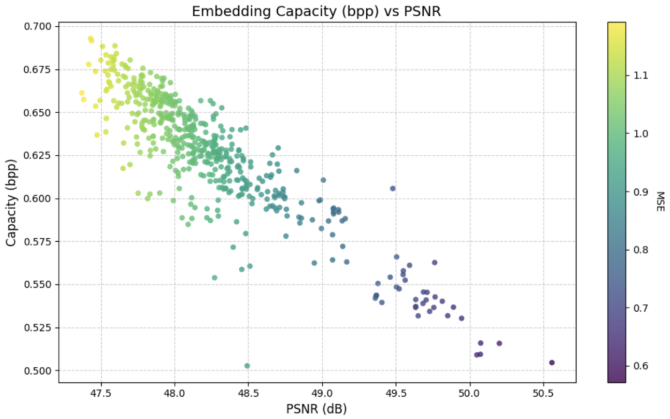


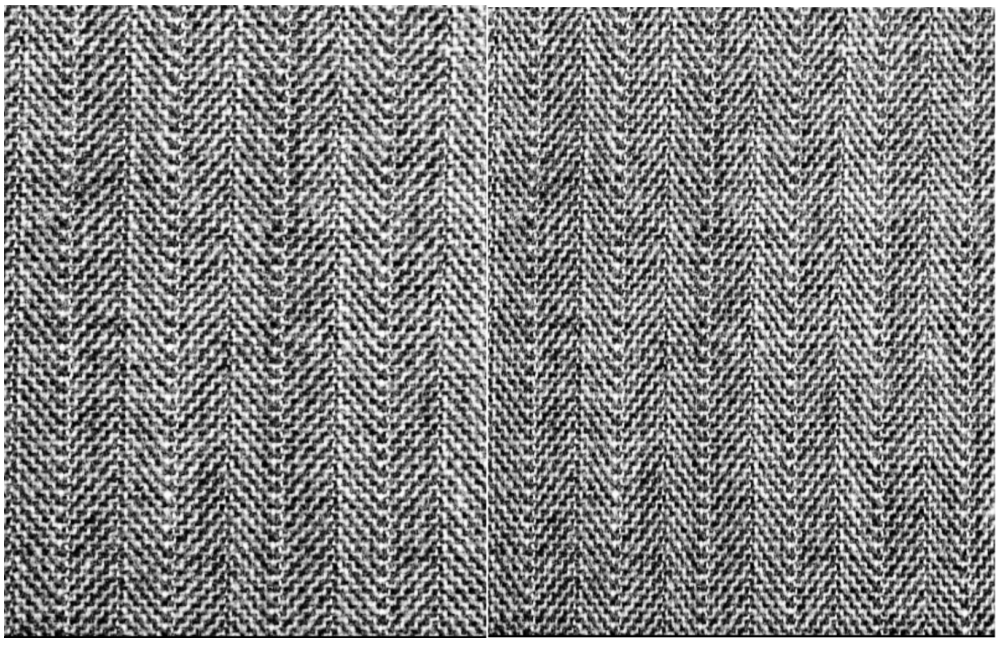
Figure 4. Capacity (bpp) versus PSNR, taking into account MSE for boss base.

Further, similar experiments were conducted for 38 image files of various textures from the SIPI database set [27]. The results of this study are shown in Table 4.

Table 4. Quality indicators for 38 SIPI base containers.

Values for the Entire Dataset	Minimum	Average	Maximum
Capacity, bpp	0.0141602	0.583689	0.66246
PSNR, dB	47.1735	48.6694	62.7363
MSE	0.0346298	0.946319	1.24662

The maximum capacity in the given dataset was ≈0.7 bpp and was achieved for this container, shown in Figure 5, with the actual calculated values $t_1 = 79.8171$, $t_2 = 85.9449$, PSNR = 47.1735, and MSE = 1.24662.



(**a**) (**b**)

Figure 5. Cover (**a**) and stego (**b**) at this value, 0.7 bpp, for SIPI base.

The minimal capacity in the given dataset was ≈0.01 bpp and was obtained for this container, shown in Figure 6, with $t_1 = 0$, $t_2 = 0$, PSNR = 62.7363, and MSE = 0.0346298.

(**a**) (**b**)

Figure 6. Cover (**a**) and stego (**b**) at this value, 0.01 bpp, for sipi base.

The data obtained once again confirm that the algorithm adapts depending on the container so that embedding occurs in the most secure areas of the container. The set of containers with texture images contained predominantly noisy images, but there was also a very smooth image in the set, which clearly demonstrated the case of limiting embedding into smooth areas of the image. Figure 7, similar to Figure 4, shows a graph of the ratio of the implementation capacity to PSNR for 38 images of different textures.

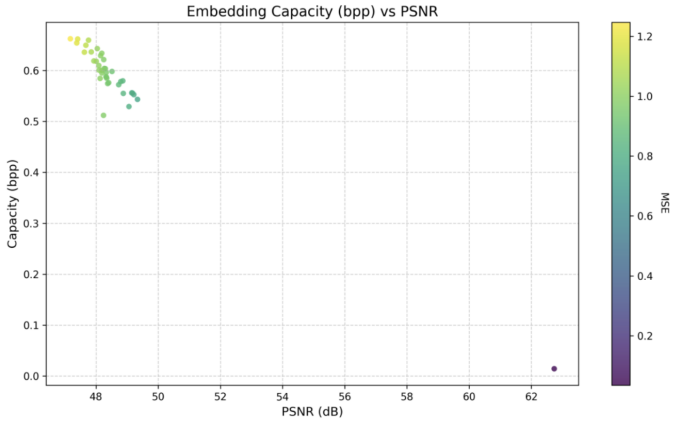


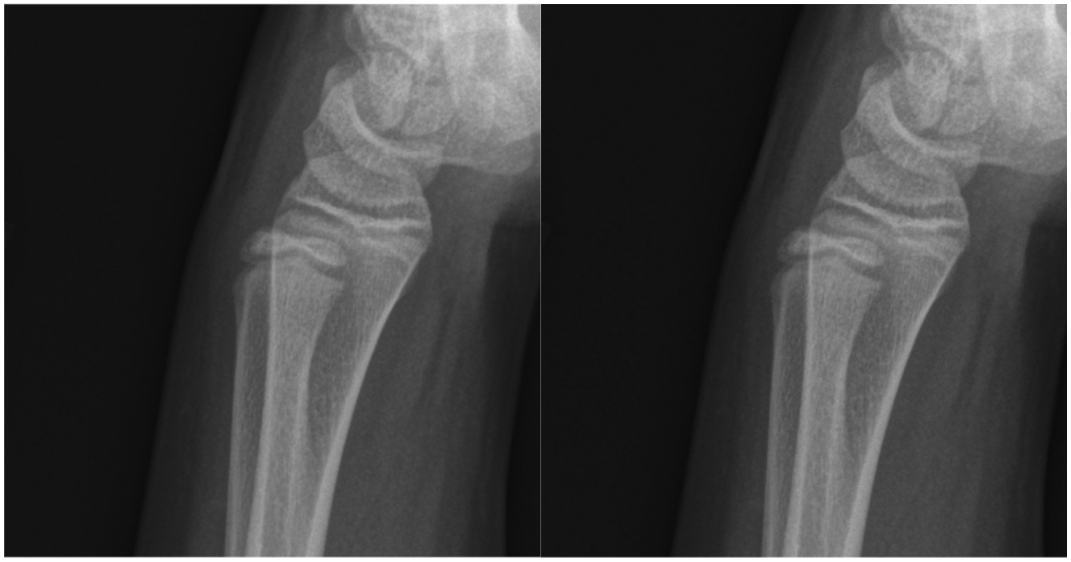
Figure 7. Capacity (bpp) versus PSNR, taking into account MSE for SIPI base.

In order to evaluate the algorithm's performance in relation to medical images, experiments were also conducted for 275 files from the Kaggle database [28,29] containing images of patient examinations. The results of this study are shown in Table 5.

Table 5. Quality indicators for 275 Kaggle containers.

Values for the Entire Dataset	Minimum	Average	Maximum
Capacity, bpp	0.0587463	0.547763	0.672256
PSNR, dB	47.6012	48.7473	58.0883
MSE	0.100983	0.877752	1.12968

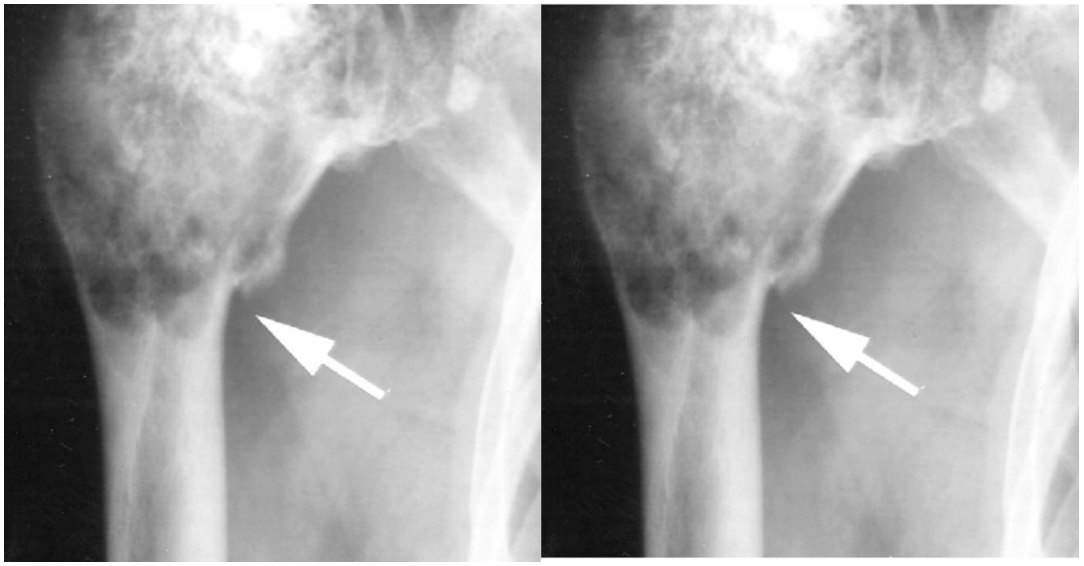
The maximum capacity in the given dataset was ≈0.7 bpp and was achieved for this container, shown in Figure 8, with the actual calculated values $t_1 = 0.532111$, $t_2 = 5.07837$, PSNR = 47.637, and MSE = 1.12041.



(**a**) (**b**)

Figure 8. Cover (**a**) and stego (**b**) at this value, 0.7 bpp, for Kaggle base.

The minimal capacity in the given dataset was ≈0.06 bpp and was obtained for this container, shown in Figure 9, with the actual calculated values $t_1 = 3.32099$, $t_2 = 7.62008$, PSNR = 58.0883, and MSE = 0.100983.



(**a**) (**b**)

Figure 9. Cover (**a**) and stego (**b**) at this value, 0.06 bpp, for Kaggle base.

From the results obtained, it follows that there is a significant variation in capacity on different datasets—the maximum capacity is achieved on textured or noise-like images, and the minimum capacity is observed on smooth, mostly homogeneous images. Figure 10, similar to Figure 4, shows a graph of the implementation capacity versus PSNR for 275 medical images.

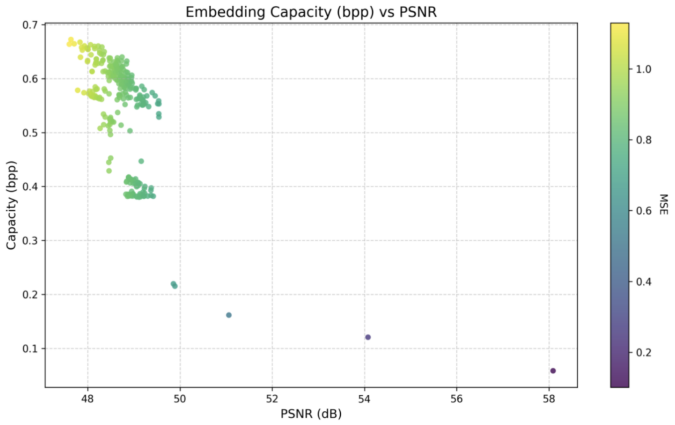


Figure 10. Capacity (bpp) versus PSNR, taking into account MSE for Kaggle base.

Thus, the proposed adaptive steganographic method demonstrates high sensitivity to the local structure of the image. The use of a linear hash function in this implementation allows to introduce the optimal number of bits per pixel, while the image quality remains high. At maximum capacity, distortion is visually invisible, as the algorithm maximizes capacity where it is safe.

The computational complexity of the proposed steganographic algorithm has the order $\mathcal{O}(N \times \log N)$, where N represents the total number of pixels in the image. This quasilinear complexity is due to the need for adaptive classification of texture regions of the image, requiring sorting of local brightness gradients. In this case, the percentile sorting operation is the dominant factor in the overall algorithm and has complexity $\mathcal{O}(K \times \log K)$, where $K = N/256$ corresponds to the number of 16×16 blocks used to calculate the gradient distribution. The 8×8 block classification has complexity $\mathcal{O}(B \times P^2)$, where $B = N/64$ is the number of blocks and $P \approx 32$ is the number of pixels analyzed in staggered order. Operations in finite Galois fields have complexity $\mathcal{O}(G \times M^2)$, where G is the number of groups in the container and $M \leq 9$ is the degree of a primitive polynomial. Implementation and data extraction use arithmetic in $\mathrm{GF}(2^M)$ through precalculated lambda tables. Pixel modification has complexity $\mathcal{O}(B \times C \times R)$, where $C \approx 32$ is the number of modified pixels in the block and $R = 41$ is the search range for the optimal value (± 20 of the original brightness). For a typical 512×512 pixel image, the algorithm performs approximately 2.6×10^7 operations, which corresponds to an execution time of 26–260 ms on modern processors at a clock rate of 1–10 operations per clock.

For experimental verification of theoretical estimates of the computational complexity of the proposed method, the execution time of the algorithm on a platform with an Intel® Core™ i5-9600KF processor (3.70 GHz), 32 GB RAM, SSD NVMe, in QT 6.4.2 environment (MinGW compiler 11.2.0, build mode Release) was measured. For 512×512 image processing, the average run time was 243.06 ms (median 245 ms, range 209–274 ms) per image.

Compared to machine learning methods (neural network approaches with $\mathcal{O}(N \times L)$, where L is the number of layers), the algorithm demonstrates predictable execution time without dependence on iterative optimization procedures.

To analyze the resistance to different types of attacks, such as pruning, compression, filtering or adding noise of the proposed method, an experiment was conducted to extract

embedded messages from stegcontainers subjected to different types of attacks. The metric used was the percentage of bits that were correctly retrieved relative to the original message. The results showed that any impact that changes the pixel values or geometric transformations destroys the difference container, making extraction impossible. Thus, the method is intended for trusted transmission channels where the stegcontainer is not subjected to additional processing.

To quantify the similarity of cover and stego images, the SSIM and PCC metrics were additionally computed, as well as RS analysis, pixel brightness histogram estimation, χ^2-square statistical analysis, JS-divergence, and difference histogram estimation for sampling containers from each of the datasets under study and mentioned above. The results of this study are presented in Table 6.

Table 6. Examining containers from BOSS database.

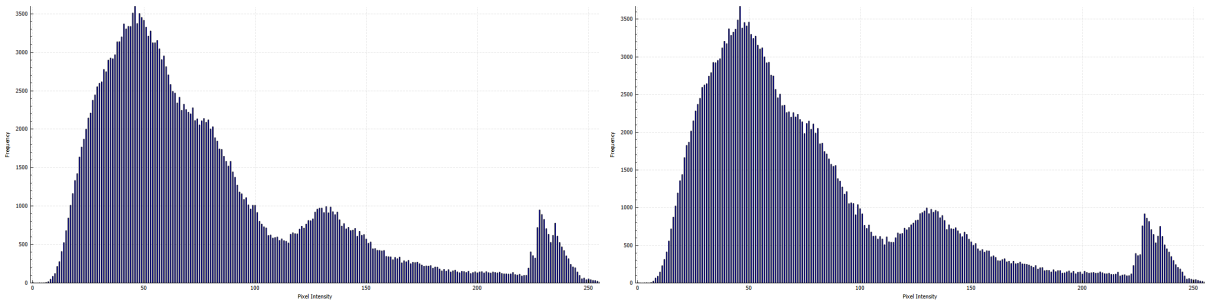
Containers	Embedded, Bytes	RS Detection, Bytes	SSIM	PCC	PSNR
Cover (1)		410	0.9998	0.9998	48.2101
Stego (1)	20,717	227			
Cover (2)		33	0.9996	0.9996	48.4743
Stego (2)	20,008	117			
Cover (3)		1347	0.9998	0.9998	48.0068
Stego (3)	20,188	1027			
Cover (4)		165	0.9998	0.9998	48.5070
Stego (4)	19,698	502			
Cover (5)		123	0.9999	0.9999	48.6797
Stego (5)	19,672	228			
Cover (6)		446	0.9998	0.9998	47.9849
Stego (6)	21,424	617			
Cover (7)		568	0.9999	0.9999	48.3136
Stego (7)	21,031	651			
Cover (8)		315	0.9997	0.9997	48.3736
Stego (8)	20,170	325			
Cover (9)		98	0.9996	0.9996	48.3424
Stego (9)	20,449	76			
Cover (10)		39	0.9998	0.9998	48.0076
Stego (10)	21,493	985			
Cover (11)		306	0.9998	0.9998	48.0949
Stego (11)	20,972	367			
Cover (12)		100	0.9996	0.9996	48.4667
Stego (12)	20,373	26			
Cover (13)		168	0.9998	0.9998	48.6966
Stego (13)	20,309	263			
Cover (14)		188	0.9999	0.9999	48.2413
Stego (14)	19,983	202			
Cover (15)		360	0.9996	0.9996	48.9881
Stego (15)	19,628	18			
Cover (16)		293	0.9995	0.9995	48.2653
Stego (16)	20,525	73			
Cover (17)		394	0.9999	0.9999	48.2919
Stego (17)	19,429	644			
Cover (18)		473	0.9999	0.9999	48.6233
Stego (18)	20,090	363			
Cover (19)		425	0.9998	0.9998	48.2629
Stego (19)	20,558	204			

Table 6. *Cont.*

Containers	Embedded, Bytes	RS Detection, Bytes	SSIM	PCC	PSNR
Cover (20)		1041	0.9998	0.9998	48.8433
Stego (20)	19,301	115			

Based on the results presented in Table 6, the high and nearly equal SSIM and PCC values indicate that the changes are locally balanced and do not violate either the linear relationship between the pixels or the perceived image structure. Estimates of the volume of hidden data for stegimages are in the same range as for the original cover images, and at the same time are greatly underestimated relative to the actual volume of embedded data. This indicates that the method does not create statistical anomalies in the container.

Figure 11 shows an example of a pixel intensity histogram for an empty and full container from the boss kit.



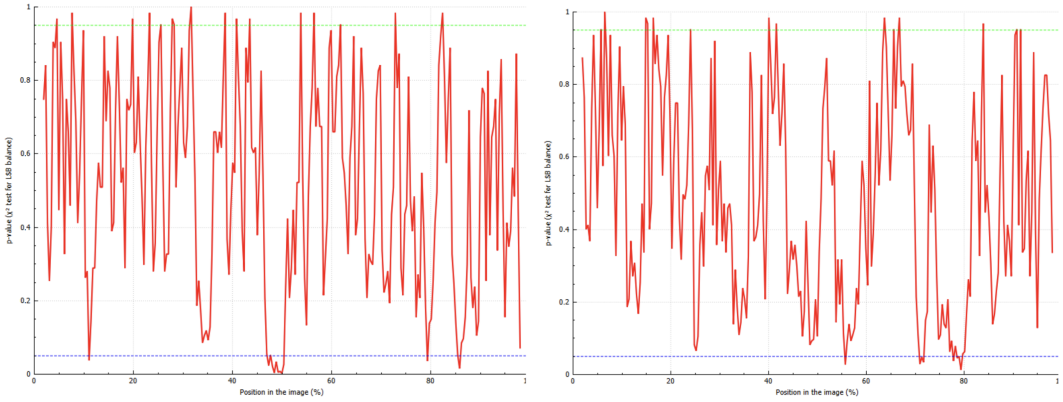
(**a**) (**b**)

Figure 11. Image histogram for the boss base test image to showcase the similarity of cover images (**a**) and stego images (**b**).

As can be seen from Figure 11, the pixel intensity distribution histograms for the original (cover) and steganographic (stego) images are almost identical. This demonstrates that the presented embedding method does not introduce noticeable changes to the global brightness statistics. Full results of the image experiments (Figure 11) are available at https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).

To compare the observed distribution of the lower significant bits (LSBs) of pixels in an image with the theoretically expected uniform distribution characteristic of natural images, we used the visual χ^2-square test. In this analysis, the results of which are shown in Figure 11, a sliding window of 1000 pixels in increments of 1000 pixels was used. For each position, the p-value of the χ^2 test was calculated, reflecting the degree of agreement of the observed LSB distribution with the natural image hypothesis. For comparison, graphs for the original cover images and images filled with our stego method are presented (the actual capacity of each container in bytes is shown in Table 6). The abscissa axis displays the relative position of the center of the sliding window, expressed as a percentage of the total length of the image. The ordinate axis represents the p-value of the χ^2 test, calculated for the distribution of the LSB pixels within the current window when testing the zero hypothesis about the uniform distribution of even and odd intensities (expected ratio 1:1). The green dotted line in the graph shows the threshold $p = 0.95$. Values above this threshold indicate an abnormally high consistency with the uniform distribution, which is a typical sign of replacing the lower bits. Thus, sections of the graph above 0.95 are interpreted as suspicious areas potentially containing a steganographic message. The blue dotted line shows the threshold $p = 0.05$. Values below 0.05 indicate a strong imbalance in the LSB, which is typical of textures or boundaries in natural images.

The conducted steganalysis shows that the developed method does not increase the probability of detecting a stego embedding in comparison with an empty container, which indicates insignificant statistical distortions or their absence for the implementation method using the linear hash function described in this paper. In Figure 12, the green ($p = 0.95$) and blue ($p = 0.05$) dashed lines denote thresholds for detecting anomalously uniform LSB behavior and strong LSB imbalance, respectively. The results for the remaining images (Figure 12) are available at: https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).



(**a**) (**b**)

Figure 12. For the boss base test image χ^2 (cover images (**a**) and stego images (**b**)).

Figure 13 shows several covers of images from the BOSS base set that are enrolled in the study number for clarity.

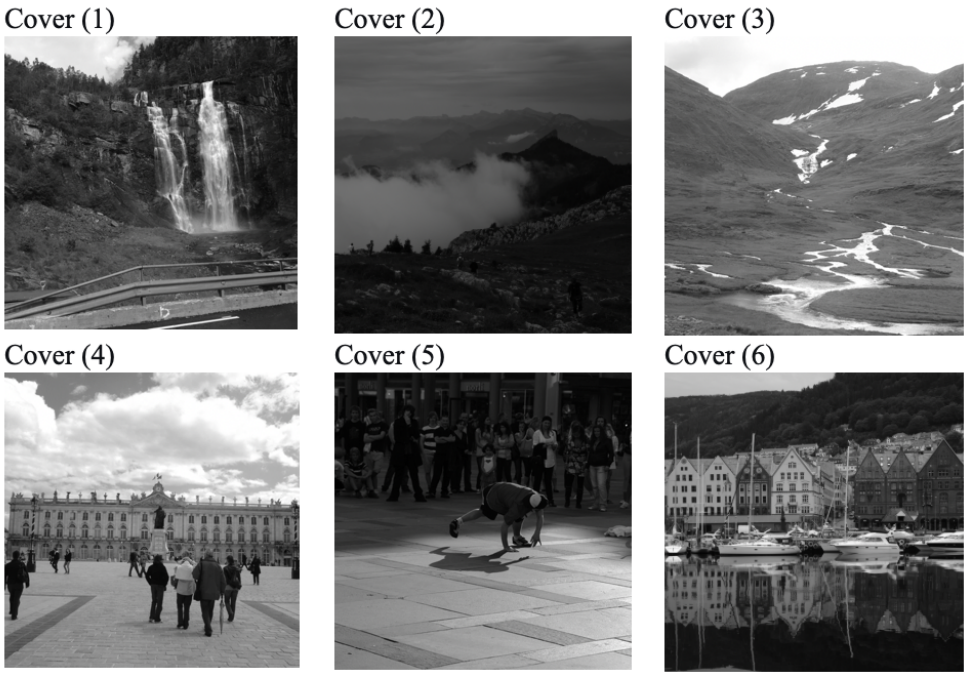


Figure 13. Cover test images of the boss base.

The following are the results of a similar study for the next SIPI dataset.

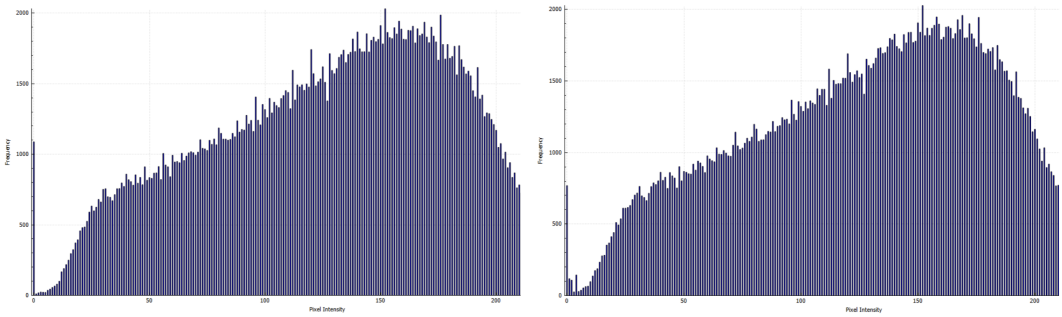
The results from Table 7 show that the trend of values has not changed from Table 6, except for the characteristics of textured containers, where there is instability in the RS analysis estimates. On a number of images with high and regular texture activity, the RS

detector gives abnormally high values for cover images. This indicates that such textures themselves contain statistical patterns that mimic the features of LSB implementation. Figure 14 shows pixel intensity histograms for several empty and full SIPI set containers corresponding to Table 7.

Table 7. Examining containers from the SIPI database.

Containers	Embedded, Bytes	RS Detection, Bytes	SSIM	PCC	PSNR
Cover (1)		1208	0.9998	0.9998	48.2989
Stego (1)	19,785	4370			
Cover (2)		0	0.9998	0.9998	48.5128
Stego (2)	19,603	314			
Cover (3)		1699	0.9997	0.9997	47.8452
Stego (3)	20,860	919			
Cover (4)		1740	0.9994	0.9994	47.4086
Stego (4)	21,691	2507			
Cover (5)		1956	0.9995	0.9994	48.7279
Stego (5)	18,751	507			
Cover (6)		4087	0.9997	0.9997	48.3188
Stego (6)	19,260	3892			
Cover (7)		1072	0.9994	0.9994	48.0906
Stego (7)	19,983	470			
Cover (8)		630	0.9982	0.9980	47.7580
Stego (8)	21,619	1389			
Cover (9)		133	0.9993	0.9993	47.3783
Stego (9)	21,440	948			
Cover (10)		1080	0.9995	0.9995	48.3273
Stego (10)	19,543	632			
Cover (11)		1205	0.9998	0.9998	48.8760
Stego (11)	18,191	1239			
Cover (12)		1250	0.9996	0.9996	48.1851
Stego (12)	20,778	341			
Cover (13)		243	0.9998	0.9998	49.2146
Stego (13)	18,113	884			
Cover (14)		3989	0.9999	0.9999	48.1874
Stego (14)	19,507	3574			
Cover (15)		850	0.9999	0.9999	49.1487
Stego (15)	18,226	3751			
Cover (16)		39,856	0.9999	0.9999	48.0080
Stego (16)	20,258	33,440			
Cover (17)		12,080	0.9999	0.9999	47.1735
Stego (17)	21,707	11,288			
Cover (18)		15,515	0.9999	0.9999	48.7941
Stego (18)	18,949	16,134			
Cover (19)		15,084	0.9999	0.9999	48.1301
Stego (19)	19,155	17,765			
Cover (20)		9558	0.9999	0.9999	48.3723
Stego (20)	18,825	11,955			

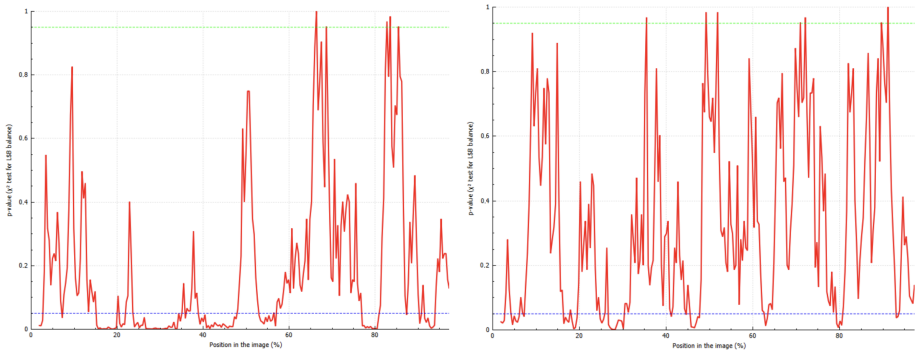
The histograms in Figure 14 demonstrate that the proposed adaptive method does not introduce statistically significant changes to the global brightness distribution, even for textured containers. Additional image results can be found (Figure 14) at: https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).



(**a**) (**b**)

Figure 14. Image histograms for the SIPI base test image to showcase the similarity of cover images (**a**) and stego images (**b**).

Next, the results of the χ^2-square test for the corresponding test set are shown in Figure 15, where the green ($p = 0.95$) and blue ($p = 0.05$) dashed lines denote thresholds indicating anomalously high conformity to a uniform distribution and strong LSB imbalance, respectively.



(**a**) (**b**)

Figure 15. (**a**,**b**) Research method via χ^2-square for SIPI base test images.

The results of the analysis show, similar to the previous data, that the studied method does not introduce significant statistical distortions in the container. The results of the experiments performed on the remaining images are available at: https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).

Figure 16 shows some of the SIPI base images included in the study for clarity.

The following are the results of a similar study for the next Kaggle dataset (Table 8).

Table 8. Examining containers from the Kaggle database.

Containers	Embedded, Bytes	RS Detection, Bytes	SSIM	PCC	PSNR
Cover (1)		295	0.9998	0.9998	48.3412
Stego (1)	17,500	245			
Cover (2)		109	0.9998	0.9999	48.5353
Stego (2)	20,416	130			
Cover (3)		262	0.9998	0.9998	48.7026
Stego (3)	19,872	260			
Cover (4)		106	0.9997	0.9997	48.8763
Stego (4)	19,854	14			
Cover (5)		8	0.9999	0.9999	48.4894
Stego (5)	20,897	121			

Table 8. *Cont.*

Containers	Embedded, Bytes	RS Detection, Bytes	SSIM	PCC	PSNR
Cover (6)		132	0.9995	0.9995	48.5076
Stego (6)	19,775	107			
Cover (7)		88	0.9997	0.9997	47.9293
Stego (7)	21,641	84			
Cover (8)		125	0.9999	0.9999	48.9166
Stego (8)	16,483	125			
Cover (9)		302	0.9991	0.9991	48.5289
Stego (9)	20,393	418			
Cover (10)		82	0.9998	0.9998	48.8194
Stego (10)	19,606	43			
Cover (11)		82	0.9998	0.9998	48.8194
Stego (11)	19,606	43			
Cover (12)		282	0.9999	0.9999	49.1604
Stego (12)	14,641	404			
Cover (13)		409	0.9998	0.9998	48.6079
Stego (13)	20,481	475			
Cover (14)		198	0.9998	0.9998	48.7305
Stego (14)	19,312	440			
Cover (15)		461	0.9999	0.9999	48.1776
Stego (15)	20,595	865			
Cover (16)		197	0.9998	0.9998	48.6119
Stego (16)	20,582	94			
Cover (17)		1258	0.9999	0.9999	49.1553
Stego (17)	18,115	1079			
Cover (18)		281	0.9998	0.9998	48.6219
Stego (18)	20,472	105			
Cover (19)		197	0.9998	0.9998	48.7305
Stego (19)	19,312	440			
Cover (20)		25	0.9995	0.9995	48.2285
Stego (20)	21,630	196			

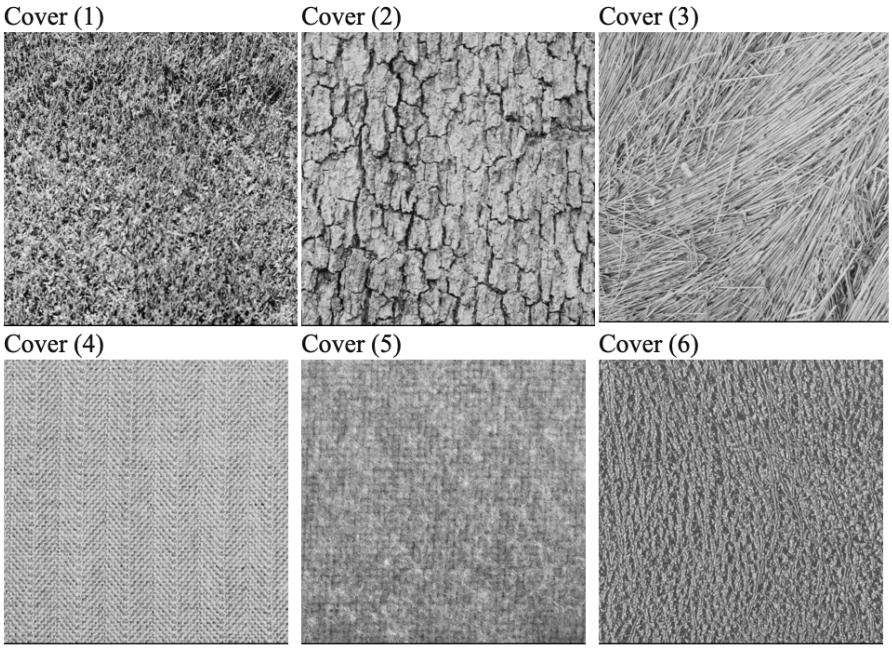
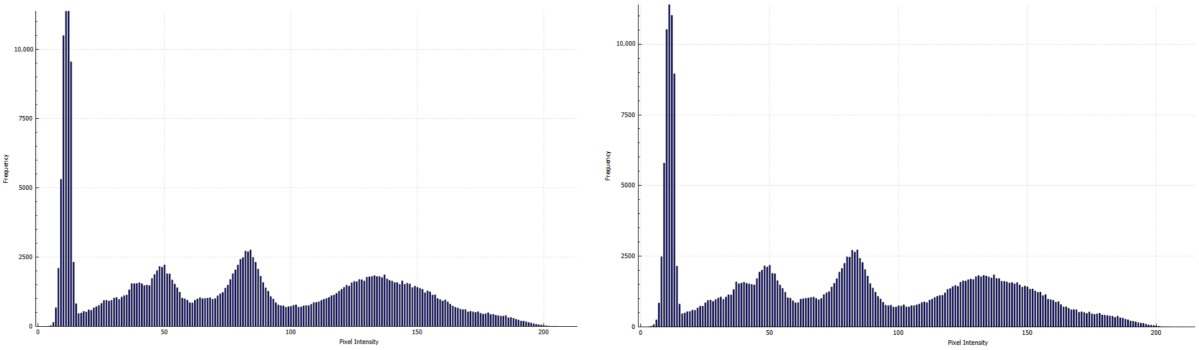


Figure 16. Cover test images of the sipi base.

The results from Table 8 show that the trend of results has not changed from Table 6. Figure 17 shows pixel intensity histograms for an example of an empty and full Kaggle container corresponding to Table 8.

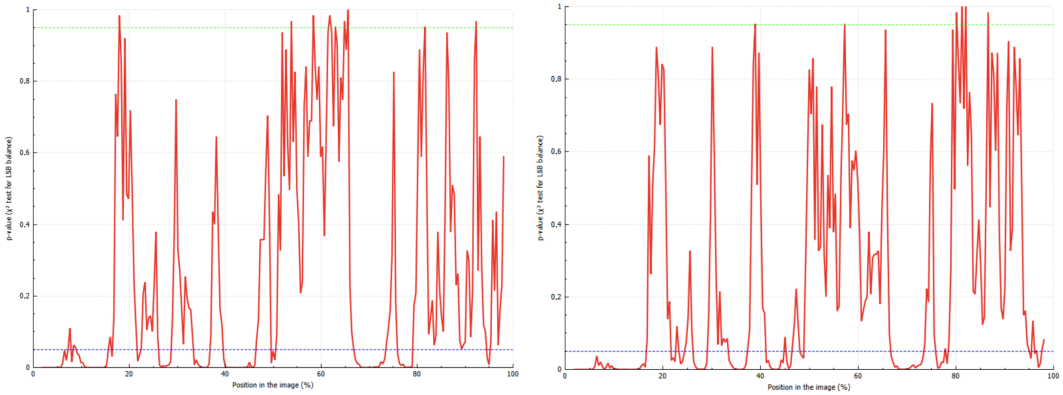


(**a**) (**b**)

Figure 17. Image histograms for the kaggle base test image to showcase the similarity of cover images (**a**) and stego images (**b**).

The histograms in Figure 17 demonstrate that the proposed adaptive method does not introduce statistically significant changes to the global brightness distribution for medical images. All additional results of the experiment on the images (Figure 17) are available at: https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).

Next, the results of the χ^2-square test for the corresponding test set are shown in Figure 18. The results for the remaining cover image data are available at: https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).



(**a**) (**b**)

Figure 18. For the Kaggle base test image χ^2 (cover images (**a**) and stego images (**b**)).

The results of the analysis show, similar to previous data, that the study method does not introduce significant statistical distortions in the container, including in the case of medical images.

Figure 19 shows some of the Kaggle base cover images included in the study number for clarity.

To quantify the visual and statistical indistinguishability of the stegimages from the original (cover), the Jensen–Shannon distance (JS divergence) between the pixel intensity histograms for containers from three test sets was calculated. The results are presented in Figures 20, 21 and 22, respectively.

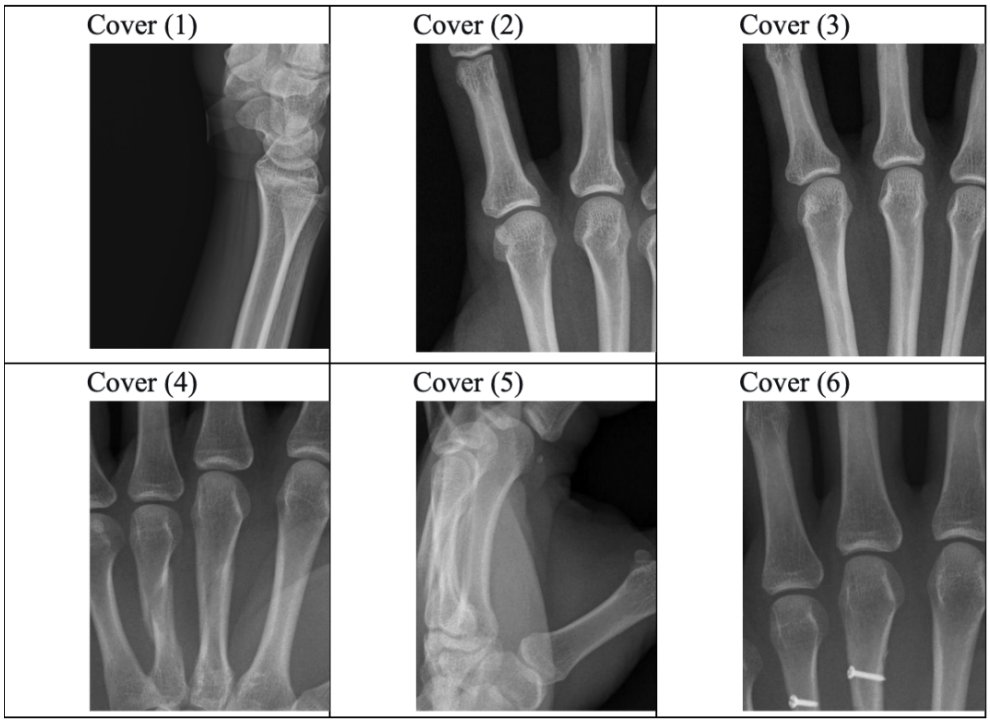


Figure 19. Cover test images of the SIPI base.

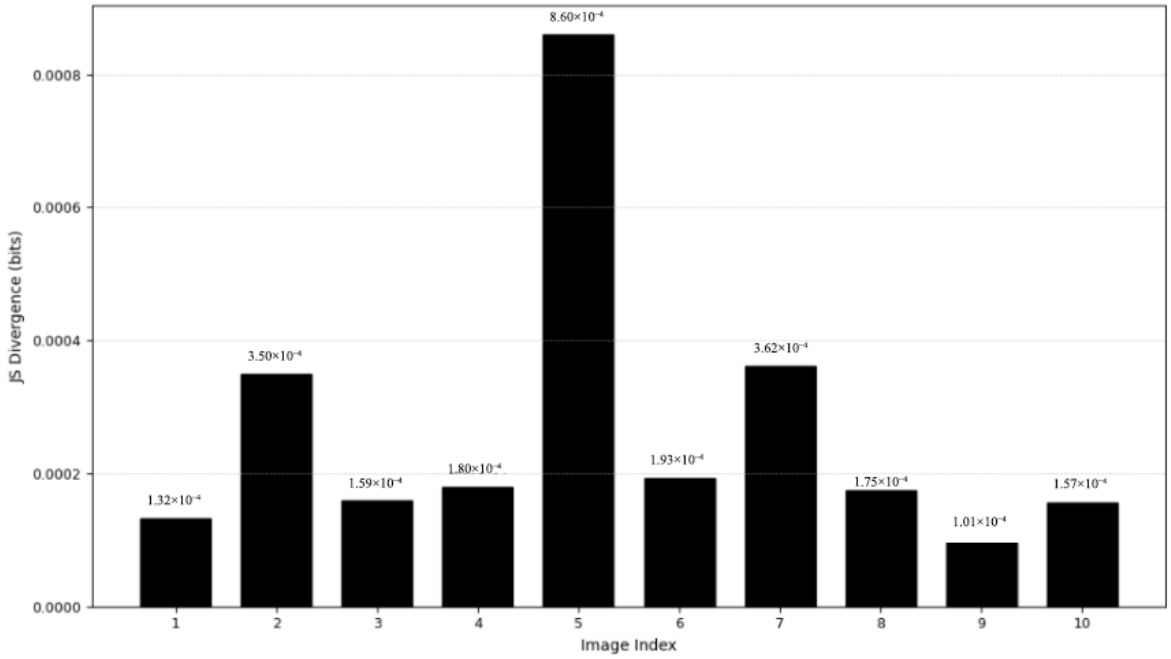


Figure 20. The Jensen–Shannon divergence between the pixel intensity histograms of cover and stego images for 10 base test samples.

The results of the JS-divergence analysis between the pixel intensity histograms for 10 image pairs (cover and stego) from the our dataset [26] show that all values are in the range from 9.97×10^{-5} to 8.60×10^{-4} bits. This indicates a high degree of indistinguishability of stego images from the originals at the level of global brightness distribution. The maximum value is observed for image number 5, but even it remains extremely small, which confirms the effectiveness of the method used.

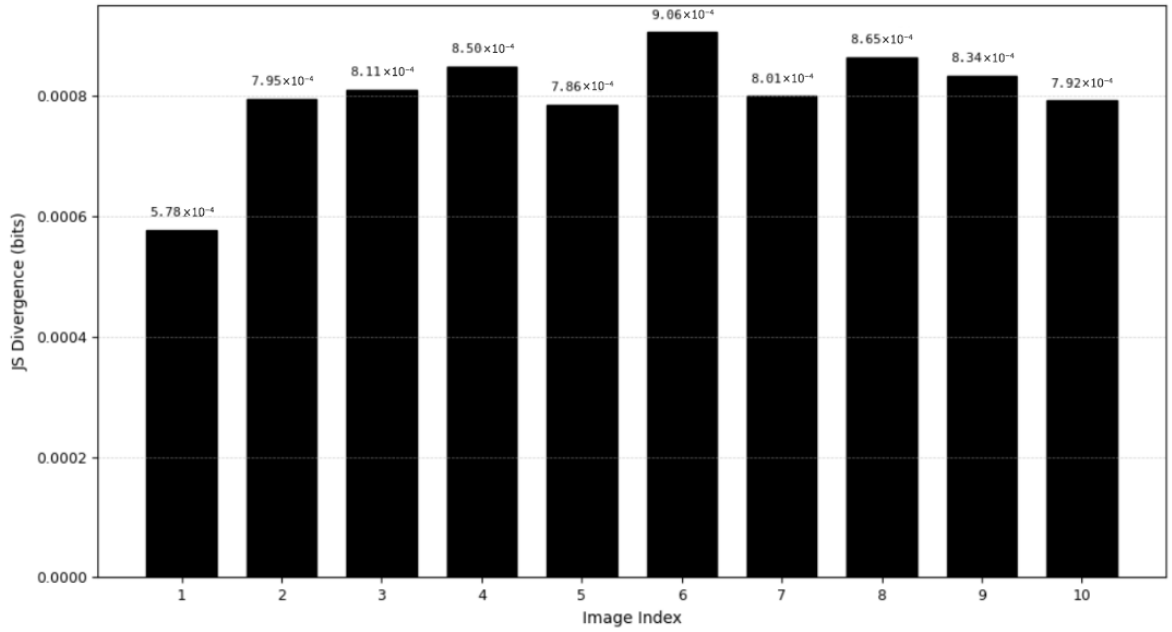


Figure 21. The Jensen–Shannon divergence between the pixel intensity histograms of cover and stego images for 10 test samples of SIPI base.

For the SIPI texture set, the average JS divergence was 7.82×10^{-4} bits, slightly higher than for our dataset [26], which may be due to higher texture uniformity. However, all values remain extremely small.

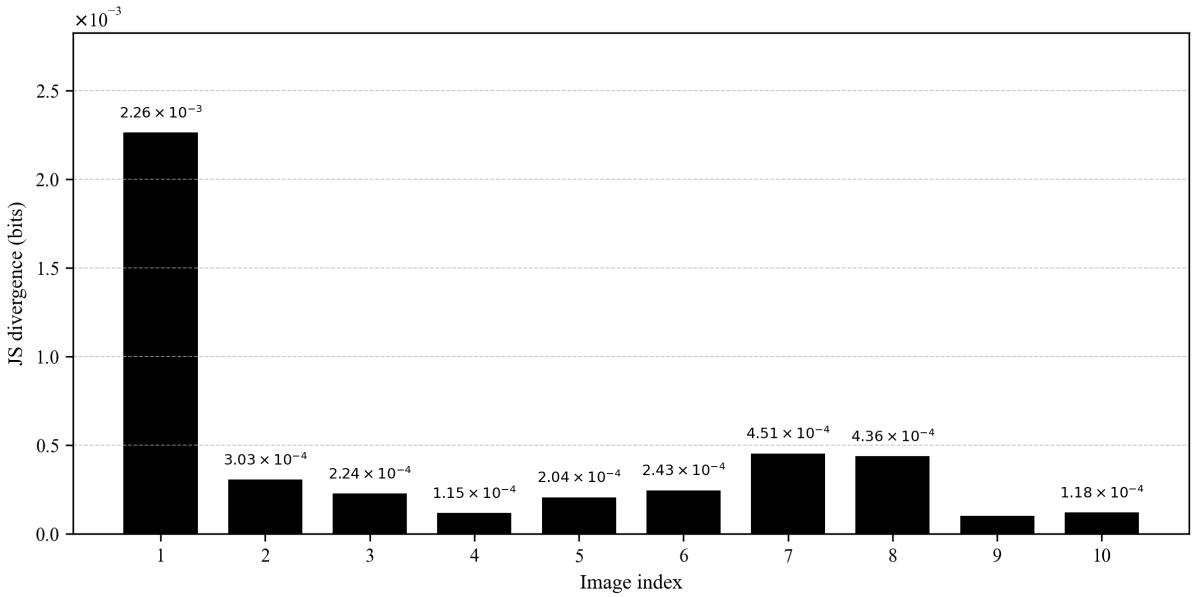
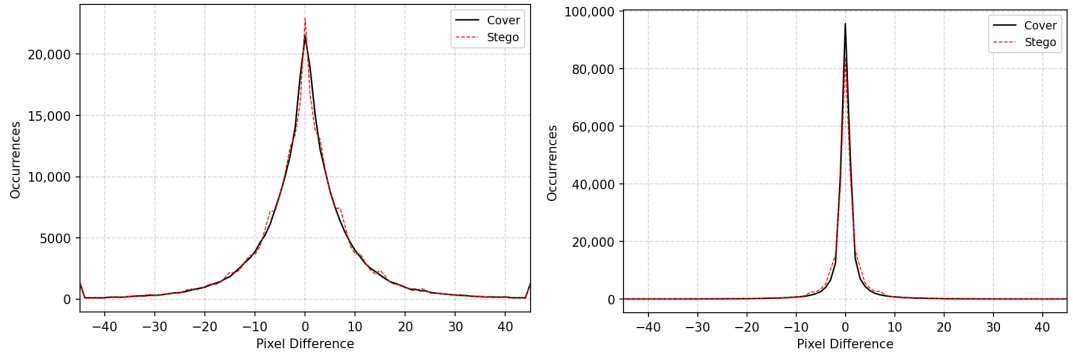


Figure 22. The Jensen–Shannon divergence between the pixel intensity histograms of cover and stego images for 10 test samples Kaggle base.

For medical images, the JS divergence varies from 1.18×10^{-4} to 2.26×10^{-3} bits. Despite one anomalous case, the average value remains low, which confirms the applicability of the method, taking into account the specifics of medical content.

Figure 23 shows pixel difference histograms for two examples of cover images and their corresponding stego images from the dataset [26] corresponding to Table 6.

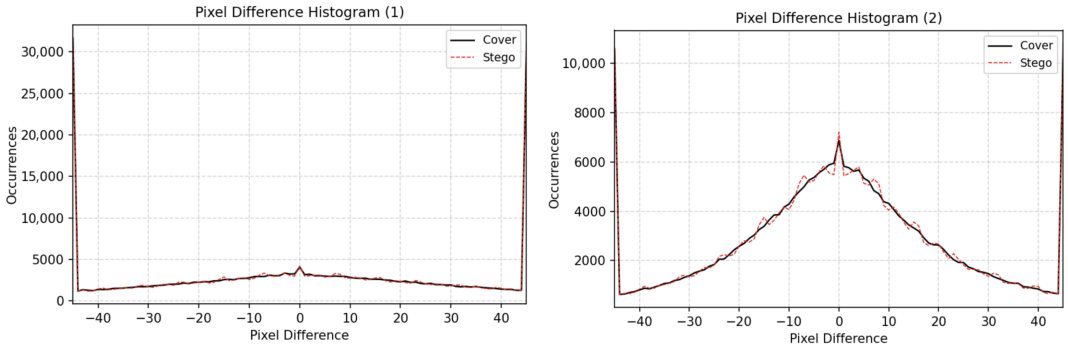


(**a**) (**b**)

Figure 23. Histogram differences between covers (**a**) and theirs corresponding stegos (**b**) for 2 base test images.

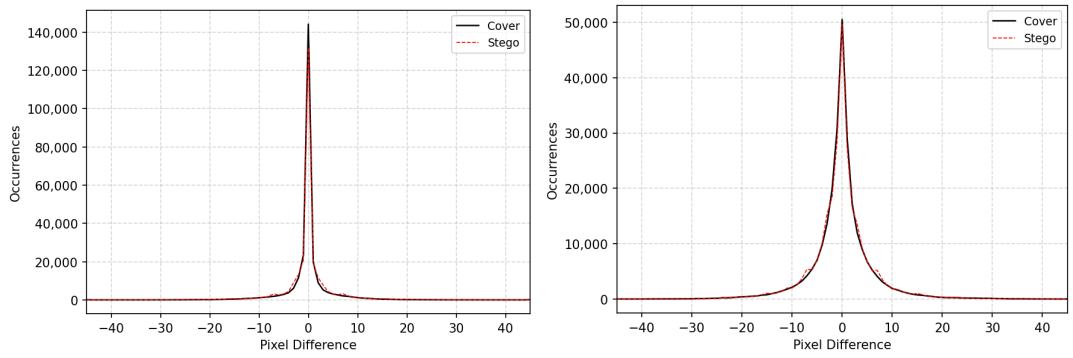
The graphs in Figure 24 show that the histograms are well preserved after the data are hidden according to the proposed adaptive scheme, despite the capacity of up to 0.69 bpp.

Figures 24 and 25 show pixel difference histograms for several cover images and their respective stegous from SIPI and Kaggle sets corresponding to Tables 7 and 8, respectively.



(**a**) (**b**)

Figure 24. Histogram differences between covers (**a**) and theirs corresponding stegos (**b**) for 2 SIPI base test images.



(**a**) (**b**)

Figure 25. Histogram differences between covers (**a**) and theirs corresponding stegos (**b**) for 2 Kaggle base test images.

The peculiarity of the graphs in Figure 25 is that the SIPI set in our case contained images of textures, which are characterized by many sharp transitions. The general form of the graphs for cover and stegois is preserved, but there are small local deviations in places. This indicates that the proposed method does not introduce pronounced statistical artifacts

in the histogram of pixel differences, even on textured images. The results of the graphs in Figure 25 are similar to those of Figure 23 and also show no significant deviations in the histograms.

For an objective assessment of the stability of the proposed method, a modern stegoanalysis based on spatial rich models (SRMs) [30] was applied, implemented in the form of console utility srm.exe (version from 2013, DDE Lab, Laboratory in Durgapur, India). The SRM method extracts a complete set of 34,671 characters per image. The obtained features were used as input data for the ensemble classifier (Ensemble Classifier, version 2.0) [31]. The results showed a detection accuracy of 100%, which indicates the high vulnerability of the current implementation to modern stegoanalysis. When the capacity in our proposed method was reduced to 0.1 bpp, this stegoanalysis showed 98.5% detection. However, this result is valuable because it involves the integration of more complex adaptive strategies into an existing architecture, where embedding is controlled by a deterministic linear hash function.

Comparative Analysis

In contrast to reversible methods such as PVO + PEE [4], adaptive multi-pass embedding [5], PVO with optimized thresholds L, H [6] and adaptive histogram set modification (AMHM) strategy [7], the proposed approach solves the problem of irreversible blind steganography. All the listed RDH methods are focused on restoring the original image. Methods [4,5] reach PSNR above 55 dB with a capacity of up to 0.076 bpp, and method [7]—up to 0.19 bpp. Method [6] in the presented experiments demonstrates PSNR in the range from 37 to 56 dB, depending on the chosen embedding mode. Our method does not require the restoration of the original, provides a capacity of up to 0.69 bpp with a stable PSNR of 47 to 50 dB, and at the same time has low computational complexity (O(NlogN)), which makes it suitable for deployment on devices with limited resources. Figure 21 presents a graph of comparison of the proposed irreversible steganographic method with modern reversible RDH-approaches ([4–7]) in terms of capacity (bpp) and visual quality (PSNR).

According to the presented graph, the inverse dependence is visible, when high capacity is achieved by reducing the PSNR, but its value remains at a good level. The authors of works [4–7] evaluate the quality by PSNR and visual analysis of histograms, but our method passes a comprehensive assessment, including χ^2-test, JS-divergence, RS-analysis, and stegoanalysis based on SRM and ensemble classifier.

The approach we developed can be easily adapted to the problems of RDH methods. To achieve this, the algorithm is only supplemented by the interpolation process of the source container to use the interpolated values as immutable “white” pixels in our described method.

The HILL method [8], on the contrary, is designed for maximum resistance to modern steganalysis at a moderate capacity of about 0.4 bpp. With PSNR in the range of approximately 48–52 dB, it requires significantly more computational resources compared to the approach proposed in this paper.

PVD [11] provides up to 0.215 bpp (56,291 bits in a 512×512 image) with PSNR in the range of 37.9–48.4 dB depending on the texture of the image. In contrast, the proposed method reaches a significantly higher capacity of up to 0.69 bpp with a comparable PSNR of approximately 47–50 dB. However, like PVD, the current implementation is vulnerable to modern SRM-based steganalysis. The methods proposed in [12,13] further increase resistance to steganalysis due to a more accurate assessment of local complexity and preservation of pixel ordering. However, their maximum capacity is limited to 0.16 bpp in [12] at PSNR $\approx$ 47 dB, and up to 0.2 bpp in [13] with PSNR in the range of 44.2–50.6 dB.

The authors in [11–14] argue that their methods are resistant to RS analysis because they do not use LSB substitution but instead modify pixel values as a whole. The proposed method also demonstrates resistance to RS analysis, which is experimentally tested and confirmed in Tables 6–8. In addition, the visual quality of the resulting stegimages, measured in terms of PSNR as a function of payload, is illustrated in Figure 26. In [12], methods [11,12] are evaluated for resistance to modern steganalysis using feature sets Shi-78D, Farid-72D, Moulin-156D, and Li-110D, which are direct precursors of the spatial rich model (SRM). At a capacity of 0.2 bpp, the classical PVD [11] is detected with an accuracy of approximately 88%, and adaptive PVD [12] at a capacity of 0.16 bpp with an accuracy of approximately 60% (still significantly higher than random guessing at 50%), which confirms the vulnerability of both methods to SRM-like analysis. Methods [13,14], being PVD-based approaches without optimization of the cost function for statistical indistinguishability, are presumably also vulnerable to SRM-based steganalysis, although direct tests were not conducted in the referenced studies. For clarity, Figure 27 presents a comparison of methods [5,11] and the proposed approach in terms of steganographic resistance as a function of embedding capacity (bpp). On the ordinate axis, a generalized resistance level is shown: values up to 20% correspond to vulnerability to simple statistical characteristics; 20–40% indicate resistance to RS analysis but vulnerability to statistical detectors; and 40–100% correspond to resistance to modern steganalysis based on spatial rich models (SRMs).

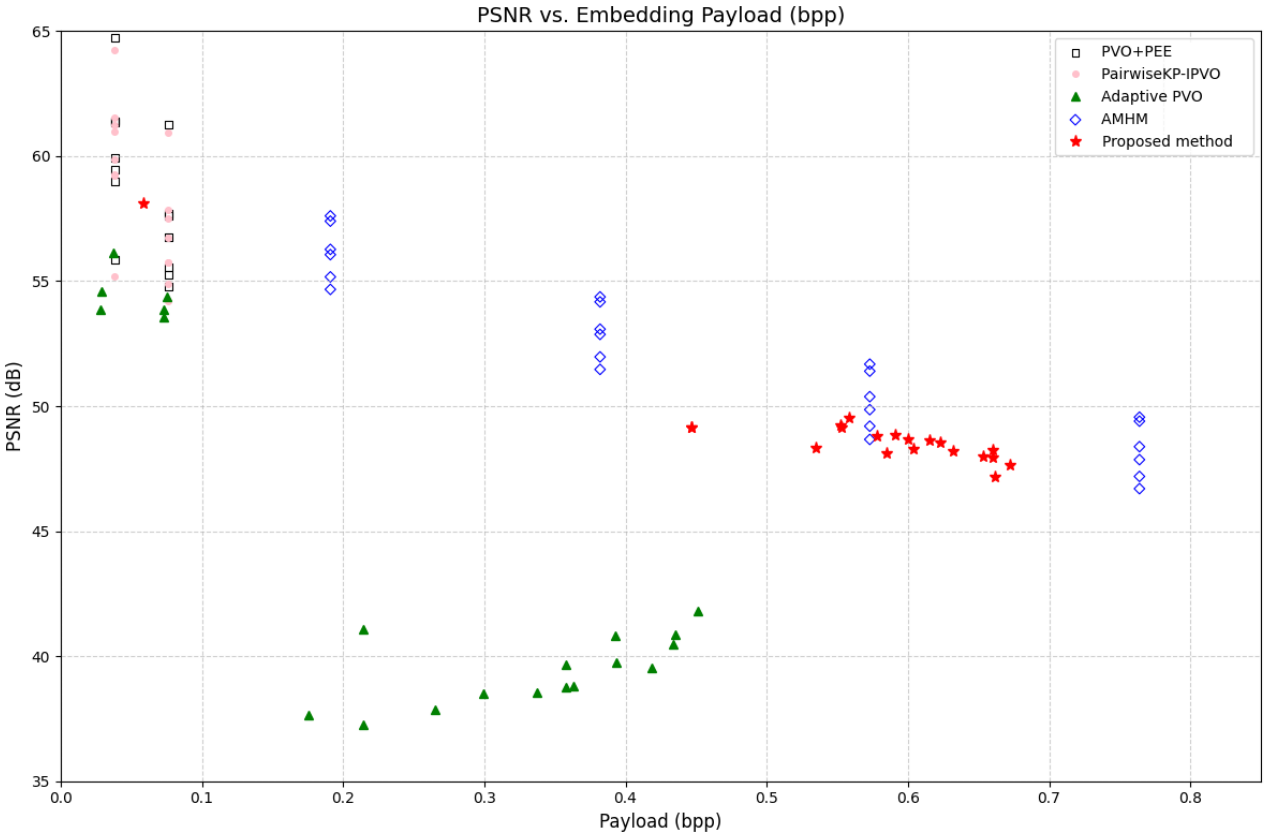


Figure 26. Quality of stegimages (PSNR, dB) depending on payload (bpp) for the proposed method and methods of reversible data hiding [4–7]. https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026).

The rest of the results can be viewed at the following link: https://github.com/KaterinaMerzlyakova/Application-of-a-Linear-Hash-Function-in-Adaptive-Image-Steganography/wiki (accessed on 6 January 2026). In work [15], an adaptive approach with a threshold strategy is interesting, as it dynamically distributes bits depending on

the structure of the codeword, but the process takes place in the VQ codebook, not in the spatial region, so comparison with our method is not appropriate.

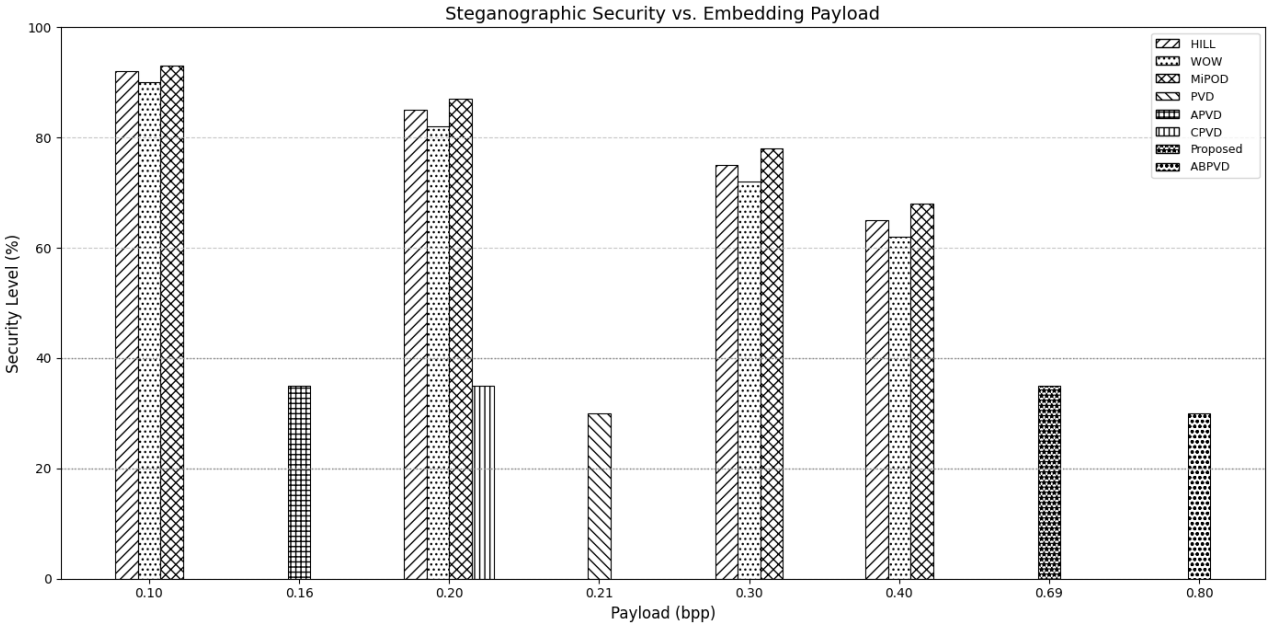


Figure 27. Comparison of steganographic methods in terms of the level of resistance and capacity of the embedding.

Methods [16,17] offer adaptive embedding based on fuzzy logic and contrast analysis, respectively, reaching a capacity of up to 0.8–1.2 bpp at PSNRR from 47 to 58 dB. However, these results are obtained either at low capacitance in work [17], PSNR is about 62 dB at ≈0.03 bpp, or on textured images in the work [16], where it is up to 0.8 bpp on the image aerial. Both methods, like ours, are RS-resistant, but have not been evaluated for SRM resistance, indicating their potential vulnerability to modern stegoanalysis. In the context of applying a linear hash function in methods with adaptive strategies, it is worth noting the idea from paper [16] that more bits can be introduced into darker areas of an image. By modifying our approach with only an additional assumption regarding this fact, it is possible to easily increase the capacity of the implementation by adjusting the type of block τ depending on the brightness, along with the gradient. In addition, in [17], the idea of using a combined criterion for assessing local complexity is of value, which can be useful in our adaptive method, since we can easily complement this aspect without changing the approach with the linear hash function.

In works [19–21], modern but resource-intensive approaches are proposed using the trained cost function, bearing the idea of distributing the capacity depending on the local texture complexity of the image. Although such methods provide high resistance to modern stegoanalysis, their computational complexity and the need for training make them unsuitable for resource-limited systems. However, the idea of adaptive capacity allocation can be integrated into our architecture to increase its statistical indistinguishability.

Table 9 presents the comparative characteristics of the proposed method and the most relevant modern analogs.

Thus, the proposed method is positioned not as a direct alternative TO HILL or WOW in the problems of maximum stealth but as a basic protocol for trusted transmission channels. In contrast to methods based on deep learning, which require a learning stage and do not guarantee the reproducibility of the result on different equipment, the proposed approach provides deterministic complexity.

Table 9. Comparison of the proposed method with modern adaptive approaches.

Method	Testing Range, bpp	Cost Function Domain	Complexity	Security Metric
Proposed method	0.50–0.69	Spatial (gradients + paired differences)	$O(N \log N)$	χ^2, RS, JS divergence
HILL [8]	0.10–0.50	Spatial (3-layer filtering)	$O(N^2)$	SRM
WOW [9]	0.10–0.50	Wavelet domain (Directional Filters, WDFB-D, Daubechies-8)	$O(N^2)$	SRM
S-UNIWARD [10]	0.10–0.50	Wavelet domain (relative distortion)	$O(N^2)$	SRM

5. Conclusions

In the course of the study, an adaptive steganographic method based on the use of a linear hash function over the GF(2) field and gradient classification of image blocks was implemented and comprehensively tested. An experimental evaluation conducted on three heterogeneous datasets showed that the proposed approach does not introduce statistically significant distortions in the global characteristics of the container. Analysis of brightness histograms, the χ^2-test for the uniformity of the distribution of junior significant bits, as well as the Jensen–Shannon divergence (JS) estimate confirm a high degree of indistinguishability of stego images from the original cover images at the level of macrostatistics. The advantages of the approach are high computing efficiency (243 ms per image 512×512), adaptive capacity (0.01–0.7 bpp at PSNR 47–50 dB), and modularity of the architecture. Limitations of the method include vulnerability to modern SRM-based steganalysis and lack of robustness to common attacks (BCR $\approx 50\%$), which determines its applicability exclusively to trusted transmission channels. Potential use cases include embedding metadata into medical archives, covert data transfer in corporate networks, and content authentication in IoT devices. At the same time, the architecture of the method provides a simple integration of adaptive mechanisms in the future, which makes it promising for further research. Thus, the work represents a research basis for the development of computationally efficient stegosystems combining low implementation complexity, high and adaptive capacity, and potential resistance to modern stego analysis.

Author Contributions: Conceptualization, E.M. and E.D.; methodology, E.M.; software, E.M.; validation, L.C. and A.Y.; formal analysis, B.A. and N.S.; investigation, A.Y. and G.S.; resources, E.D. and N.S.; data curation, L.C.; writing—original draft preparation, E.D. and G.S.; writing—review and editing, A.Y.; visualization, L.C. and B.A.; supervision, E.M. and N.S.; project administration, A.Y. All authors have read and agreed to the published version of the manuscript.

Funding: The study was done with a support of the state assignment of SibSUTIS (reg. no. 071-03-2026-011).

Data Availability Statement: The data supporting the findings of this study are publicly available at: https://drive.google.com/drive/folders/1LcvhZe-lhYFMVLq9hexo8HJJa3Y-uUMM?usp=drive_link, accessed on 16 April 2026.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Daiyrbayeva, E.; Merzlyakova, E.; Yerimbetova, A.; Mukhitova, A. Reversible Steganographic System for the Transmission of Personal Medical Data. In Proceedings of the 9th International Conference on Computer Science and Engineering (UBMK), Antalya, Turkiye, 26–28 October 2024; pp. 1–6. [CrossRef]
2. Daiyrbayeva, E.; Yerimbetova, A.; Nechta, I.; Merzlyakova, E.; Toigozhinova, A.; Turganbayev, A. A Study of the Information Embedding Method into Raster Image Based on Interpolation. *J. Imaging* **2022**, *8*, 288. [CrossRef] [PubMed]
3. Daiyrbayeva, E.; Yerimbetova, A.; Merzlyakova, E.; Sadyk, U.; Sarina, A.; Taichik, Z.; Ismailova, I.; Iztleuov, Y.; Nurmangaliyev, A. An Adaptive Steganographic Method for Reversible Information Embedding in X-Ray Images. *Computers* **2025**, *14*, 386. [CrossRef]
4. Kumar, N.; Kumar, R.; Malik, A.; Singh, S.; Jung, K.-H. Reversible Data Hiding with High Visual Quality Using Pairwise PVO and PEE. *Multimed. Tools Appl.* **2023**, *82*, 30733–30758. [CrossRef]
5. Kong, X.; He, W.; Cai, Z. A Novel High-Fidelity Reversible Data Hiding Method Based on Adaptive Multi-Pass Embedding. *Mathematics* **2025**, *13*, 1881. [CrossRef]
6. Nguyen, T.D.; Dao, T.T. A Novel Adaptive PVO-Based Reversible Data Hiding Method Using Optimized Low and High Threshold Estimation for Gray Images. *SN Comput. Sci.* **2025**, *6*, 740. [CrossRef]
7. He, W.; Xiong, G.; Wang, R. Reversible Data Hiding Based on Adaptive Multiple Histograms Modification. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 3000–3012. [CrossRef]
8. Li, B.; Wang, M.; Huang, J.; Li, X. A New Cost Function for Spatial Image Steganography. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 4206–4210. [CrossRef]
9. Holub, V.; Fridrich, J. Designing Steganographic Distortion Using Directional Filters. In Proceedings of the IEEE WIFS, Tenerife, Spain, 2–5 December 2012. [CrossRef]
10. Holub, V.; Fridrich, J. Digital Image Steganography Using Universal Distortion. In Proceedings of the IH&MMSec, Montpellier, France, 17–19 June 2013; pp. 29–38. [CrossRef]
11. Wu, D.-C.; Tsai, W.-H. A Steganographic Method for Images by Pixel-Value Differencing. *Pattern Recognit. Lett.* **2003**, *24*, 1613–1626. [CrossRef]
12. Luo, W.; Huang, F.; Huang, J. A More Secure Steganography Based on Adaptive Pixel-Value Differencing Scheme. *Multimedia Tools Appl.* **2011**, *52*, 407–430. [CrossRef]
13. Swain, G.; Lenka, S. Pixel Value Differencing Steganography Using Correlation of Target Pixel with Neighboring Pixels. In Proceedings of the ICECCT, Coimbatore, India, 5–7 March 2015; pp. 1–6. [CrossRef]
14. Hosameldeen, O.; Ben Halima, N. Adaptive Block-Based Pixel Value Descending Steganography. *Secur. Commun. Netw.* **2016**, *9*, 5036–5050. [CrossRef]
15. Lin, Y.; Liu, J.-C.; Chang, C.-C.; Chang, C.-C. Embracing Secret Data in a Vector Quantization Codebook Using a Novel Thresholding Scheme. *Mathematics* **2024**, *12*, 1332. [CrossRef]
16. Ananti, M.; D'Layla, A.; Ntivuguruzwa, C.; Ahmad, T. FuzzyStego: An Adaptive Steganographic Scheme Using Fuzzy Logic. *Comput. Mater. Contin.* **2025**, *84*, 1031–1054. [CrossRef]
17. Jamatia, R.; Bhuyan, B. A Contrast-Channel Embracing Approach for Image Steganography. *IEEE Access* **2025**, *13*, 180539–180565. [CrossRef]
18. Tang, W.; Tan, S.; Li, B.; Huang, J. Automatic Steganographic Distortion Learning Using GAN. *IEEE Signal Process. Lett.* **2017**, *24*, 1547–1551. [CrossRef]
19. Yang, J.; Liu, K.; Kang, X.; Wong, E.; Shi, Y.-Q. Spatial Image Steganography Based on GAN. *arXiv* **2018**, arXiv:1804.07939. [CrossRef]
20. Tang, W.; Li, B.; Tan, S.; Barni, M.; Huang, J. CNN-Based Adversarial Embedding. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 2074–2087. [CrossRef]
21. Sanjalawe, Y.; Al-Emari, S.; Fraihat, S.; Abualhaj, M.; Alzubi, E. Deep Learning-Driven Steganography. *Sci. Rep.* **2025**, *15*, 4761. [CrossRef] [PubMed]
22. Gong, L.-H.; Xiang, D.-N.; Wu, J.-Y.; Chen, S.-H. Image Encryption-Authentication Scheme Based on Pixel Adaptive Diffusion. *J. Mod. Opt.* **2026**. [CrossRef]
23. Ryabko, B.; Fionov, A. Linear Hash Functions for Distortion-Rate Optimization. In Proceedings of the IH&MMSec, Paris, France, 3–5 July 2019; pp. 235–238.
24. StegoHash Repository. Available online: https://github.com/ivannechta/StegoHash (accessed on 8 January 2026).
25. Lidl, R.; Niederreiter, G. *Finite Fields*; Mir: Moscow, Russia, 1988.
26. Image Database for Steganography Research. Available online: https://drive.google.com/drive/folders/1LcvhZe-lhYFMVLq9hexo8HJJa3Y-uUMM?usp=drive_link (accessed on 8 January 2026).
27. SIPI Image Database. Available online: https://sipi.usc.edu/database/ (accessed on 8 January 2026).
28. Fracture X-Ray Dataset. Available online: https://www.kaggle.com/datasets/bmadushanirodrigo/fracture-multi-region-x-ray-data/data (accessed on 8 January 2026).

29. Al-Dakrnai, A.M.; Aref, M.A. Medullary Spongy Renal Dataset. Kaggle. 2025. Available online: https://www.kaggle.com/datasets/sa3dola555/medullary-sponge-kidney (accessed on 8 January 2026).
30. Fridrich, J.; Kodovsky, J. Rich Models for Steganalysis. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 868–882. [CrossRef]
31. Kodovsky, J.; Fridrich, J.; Holub, V. Ensemble Classifiers for Steganalysis. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 432–444. [CrossRef]