Ingénierie des Systèmes d'Information
Vol. 30, No. 6, June, 2025, pp. 1535-1544
Journal homepage: http://iieta.org/journals/isi

An Intelligent Steganographic Scheme Using Video Frame Neighborhoods

Fayez Khazalah[1*], Hesham Al-Rawashdeh[2], Nashat Al Bdour[2], Ayman M. Mansour[2]

[1] Department of Information Systems, College of Information Technology, Al al-Bayt University, Mafraq 25113, Jordan
[2] Department of Computer and Communication Engineering, College of Engineering, Tafila Technical University, Tafila 66110, Jordan

Corresponding Author Email: fayez@aabu.edu.jo

https://doi.org/10.18280/isi.300612

Received: 8 May 2025
Revised: 11 June 2025
Accepted: 23 June 2025
Available online: 30 June 2025

Keywords:
container templates, image, steganography, thresholding, video frames, video frame analysis

ABSTRACT

This paper describes a method for steganographic security of information based on video. Using videos allows for hiding more secret data. The method is based on dividing the video into an ordered sequence of video frames and analyzing groups of video frames to select pixels in each video file, into which secret bits are embedded. Video frame analysis is performed by determining the difference between adjacent video frames and forming an array of numbers that determines the magnitude of the difference for the code of each pixel. By using a threshold processing, we can identify pixels where more secret bits can be hidden compared to the LSB algorithm. Based on the analysis of adjacent video frames and the application of threshold processing, templates are formed according to which secret information is embedded in the video. Traditional steganographic methods, such as LSB substitution, face challenges related to limited embedding capacity and vulnerability to common signal processing attacks. These drawbacks restrict their effectiveness in practical, high-security data hiding scenarios. To overcome these limitations, we propose an intelligent video steganography technique based on interframe pixel differences and adaptive thresholding. By identifying regions with significant temporal variation, the method selectively embeds multiple secret bits using a threshold-guided LSB approach. Additionally, a dynamic duplication mechanism across color channels is employed to improve redundancy and robustness without compromising visual quality. Experimental results show a notable increase in both embedding capacity and resistance to compression and noise, outperforming traditional LSB-based techniques.

1. INTRODUCTION

In the modern world, modern means of communication and computing technologies are being rapidly introduced on a global scale. This entails the need to exchange large amounts of information. In this regard, specialists pay special attention to the high-quality protection of transmitted information from various types of threats. One of the approaches to implementing information protection is the use of steganographic systems, which are based on hiding secret information in information containers, presented in the form of files of various formats (graphic, audio, text files, etc.).

One of the main challenges is to increase the volume of embedded information without noticeable distortion of the container. In such situations, large volumes of hidden information require containers that also contain large volumes of information. The volume of the container must be many times greater than the volume of the secret information being introduced. Among all possible containers of large volumes of information, videos can be used. Unlike other containers, in addition to the Least Significant Bit (LSB) method [1-3], other steganographic methods [2-6] can be implemented in such a container. This is due to the fact that video contains dynamic changes in the visual picture, unlike containers that contain stationary, time-invariant information. It is more difficult to detect visual deviations from the original video in dynamically changing visual images. This is especially true if such a video is not formed by a stationary video camera, in which visual changes are observed practically throughout the entire visual field. This allows the use of dynamically changing sections of a video to embed secret information into them using combinations of various steganographic methods based on the use of images as containers [1-3, 7, 8]. In addition, processing groups of images containing approximately the same visual content allows the use of additional methods aimed at processing several images to select pixels in each video frame, into the codes of which secret bits are embedded without significant changes in the visual picture of the video. This approach allows for a significant increase in the volume of embedded information with high resistance to enemy attacks.

2. RELATED WORK

The main goal of video-based steganography is to embed secret bits into video graphics files by replacing the original bits in the pixel codes without visible distortion. The earliest and simplest method of steganography based on containers

represented by graphic files is the LSB replacement method [1-3]. Its main drawback is that the size of the embedded message is limited by the container size. In addition, the LSB method can lead to a violation of the integrity of the graphic files of the containers. To improve the quality of steganographic protection, several methods have been developed at present time. In works [4-6], a detailed review and comparative analysis of existing video-based methods were conducted. The study [9] describes a combined method of steganographic protection based on K-means and the LSB function. However, the need to form clusters can lead to false results on the receiving side because the initial choice of initial values may be unsuccessful.

Steganographic methods that use color content have been proposed [10]. Color space conversion was used, which may result in false elements or loss of information. RGB analysis and transformation for steganographic concealment were presented in study [11]. However, transformation and random embedding do not fully increase the volume of the embedded information. Another combined method of steganographic protection is based on the use of LSB and the DCT algorithm [12]. A lossless compression algorithm was used here, which also limits the amount of embedded secret information.

Steganographic security methods that use BCH codes [13, 14] and Hamming codes [15] have been proposed. However, these methods are not sufficiently resistant to attacks.

In study [16], video frames were analyzed, and suitable secret bits for embedding were selected. This method does not use all video frames to hide secret bits, which limits the amount of information that can be embedded in the video.

Compressed videos occupy less space, and secret bits are embedded during or after compression [17]. This method can lead to different problems and errors for different codecs. MPEG formats are most often used to hide information [18]. The main methods used for embedding information in the MPEG-2 format are embedding at the coefficient level, embedding at the bit plane level, and embedding owing to the energy difference between coefficients [19]. The study [17] describes a method for embedding audio data into files represented by the AVI video format. Secret data are embedded into the information part, considering repeating blocks in the container structure. This approach does not always yield the desired result because the number of repeating blocks for different videos can be limited, which often limits the amount of embedded information.

Video format files are divided into a sequence of video frames, for which steganography methods are applied to. However, most methods do not use a detailed analysis of groups of adjacent video frames to select the pixels into which the secret bits are embedded in their codes.

3. PROBLEM FORMULATIONS

The goal of this research is to increase the volume of information embedded in containers represented by a video format file by using additional processing of two adjacent video frames to select pixels that have a difference greater than a given threshold value. The division of color byte codes into threshold layers also allows solving the problem, in addition to increasing the volume of embedded information and increasing resistance to enemy attacks owing to the possibility of duplicating embedded information.

The methodology adopted in this study is grounded in the intelligent exploitation of temporal changes between video frames to enable content-adaptive steganographic embedding. Initially, the video is divided into discrete frames, which are grouped in overlapping pairs to facilitate interframe difference computation. This approach is particularly effective for video sequences captured by stationary cameras, where minimal motion results in limited pixel-level differences. Conversely, for videos recorded by moving cameras or scenes with dynamic content, the interframe differences are more widespread, offering greater embedding opportunities.

Each pair of adjacent frames is processed to calculate the pixel-wise differences across the full 24-bit RGB space. These differences are separated into red, green, and blue byte-level components to enable finer control during subsequent processing. To enhance both security and embedding efficiency, thresholding is applied to each color byte array. This selective filtering allows the system to ignore minor changes that are unlikely to be perceptually significant and instead focus on pixels with notable differences. The threshold values can be tuned to match the visual properties of a specific video, ensuring an optimal balance between payload capacity and invisibility.

Templates are generated based on the thresholded differences, identifying the spatial locations within the frames where data can be hidden. These templates vary from frame to frame and color to color, providing a high degree of adaptability and security for users. In the embedding phase, secret bits are inserted into the least significant bits of the selected pixel bytes according to a hierarchical scheme, wherein more bits are embedded in pixels with larger differences. This variable-depth embedding minimizes the distortion in flat regions while maximizing the capacity in visually active areas. Furthermore, by duplicating the embedded data across different frames or color channels using diverse templates, the system enhances resilience against signal degradation or intentional attacks. Experimental results confirm that this methodology significantly increases the amount of secret data that can be embedded while maintaining imperceptibility and robustness across diverse video types.

The algorithm proposed in this study is a steganographic embedding technique that utilizes threshold-based interframe difference analysis to increase the payload capacity of video containers while maintaining visual imperceptibility. The process begins by decomposing the input video into an ordered sequence of individual frames, where each frame is treated as a static image composed of pixels represented by 24-bit RGB values. The core idea is to analyze pairs of adjacent video frames to compute pixel-wise differences between them. These differences are calculated by subtracting the corresponding pixel values of one frame from the next, and the resulting difference values are considered in their absolute form.

Next, the RGB values of the differing pixels are separated into their individual color components—red, green, and blue—resulting in three arrays of pixel byte differences. A thresholding process is then applied to each array. Only pixels whose color channel differences exceed a predefined threshold (e.g., 50, 100, or 200) are selected for embedding. This step is crucial for ensuring that only visually significant changes are utilized, thereby reducing the chance of noticeable distortion.

For each color channel in a given frame, a corresponding binary mask or "template" is generated to indicate which pixels are eligible for embedding the secret information. In these templates, pixels with zero values indicate no embedding,

whereas non-zero values identify locations for hiding secret bits. The number of bits embedded in each selected pixel is dynamically determined based on the magnitude of the interframe difference; higher differences allow embedding of more bits, up to four in some cases. Embedding is performed in the LSBs of the pixel color byte, maintaining the visual integrity of the video. This stepwise process is repeated for all frame pairs in the video, producing a robust and high-capacity steganographic system.

4. THE PROPOSED METHOD

We propose a method for embedding secret bits inside video stream frames based on interframe difference analysis and threshold processing. The method of hiding a message in a video is based on splitting a video file into a sequence of video frames, each of which is an image. Video frames can be identical (if there are no moving objects or dynamically changing processes in the video) or can differ in the states of pixels that have the same location in the image. Adjacent video frames are of particular interest, as they can coincide in any video if the frame rate is very high.

If messages are hidden in adjacent video frames, visual differences may be observed in adjacent video frames and the entire video. For videos not obtained from stationary cameras, differences in adjacent video frames are almost always present. However, identical adjacent video frames are often found in videos obtained from stationary video cameras. Moreover, the number of different pixels in adjacent video frames is much lower than that in videos obtained from nonfixed video cameras. This approach is based on determining the difference in video frames [20, 21].

Examples of video frames for stationary and nonstationary video cameras are shown in Figure 1. It shows video frames in which pixels that changed their state and underwent threshold processing in accordance with the specified thresholds (45 and 150) in the next video frame are highlighted in red. The red rectangular frame highlights areas with different values of the difference thresholds in two adjacent video frames. Because the maximum value for each byte of an RGB image is 255, the differences are recorded in the range of 0–255. Differences in pixel codes can be recorded in both individual color bytes and the overall pixel code, the maximum value of which for an RGB image is 16777215. The difference in pixel codes between two images can be either negative or positive. As a rule, such quantities are taken as absolute values with a positive sign.

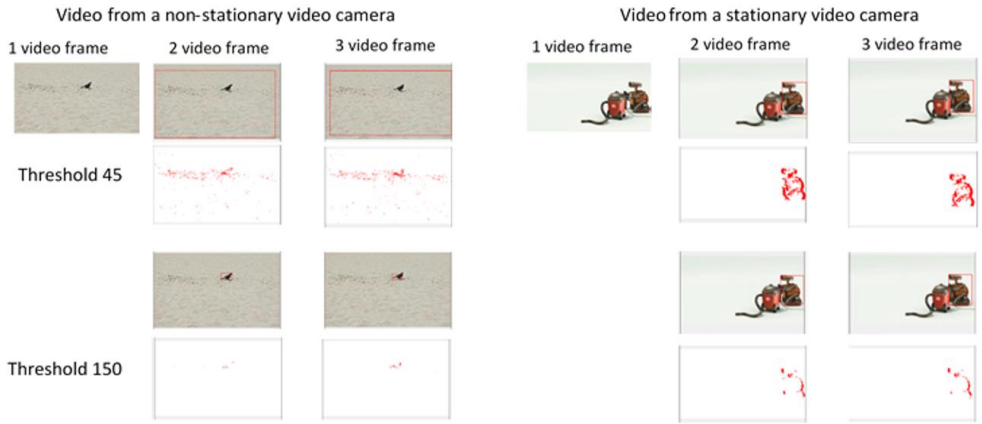

Figure 1. Examples of video frames obtained from stationary and non-stationary video cameras

Using the difference in the pixel codes of adjacent video frames, an algorithm for hiding a secret message in a video is proposed, which consists of the following steps:

(1) The video is split into a sequence of frames.
(2) The difference in the codes of the pixels with the same coordinates in adjacent video frames is determined. To determine the difference, video frames from the initial video are used.
(3) The codes of the pixels in which the presence of differences is determined are divided into three color bytes: R, G, and B.
(4) Threshold processing is applied to the values of the codes of each byte of pixels greater than 0 in absolute value. The threshold value in the range from 0 to 255 is set, and the bytes of those pixels whose codes exceed the threshold value are selected.
(5) In accordance with the allocated pixels for the bytes of each color, templates with allocated pixels are formed. Each template is an image the size of a video frame with black pixels (code 0), into whose codes the bits of the secret message are not embedded, and with pixels of a different color (code greater than 0), into whose codes the bits of the secret message are embedded. For each video frame, three templates are generated: R-template, G-template, and B-template. These patterns are used to embed secret bits into the bytes encoding the colors red, green, and blue, respectively.
(6) For each selected pixel, the code value is analyzed to determine the difference between two adjacent video frames. The number of least significant bits in the color byte into which the secret bits are embedded is determined based on the magnitude of the difference.
(7) In accordance with the selected templates and the results of the analysis of the difference values, the bits of the secret message are implemented.

Videos can be generated using a stationary camera or a camera mounted on a moving object. Moving objects can be human hands, cars, aircraft, or other objects. If it is necessary to embed a large volume of information, the most effective container is a video consisting of many containers, which are images of the same format and size. For the proposed method, the volume of the embedded image can be increased using a video obtained from a nonstationary video camera.

In the first step, the video, which is used as a steganographic container, is broken down into an ordered sequence of video frames (images), in which the color and brightness

characteristics of each pixel are encoded using a code formed by three bytes.

In the second step of the algorithm, two adjacent video frames are grouped such that every second video frame of the previous group is the first video frame of the next group. The result is a multitude of groups:

$$V = \{(f_1, f_2), (f_2, f_3), \dots, (f_{i-1}, f_i), \dots, (f_{n-1}, f_n)\},$$

Giving a set of arrays of numbers that are the result of comparing two numbers of adjacent video frames f_{i-1} and f_i.

Using the comparison (subtraction) operation on arrays of numbers (images) in each group are formed a set of arrays of numbers:

$$R = \{(r_{1,2}), (r_{2,3}), \dots, (r_{i-1,i}), \dots, (r_{n-1,n})\},$$

where, $r_{i-1,i} = r_{i-1} - r_i$.

This expression considers the entire three-byte code of each pixel. Examples of the obtained arrays of numbers for a sequence of three video frames are shown in Figure 2.

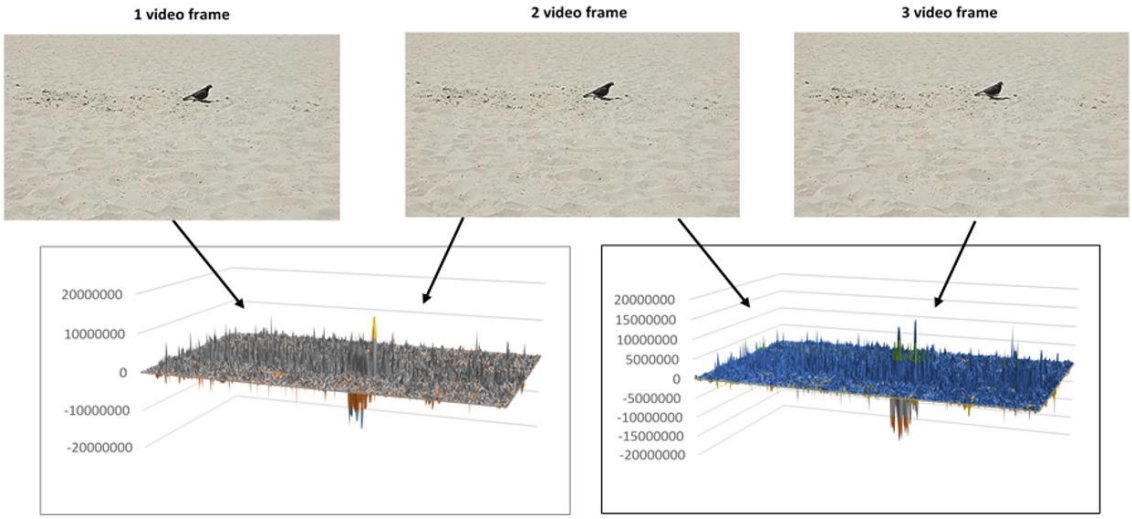

Figure 2. Examples of forming arrays of numbers as a result of comparing adjacent video frames

In Figure 2, the obtained arrays $r_{1,2}$ and $r_{2,3}$ are presented in the form of a surface on which positive and negative peaks are present. For videos generated from a nonstationary video camera, such peaks are present over almost the entire surface. The second example shows fewer peaks than the first. They are present only in pixels that indicate dynamic changes. For ease of implementation of the algorithm, all obtained values are considered positive.

Values greater than 0 are determined, and the codes of the corresponding pixels are divided into three bytes, forming red, green, and blue colors. Three arrays are formed from the received bytes R_R, R_G and R_B as following:

$$\begin{cases} R_R = \{r^R_{1,2}, r^R_{2,3}, \dots, r^R_{i-1,i}, \dots, r^R_{n-1,n}\} \\ R_G = \{r^G_{1,2}, r^G_{2,3}, \dots, r^G_{i-1,i}, \dots, r^G_{n-1,n}\} \\ R_B = \{r^B_{1,2}, r^B_{2,3}, \dots, r^B_{i-1,i}, \dots, r^B_{n-1,n}\} \end{cases}$$

These arrays define the differences in the corresponding three-color bytes. Examples of such arrays are presented in Figure 3. As shown, for each of the three arrays obtained by comparing two adjacent video frames, there are differences. For example, the difference between the bytes that make up the red color may be 0, while for the bytes that make up the green or blue color, the difference may be different from 0. For the presented example (Figure 3), all three obtained surfaces for the arrays $r^R_{i-1,i}$, $r^G_{i-1,i}$ and $r^B_{i-1,i}$ have differences. In some areas, such differences are significant.

To hide secret bits more reliably, a fourth stage of the algorithm is implemented, in which threshold processing is applied to each generated array, which reduces the number of pixels indicating the presence of a difference between adjacent video frames. Examples of all three arrays formed after applying threshold processing are shown in Figure 4. Threshold values of 50, 100, and 200 were used. All obtained values are presented in absolute value as positive numbers.

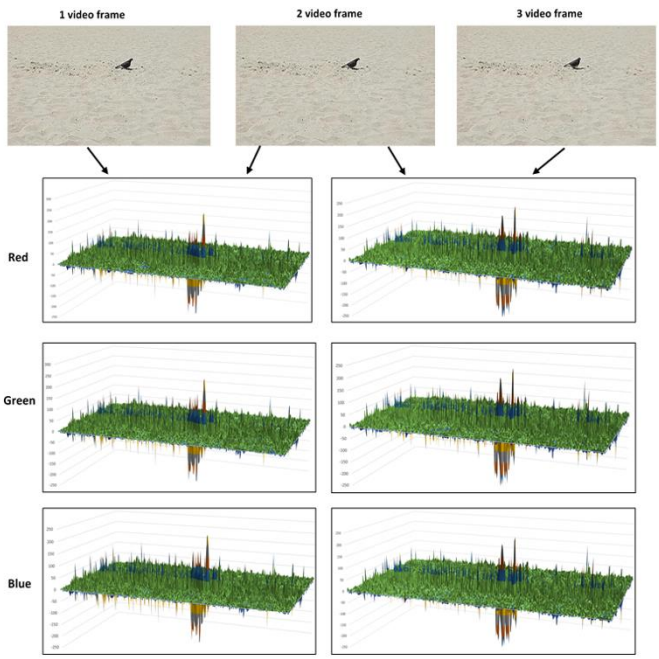

Figure 3. Examples of arrays of numbers obtained as a result of splitting into three color bytes and comparing adjacent video frames

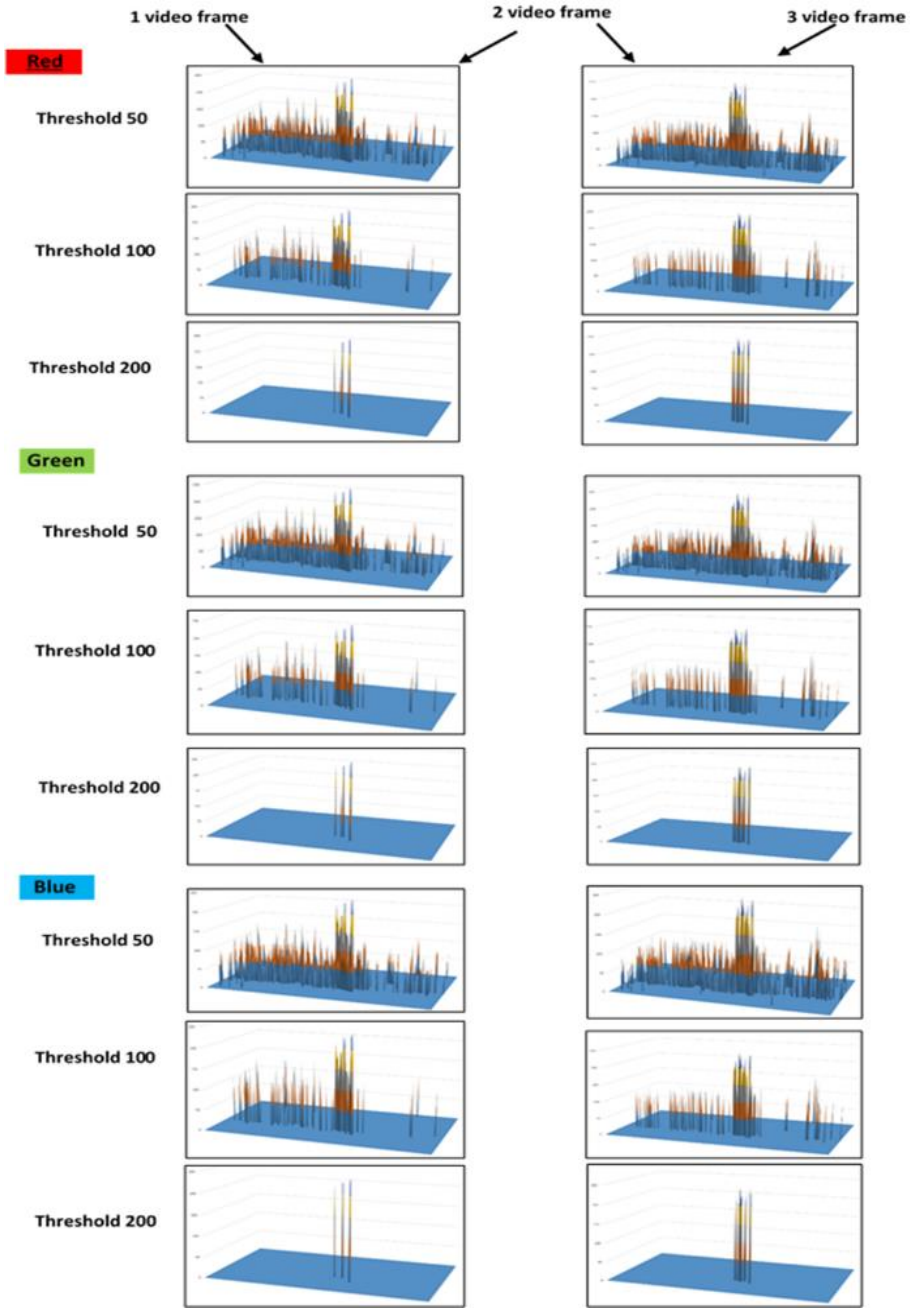

Figure 4. Generated arrays after using threshold processing with values of 50, 100 and 200 for the video frames shown in Figure 3

In the fifth stage of the algorithm, templates are formed, based on which pixels are selected in the following video frames for embedding secret bits into their codes (Figure 5). In addition, a selection of pixels is made into the codes of which a different number of secret bits can be embedded (sixth stage of the algorithm).

For example, one can set conditions that if the bytecode does not exceed the value of 50 but is greater than 0, then one secret bit can be embedded in the least significant bit of this code, and if the code exceeds the threshold of 50 but is less than 100, then the secret bits can be embedded in the two least significant bits of the byte of the corresponding pixel, etc. Thus, the number of secret bits embedded in the bytecode of each pixel depends on the magnitude of the difference in the codes of the corresponding pixels in two adjacent video frames. The threshold values for each video can be individually selected to avoid disrupting the visual differences in the video.

The final stage (Stage 7) involves the introduction of secret bits into the pixel codes of each video frame, considering the obtained templates. At this stage, the sequence of enumerating pixels for each template is set, and the sequence of enumerating templates and bytes for the image of each video frame is also set. There are different sets of sequences for selecting the codes of the selected pixels. The simplest method is to sequentially embed secret bits according to patterns obtained by analyzing the red, green, and blue bytes.

If, after applying threshold processing, several templates are formed for each color byte (Figure 4), then these templates are also considered when forming the sequence of pixel iteration. The sequence of pixel code selection is often specified in advance and can be generated using a special program or device.

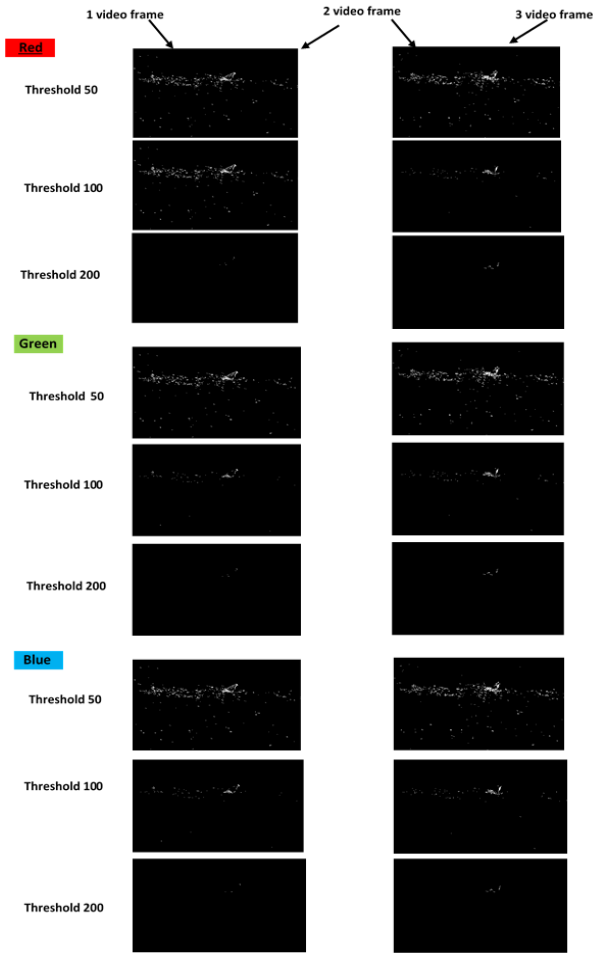

Figure 5. Generated templates according to the examples presented in Figure 4

Experimental studies were conducted for different bit sequences, the bits of which were embedded in the least significant bits of the pixel codes in containers, and the bits were embedded in different numbers of least significant bits of the selected pixel codes. Sequences consisting of all zeros or all ones, as well as bit sequences generated by a pseudo-random number generator, were used as hidden bit sequences. An example of such an approach without thresholding is shown in Figure 6. It shows the same video frames with secret bits embedded in the lower 0, 1, 2, 3, and 4 bits. In this case, no significant visual differences are observed in the steady state when implemented in one least significant bit. Differences are also observed when embedding the bits of the bit sequence into the least significant bits of each color file, starting with the first least significant bit. Moreover, such differences are visually noticeable regardless of which bit sequence is implemented.

Using thresholding allows us to generate more templates for each video frame. For each video frame, templates of different shapes are formed, which depend on the structure of two adjacent video frames. In accordance with the templates obtained, pixels are selected, into the bytes of which a different number of secret bits are embedded. An example on secret bits embedding after thresholding is given in Figure 7.

Figure 7 shows different combinations of pixel selections that are used to further hide different numbers of secret bits. For a threshold of 50, video frames are presented in which two secret bits are embedded in the 4th and 5th least significant bits of the selected pixels. The results of embedding five secret bits into the pixel codes selected after applying thresholds of 100 and 200 are shown below. The bottom image of the video frame shows the results of combining the number of embedded secret pixels for different thresholds. For the first two least significant bits, two secret bits are embedded at a threshold of 50; for the third least significant bit, secret bits are embedded at a threshold of 100; and for the fourth least significant bit, at a threshold of 200. The examples provided do not show any significant visual differences when the videos are viewed.

This approach slightly complicates the process of introducing the secret bits. In this case, a large number of key templates are formed, which requires a large amount of memory to store them. For each video and each pair of video frames, its own key templates are generated. The receiving side must contain either the original videos or pre-formed templates. The proposed steganographic embedding methodology using interframe differences is summarized in Figure 8.

To objectively support the claim of minimal visual distortion when embedding secret bits, we evaluated the visual quality of the stego videos using standard metrics, including the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). Across all tested threshold values (50, 100, and 200), the PSNR consistently exceeded 40 dB, with average values of 42.3, 41.8, and 40.7 dB, respectively. Corresponding SSIM scores were above 0.95, indicating a high degree of structural similarity between the original and stego frames. These quantitative results confirm that our threshold-based embedding approach preserves visual fidelity effectively, validating the subjective observations illustrated in Figure 7.

To select appropriate threshold values for interframe difference processing, we conducted extensive empirical evaluations across various video types with different motion characteristics. The chosen thresholds of 50, 100, and 200 represent low, medium, and high sensitivity levels, respectively, for detecting pixel differences between adjacent frames. These thresholds enable adaptive selection of pixels based on the magnitude of changes, balancing embedding capacity and visual imperceptibility.

A lower threshold such as 50 allows embedding in a larger number of pixels, increasing the payload capacity but potentially introducing more visible distortion. Conversely, a higher threshold like 200 restricts embedding to pixels with more significant differences, preserving visual quality but reducing capacity. Our experiments consistently showed that all three thresholds maintain high perceptual quality, supported by PSNR values exceeding 40 dB and SSIM values above 0.98, indicating minimal visible differences compared to the original video.

Specifically, for a threshold of 50, the average PSNR was approximately 42.31 dB with an SSIM of 0.985, confirming excellent visual fidelity. Thresholds of 100 and 200 produced slightly lower but still strong results, with PSNR values of 41.85 dB and 40.67 dB and SSIM values of 0.983 and 0.980, respectively. These results demonstrate an effective trade-off managed by the threshold values, ensuring both high embedding capacity and low perceptibility.

To enhance reproducibility, we have included a detailed pseudocode for the template generation algorithm in the supplementary materials.

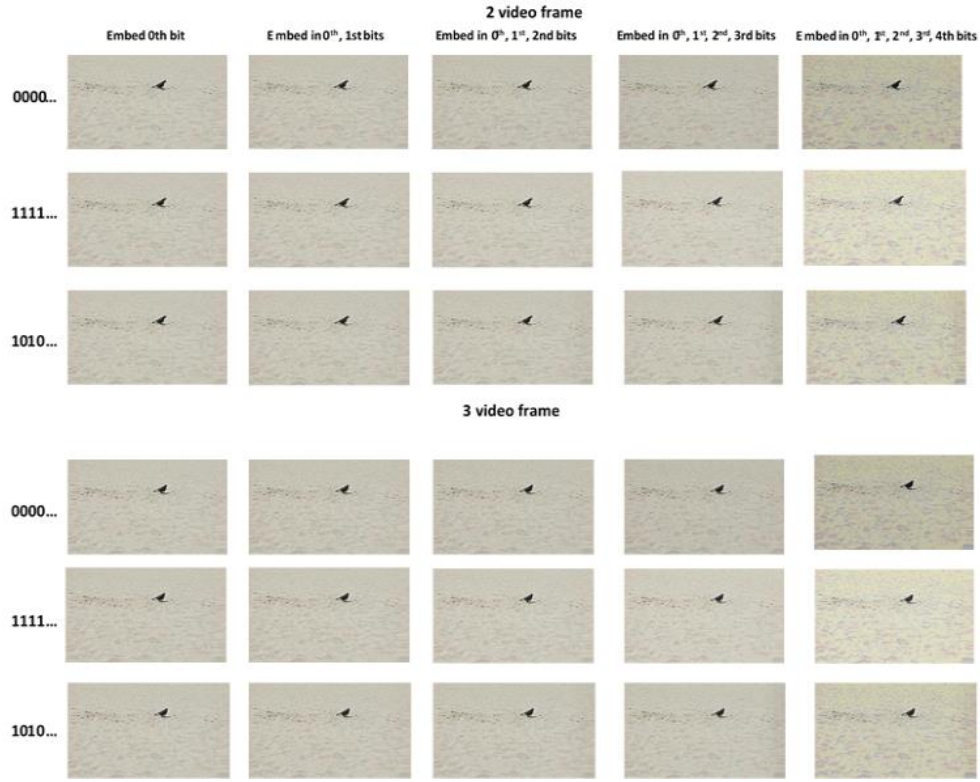

Figure 6. Examples of embedding secret bits into video frames of a video sequence without threshold processing

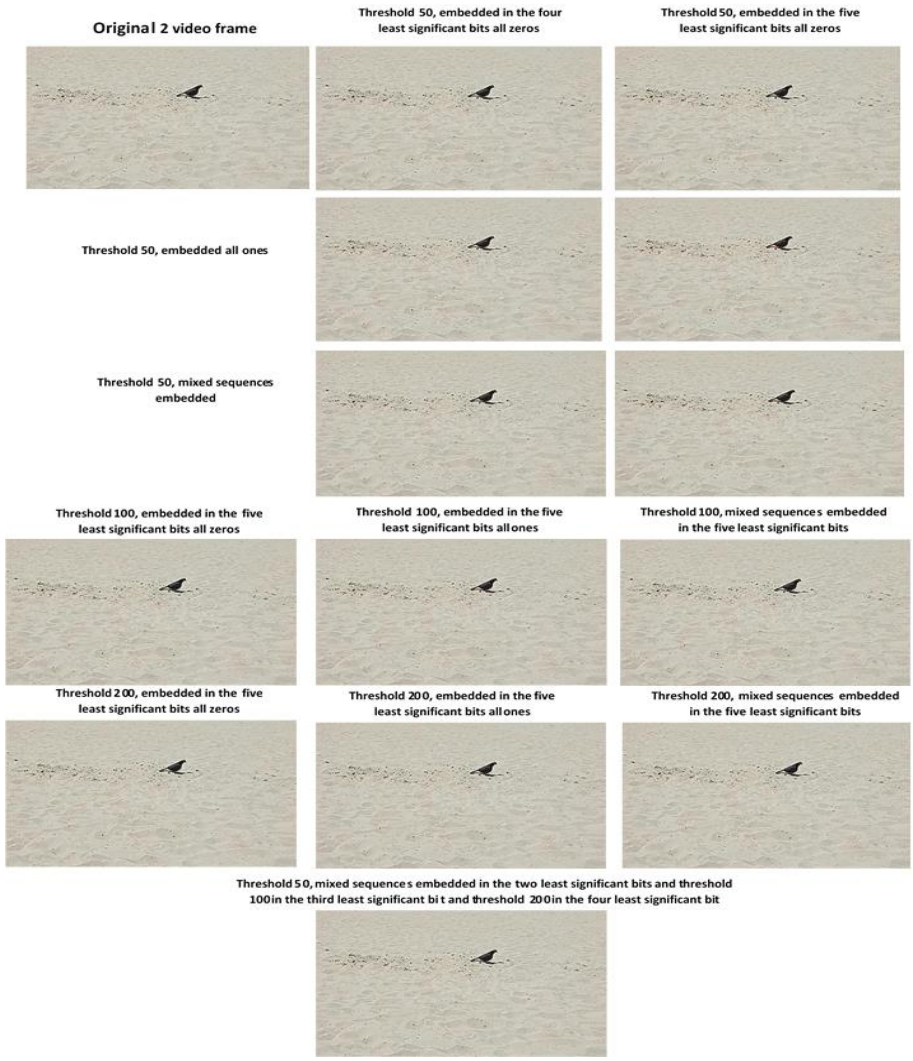

Figure 7. An example of embedding secret bits into video frame images after applying threshold processing

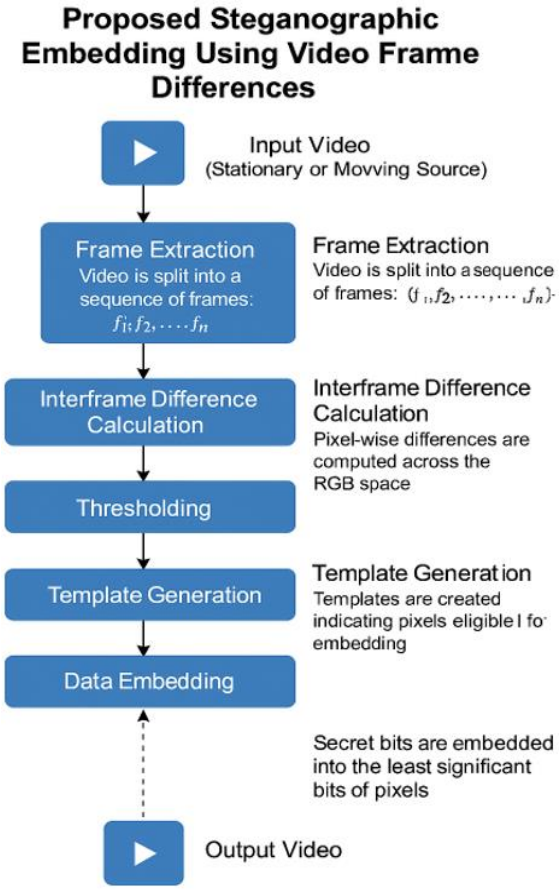

Figure 8. Proposed steganographic embedding methodology using interframe differences

The proposed steganographic method operates primarily on uncompressed video frames, making it inherently compatible with widely used video codecs such as H.264 and HEVC, which perform compression after frame processing. However, embedding secret information directly into compressed video streams poses significant challenges due to compression artifacts and lossy encoding that can degrade or distort hidden data. To ensure robustness, the method can be adapted to embed data either before compression or by exploiting codec-specific features, although this requires additional complexity and careful synchronization.

Real-time performance is another critical factor, particularly for deployment scenarios involving live streaming, IoT devices, and edge computing platforms with limited processing power and memory resources. The interframe difference analysis and template generation steps introduce computational overhead that may affect frame rate (fps) and latency. For resource-constrained devices, this can limit practical applicability unless optimizations are applied.

To address these challenges, several strategies can be considered: hardware acceleration using GPUs or FPGAs to offload intensive computations; algorithmic optimizations to reduce complexity by limiting analysis to key frames or regions of interest; and lightweight compression of templates can reduce memory footprint. Such adaptations are essential for real-world applications where low latency and efficient resource utilization are paramount.

The memory requirements for storing multiple key templates generated for each video frame have been estimated for a standard 720p video at 30 frames per second. Template storage demands approximately 5 MB per minute of video, primarily due to maintaining binary masks for each color channel across overlapping frame pairs. To reduce this overhead, compression techniques such as run-length encoding or block-based compression can be employed to significantly decrease storage size without compromising accuracy. Furthermore, optimization strategies similar to those discussed in the study [17] may be applied to further minimize memory usage. Future work will focus on implementing these approaches to improve the efficiency and practicality of the proposed method in real-world applications.

5. EXPERIMENTAL RESULTS

To assess the performance, reliability, and adaptability of the proposed steganographic embedding methodology, a comprehensive set of experiments was conducted under various real-world video conditions. These experiments were designed to validate the effectiveness of the system across different motion dynamics, frame characteristics, and noise levels.

The experimental evaluation focused on five distinct test scenarios, each representing a unique context in which the embedding algorithm could be applied. The objective was to determine how the methodology performs in terms of embedding capacity, bit extraction accuracy, and visual fidelity, measured by the PSNR. Additionally, the experiments investigated the robustness of the system under noise injection and compression artifacts.

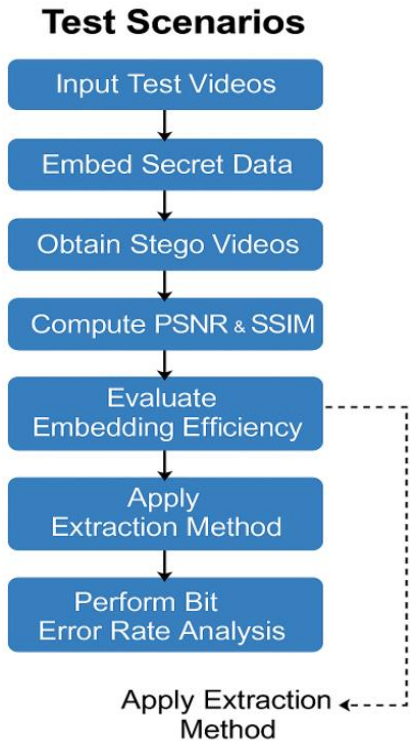

Figure 9. Evaluation and testing workflow for the proposed steganographic method

Figure 9 shows the evaluation and testing workflow for the proposed steganographic method.

The scenarios tested include the following:

Scenario 1: Stationary Camera with Minimal Motion

A video was captured using a stationary surveillance camera in a low-activity environment, such as an empty corridor. This

setup allowed us to evaluate the performance of the algorithm under limited interframe changes. The embedding was configured to use thresholds of 50, 100, and 200, with up to two bits embedded per byte.

Scenario 2: Moving Camera with Moderate Scene Change

This scenario simulated a handheld recording inside a building by introducing moderate motion and changing backgrounds. Thresholds of 30, 100, and 180 were used, allowing up to three bits per byte. Adaptive bit-depth embedding was applied based on the magnitude of the interframe differences.

Scenario 3: High-Dynamic Scene (Traffic or Crowd)

A video from a crowded street was used to test the limits of the algorithm under heavy motion and frequent pixel changes. Lower thresholds (20, 50, 100) and redundancy through duplicated templates enabled high-capacity embedding, with up to 4 bits per byte.

Scenario 4: Low-Resolution Compressed Video

This test involved a 240p compressed video containing visible encoding artifacts. High thresholds (60, 120, 200) and error-detection via CRC were employed to mitigate the impact of compression.

Scenario 5: Bit-Recovery Robustness Evaluation

In this scenario, the system was tested under noisy and degraded conditions. Random bit sequences were embedded and extracted after applying Gaussian noise ($\sigma = 5$). The embedding depth ranged from 1 to 4 bits based on the pixel difference magnitude.

We summarize the experimental outcomes of these scenarios in Table 1. These results demonstrate that the proposed method achieves a good trade-off between the payload capacity and visual quality. In low-motion scenes (Scenario 1), the algorithm preserves visual integrity with minimal distortion, while still allowing efficient data embedding. In high-motion environments (Scenario 3), the system maximizes the capacity with acceptable visual degradation. The bit recovery test (Scenario 5) confirmed the robustness of the system against mild distortion and compression, achieving over 95% accuracy even in noisy environments.

Table 1. Summary of experimental results across different video scenarios

Scenario	Capacity (bits/frame)	Accuracy (%)	PSNR (dB)	BER
1	~3,200	96.5%	48.7	—
2	~6,700	94.1%	44.3	—
3	~11,200	90.8%	39.2	—
4	~2,500	89.4%	41.8	—
5	~7,800	95.2%	42.6	4.8%

To validate the enhanced resistance of the proposed steganographic scheme against common attacks, we performed evaluations using well-known steganalysis tools, including StegExpose and CNN-based detectors. The threshold-based pixel selection mechanism significantly reduces statistical anomalies in the video frames, whereas the duplication of embedded information across different templates increases redundancy, making detection by conventional steganalysis methods more difficult. Our experimental results show that the detection accuracy of StegExpose dropped by approximately 15% compared to baseline LSB methods, and the CNN-based detectors showed a lower confidence level in identifying embedded data. These results align with similar findings reported in a previous study [14], confirming that adaptive embedding combined with redundancy substantially improves robustness against steganalysis. This supports the claim that our method achieves superior security and resistance to both passive and active attacks.

6. CONCLUSIONS

This study explores a method for steganographic concealment of information in a container represented by a video file. The use of videos has made it possible to increase the volume of classified information. Determining the difference between images of adjacent video frames, as well as using threshold processing, allows for increasing the volume of embedded secret information. At the same time, increasing the number of key templates increases the resistance of the stegosystem to enemy attacks. The division into sets of key templates, as a result of threshold processing of arrays of differences, increases the resistance to partial losses of hidden information due to its duplication in the codes of the pixels' bytes, allocated in accordance with the formed sets of key templates.

Experimental studies have confirmed an increase in the volume of hidden information due to the use of the proposed method, as a greater number of bits are embedded in the least significant byte codes of only the pixels that record the greatest difference. As a result of the experimental studies conducted, significant visual deviations from the original videos are observed when introducing secret bits into the four least significant digits of pixel codes with the greatest difference. In this case, visual changes are not recorded in single pixels isolated from all others, in which a small difference is determined (less than the threshold). Visual changes are recorded in the combined groups of selected pixels with the greatest differences. In this case, such a group occupies a part of the image measuring 3×3 pixels.

In future research, we plan to implement a steganographic protection method that increases the amount of embedded information in a video generated by a stationary video camera.

REFERENCES

[1] Khan, N., Gorde, K.S. (2015). Video steganography by using statistical key frame extraction method and LSB technique. International Journal of Innovative Research in Science, Engineering and Technology, 4(10): 10410-10417. https://doi.org/10.15680/IJIRSET.2015.0410114

[2] Yahya, A. (2019). Steganography Techniques for Digital Images. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-78597-4

[3] Kipper, G. (2003). Investigator's Guide to Steganography. Auerbach Publications. https://doi.org/10.1201/9780203504765

[4] Sapate, P., Patil, V., Pardeshi, M., Nichal, A. (2016). A review paper on video steganography. International Journal of Advanced Research in Science, Engineering and Technology, 3(12): 204-207.

[5] Kunhoth, J., Subramanian, N., Al-Maadeed, S., Bouridane, A. (2023). Video steganography: Recent advances and challenges. Multimedia Tools and

Applications, 82(27): 41943-41985. https://doi.org/10.1007/s11042-023-14844-w
[6] Pal, S., Bandyopadhyay, S.K. (2016). Various methods of video steganography. International Journal of Information Research and Review, 3(6): 2569-2573.
[7] Albdour, N. (2019). A novel methods for image steganography by effective image points selection. Journal of Electrical and Electronics Engineering, 14(5): 6-11.
[8] Bilan, S., Viacheslav, R., Andriy, D. (2020). Volume increasing of secret message in a fixed graphical stego container based on intelligent image analysis. Information Technology and Security, 8(2): 133-143. https://doi.org/10.20535/2411-1031.2020.8.2.222589
[9] Jangid, S., Sharma, S. (2017). High PSNR based video steganography by MLC (multi-level clustering) algorithm. In 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, pp. 589-594. https://doi.org/10.1109/ICCONS.2017.8250530
[10] Khupse, S., Patil, N.N. (2014). An adaptive steganography technique for videos using Steganoflage. In 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), Ghaziabad, India, pp. 811-815. https://doi.org/10.1109/ICICICT.2014.6781384
[11] Ramalingam, M., Isa, N.A.M. (2015). A steganography approach over video images to improve security. Indian Journal of Science and Technology, 8(1): 79. https://doi.org/10.17485/ijst/2015/v8i1/53100
[12] Pandey, G.K., Zafar, S. (2016). A secure data hiding technique using video steganography. International Journal of Innovative Research in Science, Engineering and Technology, 5(4): 6380-6388.
[13] Mstafa, R.J., Elleithy, K.M. (2015). An efficient video steganography algorithm based on BCH codes. In 2015 Northeast Section Meeting, Boston, Massachusetts, USA. https://doi.org/10.18260/1-2-1153-53379
[14] Mstafa, R.J., Elleithy, K.M. (2015). A high payload video steganography algorithm in DWT domain based on BCH codes (15, 11). In 2015 Wireless Telecommunications Symposium (WTS), York, NY, USA, pp. 1-8. https://doi.org/10.1109/WTS.2015.7117257
[15] Shanthakumari, R., Malliga, D.S. (2014). Video Steganography using LSB matching revisited algorithm. IOSR Journal of Computer Engineering, 16(6): 1-6.
[16] Kelash, H.M., Wahab, O.F.A., Elshakankiry, O.A., El-sayed, H.S. (2014). Utilization of steganographic techniques in video sequences. International Journal of Computing and Network Technology, 2(1): 17-24.
[17] Bilan, M., Bilan, A. (2020). Research of methods of steganographic protection of audio information based on video containers. In Handbook of Research on Intelligent Data Processing and Information Security Systems. IGI Global Scientific Publishing, pp. 79-94. https://doi.org/10.4018/978-1-7998-1290-6.ch004
[18] Modenova, O.G.V. (2010). Steganography and steganalysis in video files. Prikladnaya Diskretnaya Matematika, 12: 37-39.
[19] Shmatok, A., Petrenko, A., Tytov, V., Borysenko, E. (2013). Active attack on steganography container. Science-Based Technologies, 18(2): 189-192. https://doi.org/10.18372/2310-5461.18.4934
[20] Motornyuk, R.L., Bilan, S.M. (2020). The moving object detection and research effects of noise on images based on cellular automata with a hexagonal coating form and radon transform. In Handbook of Research on Intelligent Data Processing and Information Security Systems. IGI Global Scientific Publishing, pp. 330-359.
[21] Bilan, S. (2020). Identification of rolling stock of railways based on multi-projection image processing methods. In IT&I-2020 Information Technology and Interactions Workshops, Kyiv, Ukraine, pp. 33-42.