Reversible Data Hiding in Encrypted Images with Pixel Prediction and ERLE Compression

Remigiusz Martyniak[1[0009-0005-9337-1082]] and Mariusz Dzwonkowski[1,2[0000-0003-3580-7448]]

[1] Department of Teleinformation Networks, Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, Gabriela Narutowicza 11/12, 80-233 Gdańsk, Poland
[2] Department of Radiology Informatics and Statistics, Faculty of Health Sciences, Medical University of Gdansk, Tuwima 15, 80-210 Gdańsk, Poland

remigiusz.martyniak@pg.edu.pl

Abstract. Reversible Data Hiding in Encrypted Images (RDHEI) is a technique that enables the embedding of additional information into encrypted carrier images, facilitating data extraction and exact restoration of the original image upon decryption. In this paper, enhancements to an RDHEI algorithm utilizing a block-wise pixel value prediction scheme have been analyzed and proposed. To better exploit spatial pixel correlations, seven additional prediction models have been introduced, along with the identification of reference pixels to improve the variable block-wise reconstruction of the carrier data. Furthermore, the integration of Huffman coding within the RDHEI scheme has been evaluated and compared with Extended Run-Length Encoding. Benchmark results against other RDHEI methods from the literature are presented at the end of the paper.

Keywords: Encrypted images, ERLE, lossless scheme, RDHEI.

1 Introduction

The evolution of data security has become a critical aspect in contemporary digital communication. Digital images can function not only as a medium for visual information but also as vessels for securely embedded data, accessible exclusively through designated decryption keys. However, most data embedding methods introduce permanent losses in the utilized carrier upon extracting the embedded data, making it impossible to restore its original form. In fields such as medicine, forensics, or military systems, such losses are deemed unacceptable. Hence, a specialized data embedding method known as Reversible Data Hiding (RDH) was developed, ensuring the recovery of embedded data and the data carrier without any information loss. To achieve a high embedding capacity, various embedding techniques have been employed. These include approaches based on histogram shifting [1,2], difference expansion [3], and pixel value ordering [4].

Over the years, a growing number of RDH algorithms in encrypted domain (RDHEI) have been developed, providing the capability for lossless recovery of carri-

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

er data and embedded data while ensuring security. Advancements in the RDHEI field can be particularly crucial in medical imaging, e.g. when embedding patient information and metadata into medical images such as X-rays, MRIs, or CT scans, ensuring that the authenticity and integrity of the images can be verified without compromising their quality and usability for diagnosis [5]. Several notable methods have contributed to the development of Reversible Data Hiding in Encrypted Images. Yi and Zhou [6] introduced a Parametric Binary Tree Labeling (PBTL) approach that divides and labels pixels for data embedding, using parameters stored within the encrypted image. The method involves a detailed process of labeling, block permutation, and pixel restoration to recover both the image and embedded data.

The Tang method [7] focuses on block-wise data hiding, utilizing a logistic map for encryption and compression to create embedding space. This method requires auxiliary information for successful data extraction and image recovery. In the Yin method [8], a median edge detector is employed to generate a label map, which plays a crucial role in determining the data embedding capacity. The embedding process involves label maps, Huffman coding, and multiple MSB substitutions, offering a balance between data capacity and security.

Mohammadi, Nakhkash, and Akhaee [9] introduced a high-capacity RDHEI technique that uses a local difference predictor. This approach increases embedding capacity by locally predicting pixel differences, allowing for the concealment of larger amounts of data while ensuring the complete recovery of the original image. Huiqi Zhang, Lin Li, and Qingyan Li [10] introduced a reversible data embedding algorithm based on block-wise multi-prediction, where the original image is divided into blocks of 8×8 px. The value of each pixel in every block is computed based on one or more adjacent pixels with known values.

This paper builds upon an existing RDHEI approach based on specific block-wise pixel value prediction models [10]. The considered approach efficiently embeds data within encrypted images, ensuring both high-capacity data embedding and the integrity of the original image. The contributions presented in this paper include:

- Expanding the block-wise pixel prediction scheme by incorporating seven additional prediction models to better exploit spatial pixel correlations;
- Identifying reference pixels to aid in the reconstruction of the carrier data and formulating the structure of the auxiliary information binary sequence to ensure independent extraction of the embedded data and reconstruction of the cover image;
- Conducting a detailed comparative analysis of the RDHEI scheme implemented with Huffman coding and fine-tuned ERLE compression for various block sizes of the segmented cover image.

2 Considered RDHEI scheme

The RDHEI algorithm comprises three stages:

- Preprocessing (content owner's side).
- Data hiding (data hider's side).

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

– Data extraction and image recovery (receiver's side).

Preprocessing consists of three sequentially performed operations on the original image: block-wise pixel prediction, compression, and encryption with key K_e. Next, the embedding entity (i.e., data hider) embeds secret data using the data hiding key K_h in the previously encrypted image. On the receiver's side, the embedded data is recovered, and the original image is reconstructed using the corresponding data hiding key and the image encryption key.

2.1 Preprocessing

The content owner conducts operations on the carrier image, allowing the data hider to embed data on their end and facilitating the receiver in recovering the embedded data and reconstructing the original image. Additionally, the outcome of the preprocessing is the acquisition of auxiliary information (AI), which is a binary sequence containing the necessary information for embedding and data recovery. AI is passed to the data hider within an encrypted placeholder of the carrier image.

Block-wise pixel prediction. The carrier image (i.e., algorithm's input) is divided into smaller blocks of size $b \times b$. Pixel prediction is performed on these prepared blocks. Each pixel value is predicted based on one or more neighboring pixels, whose values are already known. The value of pixel (P) is predicted based on neighboring pixels: X_{ul} (upper-left), X_u (upper), X_{ur} (upper-right), X_l (left).

In this work, 23 prediction models were used for pixel prediction. The initial 16 models were integrated based on the methodology described in [9], while the additional models introduced in this paper are presented in Table 1. This approach leaves room for further improvements, e.g. by increasing the number of prediction models even further to a total of 32, while still maintaining their binary representation at 5 bits.

Table 1. Prediction models added to the block-wise pixel prediction step.

Model	Predicted value of a pixel	Model	Predicted value of a pixel
17	$\text{Round}((X_{ul} + X_{ur}) / 2)$	21	$\text{Max}(X_u, X_l, X_{ul}, X_{ur})$
18	$\text{Round}((X_{ul} + X_{ur} + X_u) / 3)$	22	$\text{Min}(X_u, X_l, X_{ul}, X_{ur})$
19	$\text{Round}((X_l + X_{ul} + X_u) / 3)$	23	$\text{Max}(X_{ul}, X_{ur})$, if $X_u \leq \min(X_{ul}, X_{ur})$
20	$\text{Round}((X_l + X_u + X_{ur}) / 3)$		$\text{Min}(X_{ul}, X_{ur})$, if $X_u \leq \max(X_{ul}, X_{ur})$

Reference pixels are essential for the accurate reconstruction of the image on the recipient's side. Due to their significant importance, they constitute a part of the AI sequence. For each block, prediction is carried out using each of the 23 models. To identify the model with the smallest error, the SAD function is employed [10]. The model characterized by a SAD function value close to zero for a specific block is utilized for predicting pixel values within that block. After conducting predictions for

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

each block, the difference between the original image and the image obtained through prediction is calculated, thus creating an error map. Additionally, a map of models used to predict pixel values within each block is generated. Due to the presence of negative values in the error map, it is essential to apply suitable encoding for their representation. For this purpose, sign-magnitude encoding is employed.

Compression and encryption. To maximize embedding capacity, the error map is compressed using Huffman coding and Extended Run-Length Encoding (ERLE). ERLE, an advanced form of Run-Length Encoding (RLE) proposed by Chen and Chang [11], optimizes lossless compression for consecutive symbol sequences, common in image data. RLE compresses runs by recording the symbol and its run length, requiring additional markers for run transitions in RDHEI algorithms. ERLE enhances this with fixed-length codewords (prefix 0 plus L_{fix} bits) for short runs (<4 symbols) and variable-length codewords for longer runs (≥4 symbols), featuring a prefix (L_{pre} bits: $L_{\text{pre}} - 1$ ones, then 0), a length symbol encoding value of $L - 2^{L_{\text{pre}}}$ (L being the length of the run), and a tail bit (indicating symbol of the run). After error map compression, the AI sequence—which stores the data for image reconstruction—is formulated. It comprises reference pixel bits (first row and column of the carrier), prediction models bits, sign-magnitude encoded module bits, and the compressed error map. This AI data is placed at the beginning of a placeholder sequence of length $m \cdot n \cdot 8$, where m and n are the width and height of the image, respectively. The remaining space in the placeholder is designated for data embedding. The placeholder is encrypted using a pseudo-random bit sequence (key K_e) of equal length via bitwise XOR operation, yielding an encrypted sequence for the data hider.

2.2 Data hiding

The empty space in the encrypted placeholder is designated for the data hider, who can embed additional, secret data into it. To help identify the starting point for the data embedding process, the length of the data hiding key K_h, used for encrypting the additional information, is associated with the length of the compressed error map. Thus, knowing the key K_h, the data hider can locate the end of AI in the encrypted placeholder and start embedding additional information without interfering with previously embedded auxiliary data. To obtain the encrypted secret sequence, an XOR operation is performed between the bits of additional information and the key K_h. The encrypted sequence, obtained in this manner, is then embedded in the encrypted placeholder immediately after the encrypted AI sequence.

2.3 Data extraction and image recovery

After receiving the encrypted message with embedded data, the recipient is able to extract the necessary information to reconstruct the original carrier image and the embedded data according to the image encryption key K_e and the data hiding key K_h.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

The first step is to extract the embedded AI sequence from the encrypted message. The length of this sequence is calculated based on the length of the received information and the length of the key K_h. First, the size of the original image (assuming $m \times m$ for simplicity) is calculated according to the formula (1): $m = \sqrt{\left(n_p / 8\right)}$, where n_p is the length of the encrypted placeholder. The number of bits corresponding to the stored reference pixels is calculated according to formula (2): $R = 8 \cdot \left(2m - 1\right)$. Next, the number of bits of the prediction models map is calculated (3): $P = \left(m / b\right)^2 \cdot 5$, where b is the block size chosen during the preprocessing.

The number of sign bits generated during sign-magnitude encoding is equal to m^2. In this way, the bit number from which the compressed error map begins in the AI sequence is calculated as the sum of bits from reference pixels, bits of the prediction models map, and bits generated during sign-magnitude encoding, increased by 1.

The length of the data hiding key K_h is equal to the length of the compressed error map. Starting from the initial bit of the compressed error map, a sequence of l_{kh} bits is extracted, where l_{kh} is the length of the key K_h. In this way, the encrypted AI sequence is obtained, which is then decrypted using the encryption key K_e. The remaining part in the encrypted message contains information embedded by the data hider, which is extracted and decrypted using the key K_h.

In order to recover the carrier, the obtained error map is decompressed and reshaped into a matrix using sign bits. This matrix, representing the reconstructed image, is initiated with reference pixel values placed initially in the first row and then in the first column. The order of bits representing reference pixel values in the AI sequence corresponds to the order of placing them in the reconstructed image. The process of image reconstruction occurs pixel by pixel, starting from the pixel in the second row and second column. Using the prediction models map extracted from the AI sequence, the pixel value is predicted, and then added to its corresponding element from the error map matrix. This operation is performed sequentially (pixel by pixel).

3 Results

This section presents tests of the implemented RDHEI algorithm, conducted in MATLAB R2023b on Windows 11 with a 12th Gen Intel® Core™ i7-12700 processor. Five standard 512 × 512 greyscale images (Baboon, Lake, Plane, Peppers, and Boat) were used [12], along with images from the BOSSbase 1.01 [13] and BOWS2 [14] datasets.

3.1 Security and image quality analysis

To assess the effectiveness of the encryption, commonly used metrics were employed: analyses of histograms of carrier and encrypted images, the Number of Pixel Change Rate (NPCR) indicator, and the Unified Average Changing Intensity (UACI) [10]. Histograms of encrypted images differ from those of original images and do not ex-

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

hibit any significant features characteristic of carriers (Figure 1). The NPCR coefficient in all cases reached nearly 100%, which implies that nearly all pixels undergo changes compared to the original image. The UACI result averages 15% for both compression methods, which indicates a satisfactory robustness of the encrypted data against differential attacks.

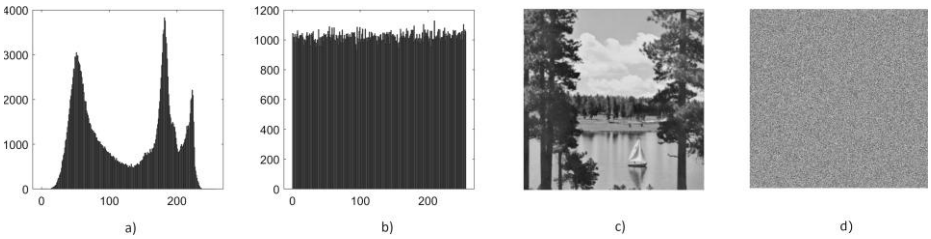

Fig. 1. Histograms of image pixel values distribution: (a) before and (b) after encryption, as well as (c) original and (d) encrypted Lake image.

After decryption and data extraction, reconstructed images were evaluated using Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). All images achieved infinite PSNR and SSIM of 1, confirming lossless reconstruction.

3.2 Performance analysis

The discussed algorithm was examined for its performance. In particular, the impact of the applied block size b and compression method on embedding efficiency was investigated. Moreover, the best average L_{fix} value for ERLE compression was identified.

ERLE performance test. The impact of the L_{fix} value (ranging from 4 to 20) on ERLE compression efficiency was tested for two block sizes (Figure 2). For BOSSbase and BOWS2, the optimal value averaged around 8, as it was selected for approximately 2000 images depending on dataset and block size. For the standard images, with varying block sizes, the optimal values of L_{fix} were essentially the same for every tested block size, but larger than those for the majority of images from the datasets.

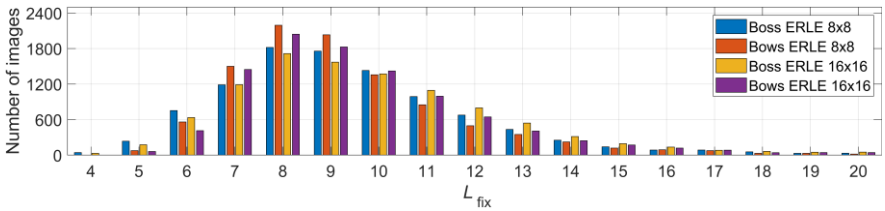

Fig. 2. Optimal L_{fix} values for BOSSbase and BOWS2 images and two fixed block sizes.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

Effectiveness of data embedding. Embedding performance varied with compression method and block size. Huffman coding provided better average embedding rates (by 0.36 bpp for 8 × 8, and by 0.34 bpp for 16 × 16) across both datasets. However, ERLE achieved higher maximum embedding rates, outperforming Huffman in 574 images (difference >0.1 bpp), and reaching improvements of over 0.54 bpp in 93 cases. Compared to existing methods (Table 2), ERLE compression provided better embedding results. Although slightly below [8] in average embedding rate, ERLE achieved higher maximum rates (by 13.75% for BOSSbase and 9.24% for BOWS2). Prediction models 1, 4, 7, 16, 21, and 22 were the most frequently used, demonstrating consistent performance across the datasets. Notably, two of the prediction models introduced in this study were among the top performers.

Table 2. Comparison of the embedding rate (bpp) for the discussed RDHEI algorithm with different block sizes and compression methods against selected state-of-the-art methods.

	This work *b*: 8 × 8		This work *b*: 16 × 16							
Images	ERLE	Huff	ERLE	Huff	Method [5]	Method [6]	Method [7]	Method [8]	Method [9]	Method [11]
Baboon	1.079	1.748	1.193	1.828	-	-	0.641	1.204	1.04	-
Lake	1.821	2.433	1.946	2.543	-	-	1.468	-	-	1.944
Airplane	2.781	3.257	2.861	3.321	-	2.219	2.281	3.067	2.75	2.340
Peppers	2.347	3.002	2.397	3.049	-	-	1.798	-	-	1.879
Boat	2.381	2.793	2.455	2.839	-	-	1.519	-	-	-
BOSS (max)	6.705	5.804	6.765	5.864	-	-	-	5.921	-	-
BOSS (aver.)	3.113	3.444	3.216	3.523	2.732	2.026	-	3.389	-	2.435
BOWS2 (max)	6.138	5.519	6.197	5.575	-	-	4.881	5.646	6.11	-
BOWS2 (aver.)	2.961	3.324	3.082	3.419	2.547	-	2.425	3.282	2.9	-

4 Conclusions

In this study, enhancements to an RDHEI algorithm based on block-wise pixel prediction were presented, focusing on increasing embedding capacity through fine-tuned ERLE compression. Results indicate that, on average, ERLE slightly underperforms compared to Huffman coding but excels for images containing repetitive pixel values. Huffman coding remains preferable for images with diverse pixel distributions, such as those in BOSSbase and BOWS2.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13

Therefore, the choice of compression should reflect image characteristics, with ERLE being likely advantageous for medical (DICOM) images exhibiting large uniform regions. Future research will explore ERLE's application in medical imaging to further optimize the embedding process.

Acknowledgments. This study was funded by the Faculty of Electronics, Telecommunications, and Informatics of Gdansk University of Technology, and by the research subsidy from the Polish Ministry of Science and Higher Education. The authors would like to express their gratitude to Mateusz Myszk for his conducted research, which laid the foundation for the discussed approach.

Disclosure of Interests. The authors declare no conflict of interest.

References

1. Ni, Z., Shi, Y.Q., Ansari, N., Su, W.: Reversible data hiding. IEEE Trans. Circuits Syst. Video Technol. **16**(3), 354–362 (2006)
2. Li, X., Li, B., Yang, B., Zeng, T.: General framework to histogram shifting-based reversible data hiding. IEEE Trans. Image Process. **22**(6), 2181–2191 (2013)
3. Tian, J.: Reversible watermarking by difference expansion. In: Proc. Workshop Multimedia Security: Authentication, Secrecy, and Steganalysis, pp. 19–22. (2002)
4. Li, X., Li, J., Li, B., Yang, B.: High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion. Signal Process. **93**(1), 198–205 (2013)
5. Dzwonkowski, M., Czaplewski, B.: Reversible data hiding in encrypted DICOM images using sorted binary sequences of pixels. Signal Process. 199, 1–14 (2022)
6. Yi, S., Zhou, Y.: Separable and reversible data hiding in encrypted images using parametric binary tree labeling. IEEE Trans. Multimedia **21**(1), 51–64 (2019)
7. Tang, Z., Xu, S., Yao, H., Qin, C., Zhang, X.: Reversible data hiding with differential compression in encrypted image. Multimed. Tools Appl. 78, 9691–9715 (2019)
8. Yin, Z., Xiang, Y., Zhang, X.: Reversible data hiding in encrypted images based on multi MSB prediction and Huffman coding. IEEE Trans. Multimedia **22**(4), 874–884 (2020)
9. Mohammadi, A., Nakhkash, M., Akhaee, M.A.: A high-capacity reversible data hiding in encrypted images employing local difference predictor. IEEE Trans. Circuits Syst. Video Technol. **30**(8), 2366–2376 (2020)
10. Zhang, H., Li, L., Li, Q.: Reversible data hiding in encrypted images based on block-wise multi-predictor. IEEE Access 9, 61943–61954 (2021)
11. Chen, K., Chang, C.C.: High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement. J. Vis. Commun. Image Represent. 58, 334–344 (2019)
12. Standard images, http://www.dip.ee.uct.ac.za/imageproc/stdimages/greyscale/, https://ccia.ugr.es/cvg/CG/base.htm, last accessed 2025/04/08
13. BOSSbase 1.01 dataset, Binghamton's University DDE download section, https://dde.binghamton.edu/download/, last accessed 2025/04/08
14. BOWS2 dataset, https://web.archive.org/web/20221129163351/http://bows2.ec-lille.fr/, last accessed 2025/04/08

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97635-3_13