Original Scientific Paper https://doi.org/10.24867/GRID-2026-p58

Comparison of Various Least Significant Bit Modifying Methods Used for Image Data Hiding

ABSTRACT

Digital communication has increased the need for techniques that protect information not only by securing its content, but also by concealing the existence of the message itself. This paper focuses on image steganography, with particular emphasis on Least Significant Bit (LSB) methods, which hide data in digital images by modifying the least significant bits of pixel values. The aim of the study is to compare selected LSB based approaches for data hiding in images and to assess their performance under different image processing conditions. The main hypothesis of this paper is that LSB based steganographic methods enable reliable extraction of hidden data from unchanged Portable Network Graphics (PNG) images, while their extraction efficiency decreases significantly when the images are exposed to processing operations that alter pixel values or image structure. At the same time, it is assumed that the use of Reed-Solomon coding can improve the resistance of hidden message when the number of errors remains limited. To test this hypothesis, three methods were examined: 1-bit LSB substitution, 2-bit LSB substitution, and an LSB method combined with Reed-Solomon error correcting coding. The comparison is performed on five selected images in the PNG format, and the stego-images are evaluated before and after different image processing operations, including Gaussian noise, blurring, image downscaling, image copying, and format conversion. The results show that all three methods provide successful extraction of the hidden message when the carrier images remain unchanged. However, the reliability of extraction is reduced after operations that modify the LSB plane or change the structure of the image. Gaussian noise negatively affects all methods, although the Reed-Solomon based approach demonstrates better resistance in cases with limited number of errors. Blurring and downscaling have the most destructive effect, while image copying and format conversion don't significantly influence extraction reliability when the pixel values remain preserved. These findings confirm that LSB steganography is suitable mainly for unchanged or losslessly preserved images, whereas its main limitation is low robustness against subsequent image processing operations..

Patricija Nakić
Aleš Hladnik

Faculty of Natural Sciences and Engineering, Department of Textiles, Graphic Arts and Design
University of Ljubljana, Slovenia

KEY WORDS
Least significant bit, steganography, image data hiding, image processing, Reed-Solomon codes

1. Introduction

The rapid growth of digital communication has considerably increased the amount of private, sensitive and/or confidential information exchanged through digital media. Consequently, methods for protecting said information have become increasingly more important. While cryptography protects content of the information

shared by transforming it into an unreadable form, steganography aims to conceal the very existence of said information by embedding it into an apparently ordinary carrier such as an image, audio file or video (Cox et al., 2008; Fridrich, 2012). Digital images are particularly suitable carriers because of their widespread use, relatively large amount of data and inherent redundancy.

1.1. Least Significant Bit image steganography

One of the simplest approaches to image steganography is the Least Significant Bit (LSB) substitution technique. A digital pixel value can be represented as a sequence of binary digits, with the LSB having the smallest contribution to the numerical value of the pixel. Replacing this bit therefore changes an 8-bit pixel value by at most one intensity level. Such changes are usually difficult to perceive visually, making the LSB-plane suitable for hiding additional information (Altaay et al., 2012).

The method of the basic 1-bit LSB approach is computationally simple and provides relatively high embedding capacity, while also producing only minor visual changes. The capacity can be further increased by modifying two LSB instead of one. However, increasing the number of modified bits also increases the modification of the carrier data and can therefore affect both image quality and statistical detectability.

The main limitation of spatial domain LSB techniques is their relatively low robustness. Because hidden information is stored directly in the pixel values, operations such as filtering, addition of noise, image resizing and lossy compression can modify or even destroy the embedded bit stream. An efficient steganographic method represents a compromise between embedding capacity, visual invisibility and robustness against subsequent image processing procedures (Fridrich, 2012).

1.2. Increasing robustness by error-correcting codes

One possible approach to improving the robustness of LSB steganography is the introduction of error-correcting codes. Reed-Solomon (RS) codes add redundant symbols to the original data before embedding and allow a limited number of errors introduced during transmission or image processing, which is then corrected during decoding. Combining RS coding with conventional LSB embedding therefore preserves the simplicity of spatial domain techniques while providing additional protection against isolated errors in the hidden data (Fontaine & Galand, 2009).

The aim of the study was to compare three LSB-based image steganography methods, which are 1-bit LSB substitution, 2-bit LSB substitution and 1-bit LSB substitution combined with RS error correction. In addition to extraction from unchanged stego-images, robustness was examined after the addition of random noise, blurring, image downscaling, file copying and image format conversion from PNG to BMP as explained in the Methods section. The comparison focuses on successful reconstruction of the hidden message, preservation of the embedded bits and bit error rate (BER).

2. Methods

2.1. Test images and data preparation

Five standard images from the USC-SIPI Image Database (USC-SIPI Image Database, n.d.) were selected to represent different image dimensions and levels of visual complexity. Images Airplane (labeled as A), Peppers (B) and Mandrill (C) were originally 512 x 512 pixel, 24-bit TIFF images. Airplane contains comparatively large smooth regions, Peppers represents an intermediate level of edges and colour transitions, whereas Mandrill contains a large amount of fine texture and local variation. In addition, the 1024 x 1024 pixel, 8-bit grayscale Airport image (D) and the 256 x 256 pixel, 24-bit Tree image (E) were used.

Before embedding, the images were converted to the Portable Network Graphics (PNG) format in order to avoid changes of pixel values caused by lossy compression. The images were standardized to 8 bits per channel using Adobe Photoshop. During program execution, the carrier images were additionally converted to RGB mode, resulting in three 8-bit colour channel values for each pixel used by the embedding algorithms.

The same secret message: “To je skrivna informacija.” (en. “This is the secret information.”) was used for all experiments. The text was encoded using UTF-8 before converting it into its binary representation. For the conventional 1-bit and 2-bit LSB methods, a 32-bit header containing the length of the message and a 32-bit CRC32 checksum were added to the data. The header allowed the extraction algorithm to determine the required number of bits, whereas CRC32 was used to verify the integrity of the reconstructed message.

2.2. Implementation of the LSB methods

All embedding and extraction algorithms were implemented in Python (version 3.13) and executed in its IDLE environment. The Pillow library was used for image handling, NumPy for array manipulation and bitwise operations and the Python zlib module for CRC32 calculation. RS coding was implemented using the reedsolo package.

For 1-bit LSB substitution, the RGB image was converted into a NumPy array and flattened into a one-dimensional sequence of colour channel values. The header, that is the CRC32 value and UTF-8 message were converted into binary form and joined into a single payload. The least significant bit of each successive channel value was cleared using a bitwise & (and) operation and replaced by the corresponding payload bit using bitwise | (or) operation. So one message bit was stored in each colour channel value. For 2-bit LSB substitution the general procedure remained unchanged, but the payload was divided into two bit symbols. The 32 bit header and CRC32 checksum were retained, while twice as many payload bits could be stored in each image channel value.

For the RS LSB method the UTF-8 encoded message was first protected using RS coding before 1-bit LSB embedding. 16 RS correction symbols were added using RSCodec(16). A 32-bit header containing the length of the RS coded message was placed before the encoded payload, after which the complete bit stream was embedded into the least significant bits of the image channel values. During extraction the header was first recovered, followed by the specified number of encoded bits. The extracted binary data was converted back to bytes and passed to the RS decoder, which attempted to correct errors before reconstructing the original message

2.3. Performance indicators

Performance of the three steganographic methods was evaluated using several complementary indicators (metrics) describing the correctness and robustness of message extraction. The primary quantitative metric was BER (Bit Error Rate) (Malathi & Gireesh Kumar, 2025). BER is calculated as

$$BER = \frac{N_{error}}{N_{total}} \quad (1)$$

where N_error is the number of incorrectly recovered message bits and N_total is the total number of bits in the original message. A BER of 0 corresponds to error free recovery, while increasing BER values indicates a progressively larger proportion of incorrectly extracted bits.

The percentage of preserved bits was derived from BER according to:

$$P_{preserved} = (1 - BER) \times 100\% \quad (2)$$

100% preserved bits corresponds to BER = 0, while lower percentages indicate increasing degradation of the embedded message.
Extraction was additionally evaluated according to the message matching, classified as complete, partial or absent. For the 1-bit and 2-bit LSB methods, the validity of the extracted header and the result of the CRC32 integrity check were also recorded. The header stored information required for determining the length of the embedded message and managed to enable the extraction algorithm to identify the relevant part of the bit stream.

The integrity of the extracted message was verified using CRC32. The CRC32 value was stored together with the hidden data and subsequently compared with the value calculated from the extracted message. The use of a CRC field as part of the data structure is consistent with the CRC32 mechanism described in the GZIP file-format specification (Deutsch, 1996).

In the present implementation, CRC32 calculation was performed using functionality provided by the Python zlib library (Python documentation, 2026). CRC32 was used solely as an integrity check and did not provide error correction. Therefore, failure of the CRC32 check was treated as evidence that the extracted data or the stored checksum had been altered.

For the RS-LSB method, RS decoding was included as an additional performance indicator. The combination of LSB image steganography and RS coding has been specifically proposed as a means of increasing the robustness of hidden-data recovery (Raiyan & Kabir, 2025).

RS coding was used because its additional redundancy can improve steganographic schemes by allowing corrupted encoded data to be reconstructed under suitable error conditions (Fontaine & Galand, 2009).

The correction capability is nevertheless limited. RS decoding can recover the encoded information only when the number and distribution of errors or erasures remain within the correction capability of the applied code (Jing et al., 2022). In the implemented procedure, the extracted bit stream was converted into RS coded bytes and passed to the decoder, which attempted to correct errors before reconstructing the original message.

For this reason, successful final message recovery was evaluated separately from the raw BER in the RS-LSB method. A nonzero BER in the encoded data does not necessarily imply unsuccessful extraction, since the RS decoder may still reconstruct the original message if the introduced errors are correctable. Conversely, failure of RS decoding was treated as unsuccessful extraction.

When the extraction algorithm could not retrieve enough bits from the modified image, extraction was classified as unsuccessful and BER and the percentage of preserved bits were not reported.

3. Results

3.1. Visual comparison of original and stego-images

No visually noticeable differences were observed between the original images and the corresponding stego-images after message embedding with any of the three methods. Across all five test images, the embedded data did not produce visible changes that could be identified by direct visual inspection.

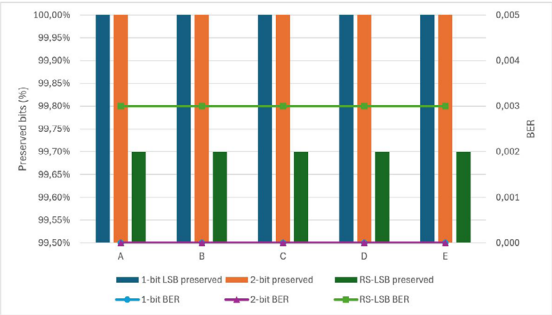
3.2. Extraction from unmodified stego-images

For the unmodified stego-images, all three methods achieved complete message recovery for all five test images. For the 1-bit and 2-bit LSB methods, the extracted headers were correct, and CRC32 verification was successful in every case. For the RS-LSB method, RS decoding was successful for all five images.

The percentage of preserved message bits was 100% and BER was 0.0 for all images and all three methods. Complete agreement between the extracted and original message was therefore obtained for all images that were not subject to an image processing operation.

3.3. Effect of noise

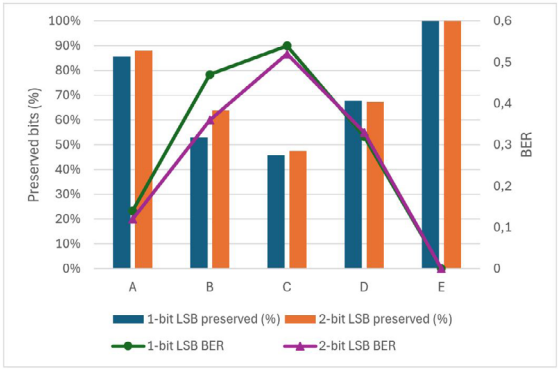
After Gaussian noise with a value of 0.005 was applied, all three methods succeeded in recovering the complete hidden message from all five images. The percentage of preserved bits remained 100%, and BER was 0.0 for every combination of image and method. For both the 1-bit and 2-bit LSB methods, the extracted header was correct, and CRC32 verification was successful for all five images. RS decoding was also successful for every image processed using the RS-LSB method. Thus, under this low noise level no differences in final message recovery were detected among the three methods studied.
At a higher Gaussian noise level of 0.01, complete message matching was again obtained for all five images and all three methods (Fig. 1). However, differences were observed for the individual performance indicators. For the 1-bit and 2-bit LSB methods, the extracted header remained correct for all images. The preserved bit percentage was 100% and BER remained 0.0. CRC32 verification, however, failed in all five cases for both methods. For the RS-LSB method, RS decoding was successful for all five images. The percentage of preserved bits was 99.7%, corresponding to a BER of 0.003. Despite this non-zero BER value, the final decoded message completely matched the original message in all cases.



» **Figure 1:** *Percentage of preserved bits after adding Gaussian noise (noise 0.01)*

3.4. Effect of blurring

Blurring with a radius of 0.4 produced a wide range of extraction results for the 1-bit and 2-bit LSB methods (Fig. 2). For images A-D, no method was able to achieve complete message recovery.

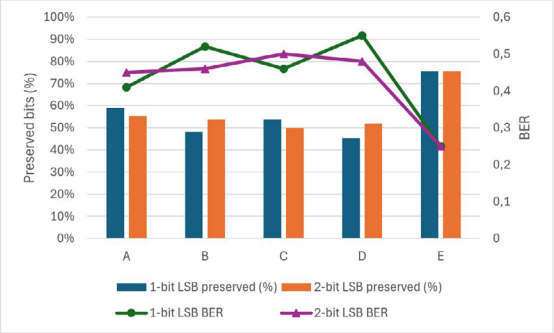


» **Figure 2:** *Percentage of preserved bits after image blurring (blur 0.4)*

For the 1-bit LSB method, the preserved bit percentage ranged from 45.7% to 85.6% for images A-D. The corresponding BER values ranged from 0.14 to 0.54. For the 2-bit LSB method, the preserved bit percentage for images A-D ranged from 47.6% to 88.0%, while BER ranged from 0.12 to 0.52. Image E differed from the other four images. For both the 1-bit and 2-bit LSB methods, the extracted header was incorrect, but complete message matching was obtained when the known message length was used during evaluation. The preserved-bit percentage was 100% and BER was 0.0 with both methods. For the RS-LSB method, extraction could not be performed for any of the five blurred images. The program returned an error indicating that the image did not contain enough bits for message extraction.

3.5. Effect of image downscaling

After image downscaling by a factor of 0.95, complete message recovery was not achieved for any of the three methods. For both 1-bit and 2-bit LSB methods, the extracted headers were incorrect for all five images and CRC32 verification failed in every case.



» **Figure 3:** *Percentage of preserved bits after image downscaling (downscale 0.95)*

For the 1-bit LSB method, the percentage of preserved bits ranged from 45.2% to 75.5%. The corresponding BER values ranged from 0.25 to 0.55. For the 2-bit LSB method, preserved-bit percentages ranged from 50.0% to 75.5%, with BER values between 0.25 and 0.5. For the RS-LSB method, extraction could not be completed for any of the five downscaled images.

3.6. Effect of image copying

After image copying, identical extraction results were obtained for all five test images and all three methods. The complete message was recovered in every case, the preserved-bit percentage was 100%, and BER was 0.0. For the 1-bit and 2-bit LSB methods, the extracted headers were correct, and CRC32 verification was successful for all images. For the RS-LSB method, RS decoding was successful in all five cases.

3.7. Effect of image format conversion

After conversion from PNG to BMP, complete message recovery was obtained for all five test images and all three methods. The preserved bit percentage remained 100% and BER was 0.0 in every case. For both the 1-bit and 2-bit LSB methods, the extracted headers were correct, and CRC32 verification was successful for all five images. For RS-LSB, RS decoding was successful in every case.

The measured extraction results after PNG to BMP conversion were therefore identical to those obtained for the unmodified stego-images as well as for copied images.

4. Discussion

The results demonstrate that the investigated LSB approaches are highly reliable as long as the pixel values and their ordering remain unchanged. The perfect extraction obtained from unmodified images, copied images, and images converted from PNG to BMP is consistent with the spatial nature of LSB embedding: when the underlying RGB values are preserved, the embedded least significant bits remain available to the extraction algorithm. Importantly, the format conversion result applies specifically to the lossless PNG to BMP conversion tested here and should not be generalized to lossy formats such as JPEG.

Noise produced a different type of disturbance. At the lower tested level, no measurable message errors occurred. At a Gaussian noise level of 0.01, the distinction between data integrity and final message recovery became more apparent. In the 1-bit and 2-bit methods, the message bits themselves remained unchanged, as reflected by BER = 0, although CRC32 verification failed. This combination is consistent with corruption of the separately embedded CRC32 value rather than the message payload itself. Consequently, CRC32 should not be used as the sole measure of extraction performance but should be interpreted together with BER and direct message matching.

The RS-LSB method behaved differently at the same noise level. Although a small number of bit errors was detected (BER = 0.003), RS decoding recovered the original message perfectly. Here we can see the clearest advantage of RS-LSB observed in the experiment: error correcting redundancy can compensate for a limited number of bit errors. The advantage is therefore not necessarily reflected in a lower BER, but in the ability to reconstruct the correct final information despite those errors.

Blurring and downscaling represented much more severe disturbances. Unlike isolated bit modifications, both operations alter pixel values systematically and, in the case of downscaling, also change the spatial organization and number of pixels. Since the investigated methods depend on reading specific least significant bits in the expected sequence, these transformations disrupt the structure required for extraction.
This also explains why RS coding provided no practical benefit under these conditions. Error correction is useful when the encoded bit stream can still be retrieved and the number of errors remains within the correction capability of the code, but it cannot recover information when the bit stream structure itself can no longer be read correctly.

The apparently successful recovery from blurred image E requires particular caution. Although the message was recovered with BER = 0, its header was incorrect. During evaluation, the actual message length was known in advance, allowing the correct section of the extracted bit stream to be selected independently of the corrupted header. Under normal extraction conditions, in which the message length is unknown, such recovery would not be reliable. The result for image E should therefore not be interpreted as evidence of genuine robustness to blurring.

No clear robustness advantage of the 2-bit LSB method over the 1-bit method was observed. Its primary advantage is increased embedding capacity rather than improved resistance to image processing. RS-LSB provided an advantage specifically when the disturbance produced a limited number of correctable errors, but it retained the fundamental vulnerability of spatial LSB embedding to transformations that modify pixel values or their arrangement. Overall, the results are related to two types of post-processing: operations that preserve the pixel data, for which all three methods remained reliable, and operations that modify the pixel structure, for which extraction became unreliable.

5. Conclusions

All three investigated LSB methods provided reliable message extraction when the carrier image remained unchanged or underwent operations that preserved its pixel values. Under these conditions, the 1-bit LSB, 2-bit LSB and RS-LSB methods all achieved error-free message recovery.

RS coding improved robustness when a limited number of bit errors was introduced, as demonstrated under the higher tested noise level, where the original message was fully recovered despite a nonzero BER. However, this protection was ineffective against blurring and downscaling because these operations disrupted the pixel values and/or spatial organization required for correct reconstruction of the embedded bit stream.

Accordingly, none of the three methods can be regarded as universally superior. The 1-bit LSB method offers the simplest implementation, the 2-bit method provides greater embedding capacity, and RS-LSB adds tolerance to a limited number of bit errors. The appropriate choice therefore depends primarily on the expected post-processing of the carrier image and on the required balance between embedding capacity and robustness. The findings further emphasize that the demonstrated resistance to format conversion applies only to the tested lossless PNG to BMP conversion and cannot be extended to lossy image formats without additional testing.

6. References

Altaay, A.A.J., Sahib, S.B. and Zamani, M. (2012) 'An Introduction to Image Steganography Techniques'. In: *Proceedings of the International Conference on Advanced Computer Science Applications and Technologies*, ACSAT 2012, 26–28 November 2012, Kuala Lumpur, Malaysia. IEEE Computer Society, pp. 122–126.

Cox, I.J., Miller, M.L., Bloom, J.A., Fridrich, J. and Kalker, T. (2008) *Digital Watermarking and Steganography*. 2nd edition. Burlington: Elsevier Inc.

Deutsch, P. (1996) GZIP file format specification version 4.3. IETF Datatracker. Available from: https://datatracker.ietf.org/doc/html/rfc1952 [Accessed 23 April 2026].

Fontaine, C. and Galand, F. (2009) 'How Reed-Solomon Codes Can Improve Steganographic Schemes', *EURASIP Journal on Information Security*, 2009(2), pp. 1–10. DOI: 10.1155/2009/274845

Fridrich, J. (2012) *Steganography in Digital Media: Principles, Algorithms and Applications*. 1st edition. Cambridge: Cambridge University Press.

Jing, Z., Choi, H. and Song, H.Y. (2022) 'Decoding Process of RS Codes with Errors and Erasures: An Overview', *The Journal of Korean Institute of Communications and Information Sciences*, 47(11), pp. 1747–1754. DOI: 10.7840/kics.2022.47.11.1747

Malathi, P. and Gireesh Kumar, T. (2025) 'Deep Steganographic Approach for Reliable Data Hiding Using Convolutional Neural Networks and Adaptive Loss Optimization', *Scientific Reports*, 15(1). DOI: 10.1038/s41598-025-26867-4

Python Software Foundation. (no date) *Python documentation – zlib — Compression compatible with gzip*. Available from: https://docs.python.org/3/library/zlib.html [Accessed 24 April 2026]

Raiyan, S.R. and Kabir, M.H. (2025) 'SCreedSolo: A Secure and Robust LSB Image Steganography Framework with Randomized Symmetric Encryption and Reed Solomon Coding', *Pattern Recognition and Computer Vision*, ACPR. DOI: 10.48550/arXiv.2503.12368

University of Southern California, Signal and Image Processing Institute. (no date) *USC-SIPI Image Database*: Volume 3: Miscellaneous. Available from: https://sipi.usc.edu/database/database.php?volume=misc [Accessed 23 April 2026]



© 2026 Authors. Published by the University of Novi Sad, Faculty of Technical Sciences, Department of Graphic Engineering and Design. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license 4.0 Serbia (https://creativecommons.org/licenses/by/4.0/deed.en).