Reversible Data Hiding in Encrypted Images using Kernel-based Prediction and Delta-Huffman Coding

Remigiusz Martyniak[1[0009-0005-9337-1082]], Mariusz Dzwonkowski[1,2[0000-0003-3580-7448]], and Tom Dhaene[3[0000-0003-2899-4636]]

[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
[3] Department of Information Technology (INTEC), IDLab, Ghent University-imec, Technologiepark-Zwijnaarde 126, 9052 Ghent, Belgium
remigiusz.martyniak@pg.edu.pl

Abstract. With the growing demand for secure and efficient handling of large-scale image data in computational environments, lossless data embedding methods for encrypted visual content are gaining importance. This paper introduces a Reversible Data Hiding in Encrypted Images (RDHEI) scheme that allows embedding additional data into encrypted images while ensuring perfect recovery of the carrier. In this approach, reference pixels are selected using a predefined binary mask and compressed using Delta-Huffman coding. The remaining non-reference pixels are predicted using a kernel-based approach, with the kernel weights optimized for each image using Ridge regression. The resulting prediction errors are compressed using standard Huffman coding, thereby increasing the achievable embedding rate. The proposed method is lossless and has been evaluated on standard grayscale images, including those from the BOSSbase and BOWS2 datasets and other benchmark images commonly used in RDHEI research. Experimental results show that the method achieves a higher embedding rate compared to existing RDHEI techniques.

Keywords: Encrypted images, Machine learning, Reversible data hiding.

1 Introduction

The need for secure and efficient handling of sensitive information has become increasingly critical across a wide range of applications, including military communications, medical imaging, legal documentation, and confidential image sharing. Traditional data hiding techniques, while effective in embedding information into digital media, often fail to ensure lossless recovery of the original content—an essential requirement in scenarios where data integrity is paramount.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

Reversible Data Hiding (RDH) addresses this challenge by enabling the embedding of additional data into a carrier image in a way that both the hidden information and the original image can be perfectly restored. Techniques such as histogram shifting [1–2], difference expansion [3–5], and pixel value ordering [6,7] have been widely explored for RDH in plaintext images. However, these methods are limited to plaintext carriers, making them inherently unsuitable for applications involving data embedding within the encrypted domain.

To address the limitations of traditional data hiding methods, Reversible Data Hiding in Encrypted Images (RDHEI) has been developed to securely embed data into encrypted images, providing an additional layer of protection in untrusted environments. RDHEI methods are typically categorized into two main categories: Reserving Room Before Encryption (RRBE) and Vacating Room After Encryption (VRAE). In RRBE, the content owner exploits spatial correlations in pixel values during preprocessing to reserve space for data embedding, often resulting in higher embedding rates. In VRAE, data embedding is performed after encryption, which prevents access to the original spatial information and thus limits the ability to exploit pixel correlations, leading to a typically lower embedding rate. Additionally, RDHEI schemes can also be classified as joint or separable, depending on whether the same or distinct keys are used for encryption and data embedding.

Over the years, a wide range of RDHEI algorithms have been introduced, primarily focused on maximizing the embedding rate through improved pixel prediction, bit-plane manipulation, and advanced compression techniques. Pixel prediction and error map–based approaches remain the most common.

Zhang et al. [8] proposed a block-wise prediction method using sixteen prediction models, which was later extended by Martyniak and Dzwonkowski [9] with seven additional models to better exploit spatial correlations, together with a fine-tuned Extended Run-Length Encoding (ERLE) which further improved the embedding rate. Yu et al. [10] introduced a hierarchical embedding strategy, generating a hierarchical bit-plane label map from prediction error magnitudes to classify pixels into small, medium, and large error categories. Yin et al. [11] employed the Median Edge Detector (MED) with multiple scanning orders to rearrange errors and maximize compression efficiency, while Sui et al. [12] combined hybrid prediction with Huffman coding, predicting the most significant bits using neighbor averaging and the least significant bits using MED. Ping et al. [13] introduced a deep learning–based asymmetric CNN predictor that captures complex spatial correlations via convolutional feature learning. Combined with an adaptive mean predictor in a two-stage embedding framework, the method performs multi-MSB prediction and substitution in the encrypted domain.

Bit-plane-oriented schemes focus on manipulating selected bit planes through compression and rearrangement to facilitate data hiding. Ren et al. [14] proposed an efficient parametric binary tree labeling (EPBTL) scheme that hierarchically partitions prediction errors into structured bit-plane categories to create embedding space. Pixels are divided into non-embedding, embedding, and self-recording sets, with the latter encoded into compact binary-tree codes for direct value recovery. Chen et al. [15] proposed a RRBE-based RDHEI scheme combining adaptive bit-plane (ABP)

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

coding with an order-index extended scrambling (OIES) encryption strategy. The ABP coding adaptively selects among multiple bit-plane coding modes to minimize prediction code length and increase embedding capacity. Fu et al. [16] proposed an adaptive RDHEI scheme that combines "L"-shaped block embedding (LBE) and improved binary-block embedding (IBBE). The method first performs pixel prediction to obtain prediction errors and decomposes them into multiple binary prediction-error bit-planes (PEBPs). LBE and IBBE are then adaptively applied to each bit-plane according to the estimated net embedding capacity, maximizing the utilization of the Laplacian-like distribution of prediction errors. Yao et al. [17] developed an RDHEI framework in which consecutive zero-valued high bit-planes of prediction errors were globally compressed in a block-wise manner. The scheme employed adaptive Huffman indicators to encode the number of compressed bit-planes, a swapping technique to cluster the embeddable planes within each block and a rearrangement method to order blocks in accordance with their embedding capacity. Yu et al. [18] combined bit-plane operations with Chinese Remainder Theorem–based Secret Sharing (CRTSS) and iterative block-based encryption to preserve spatial correlations. A hybrid coding mechanism was employed, in which each block was encoded using either entropy-based or hierarchical coding. Recently, Ankur et al. [19] proposed an rANS-driven RDHEI method, in which block bit-planes are restructured and then compressed using range Asymmetric Numeral System (rANS) coding. This combination produces compact symbol representations that significantly enlarge the available embedding space, while a synchronized block structure ensures reliable recovery within the encrypted domain.

Moreover, regression-based prediction methods have emerged as a promising approach in RDHEI. Huang et al. [20] developed a high-capacity scheme that employs an adaptive linear regression predictor, which is trained on a dataset derived from the original image prior to encryption. Li et al. [21] introduced a sophisticated double linear regression framework. The first layer of the architecture implements three independent linear regressions on distinct subsets of four neighboring pixels, while the second layer synthesizes these outputs to generate the final prediction. While regression-based methods deliver high prediction accuracy, existing approaches still produce limited embedding rates, indicating significant room for improvement.

In this work, a high-payload, separable RRBE-type RDHEI scheme that utilizes linear regression is presented. The process begins with the application of a binary grid mask that selects reference pixels at every other position within every other row and column, while the remaining pixels are treated as non-reference. Reference pixels are used to predict non-reference pixel values through kernel-based prediction, where the weights of the 5×5 kernel are trained for each image using Ridge regression. The prediction errors, calculated as the differences between original and predicted values, form an error map. To maximize the embedding rate, the reference pixels are rearranged into a dense 2D reference map and losslessly compressed by encoding MED-based prediction residuals using Delta-Huffman coding, whereas the error map is compressed with standard Huffman coding. Experimental results on the BOSSbase and BOWS2 datasets demonstrate an average embedding rate exceeding 3.8 bits per

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

pixel (bpp), outperforming other State-of-the-Art RDHEI methods. The key contributions of this study are:

— an RRBE-type RDHEI framework that uniquely integrates kernel-based prediction with image-adaptive weight training via Ridge regression and MED-based Delta-Huffman coding of reference pixels to boost embedding capacity while retaining separability and perfect reversibility;

— validation on a large-scale dataset of 8-bit grayscale images, including a comprehensive comparison with other State-of-the-Art RDHEI methods.

The structure of this paper is as follows. Section 2 provides a detailed explanation of the proposed RDHEI method. Section 3 describes the software and hardware specifications, along with information about the datasets used for evaluation. Section 4 presents the overall results on embedding rate, reconstruction quality, processing complexity and encryption effectiveness. Section 5 discusses the conclusions and key findings.

2 Proposed method

The proposed RDHEI method comprises three stages—preprocessing (at the content-owner side), data hiding (performed by a separate entity), and data extraction with image recovery at the receiver side, as illustrated in Figure 1.

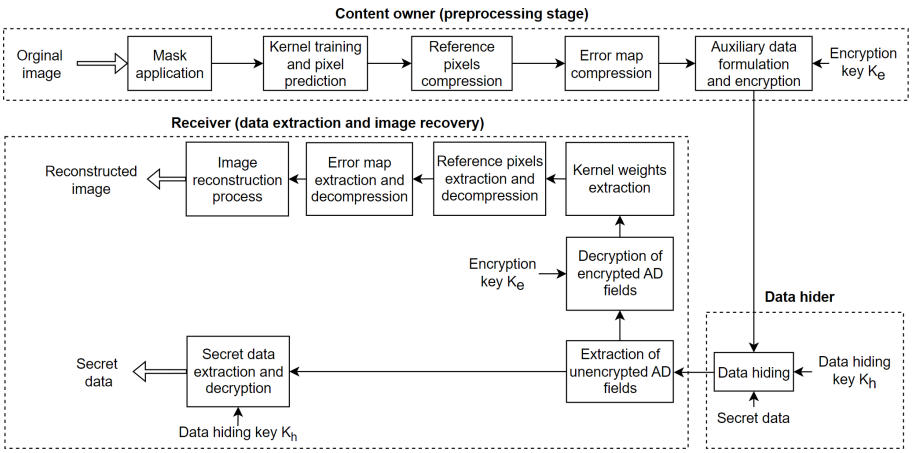

Fig. 1. Proposed RDHEI scheme.

2.1 Mask application, kernel training and pixel prediction

During preprocessing, a binary mask is applied to the input grayscale carrier image to designate pixels as either reference or non-reference. The mask marks pixels at every other position within every other row and column, effectively choosing

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

$\lceil H/2 \rceil \cdot \lceil W/2 \rceil$ reference pixels, where H and W are the height and width of the image, respectively. This grid pattern leverages spatial correlation between neighboring pixels using a sparse subset, balancing prediction accuracy and embedding capacity. A denser pattern, such as a chessboard (50% reference pixels), was empirically tested and found to significantly increase auxiliary data overhead while providing only marginal improvements in prediction accuracy, ultimately reducing the achievable embedding rate for the majority of the tested images.

Prediction of non-reference pixels is performed using a kernel-based linear model. For each non-reference pixel, a $K \times K$ neighborhood centered on that pixel is extracted. Although the neighborhood includes both reference and non-reference pixels, only the reference pixels contribute to the prediction. The neighborhood is flattened in row-major order into a vector of length K^2, denoted as $\mathbf{V}_{\text{ref}}$. Positions corresponding to reference pixels retain their original intensities, while positions associated with non-reference pixels (including the center pixel) are set to zero. Next, the $K \times K$ kernel is used to form the corresponding weight vector of length K^2, denoted as $\mathbf{V}_{\text{kernel}}$. The predicted value of the non-reference pixel is then computed as the dot product $p_{i,j} = \mathbf{V}_{\text{kernel}}^{\text{T}} \cdot \mathbf{V}_{\text{ref}}^{i,j}$.

For non-reference pixels near the image borders, where a complete $K \times K$ neighborhood would extend beyond the carrier image, the window is clipped to the available area. The available reference pixels are then averaged to obtain the prediction. Since kernel-based prediction may not provide optimal results without sufficient neighboring pixels, the averaging approach ensures a simple yet stable estimate that maintains reversibility.

The kernel weights are learned for each image using Ridge regression [22], a linear regression technique that incorporates L2 regularization. Ridge solves the following optimization problem:

$$\beta^* = \arg\min_{\beta} \| y - X\beta \|_2^2 + \alpha \| \beta \|_2^2 \tag{1}$$

Here, X is the feature matrix of size $L \times K^2$, where L is the number of training samples (all non-reference pixels in the carrier). Each row corresponds to one feature vector constructed from the $K \times K$ neighborhood around a non-reference pixel; y is the target vector (one-dimensional array of L length), composed of the original intensity values of non-reference pixels; β represents the K^2 kernel weights to be learned, and $\alpha > 0$ is a regularization strength hyperparameter. In equation (1), $\| y - X\beta \|_2^2$ measures the prediction error, i.e., the sum of squared differences between the true intensities and those predicted by the model; $\| \beta \|_2^2$ is the regularization term, which penalizes large weights, reducing overfitting and improving stability when reference pixel values are correlated. In this study, Ridge regression is implemented using scikit-learn's Ridge class with the following settings: $\alpha = 1$, no intercept term, and Singular Value Decomposition (SVD) as a solver, enabling efficient, direct computation of the weights using closed-form analytical solution. This approach makes kernel

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

training substantially faster than L1 regularization methods such as Lasso, while ensuring comparable prediction performance (as presented later in Table 2).

Figure 2 illustrates a 5 × 5 kernel containing sample Ridge regression weights, an 8 × 8 image with reference pixels shown in white and non-reference pixels shown in violet, and two prediction scenarios. In the border case, where the kernel window extends beyond the image boundary, the predicted value is calculated as the average of the available reference pixels in the clipped window. In the interior case, where the kernel fits entirely within the image, the predicted value is computed as the dot product of the kernel weights $\mathbf{V}_{\text{kernel}}$ and the feature vector $\mathbf{V}_{\text{ref}}$. The center kernel position always corresponds to the pixel being predicted and is therefore set to zero in the feature vector, ensuring that only surrounding reference pixels contribute to the prediction.

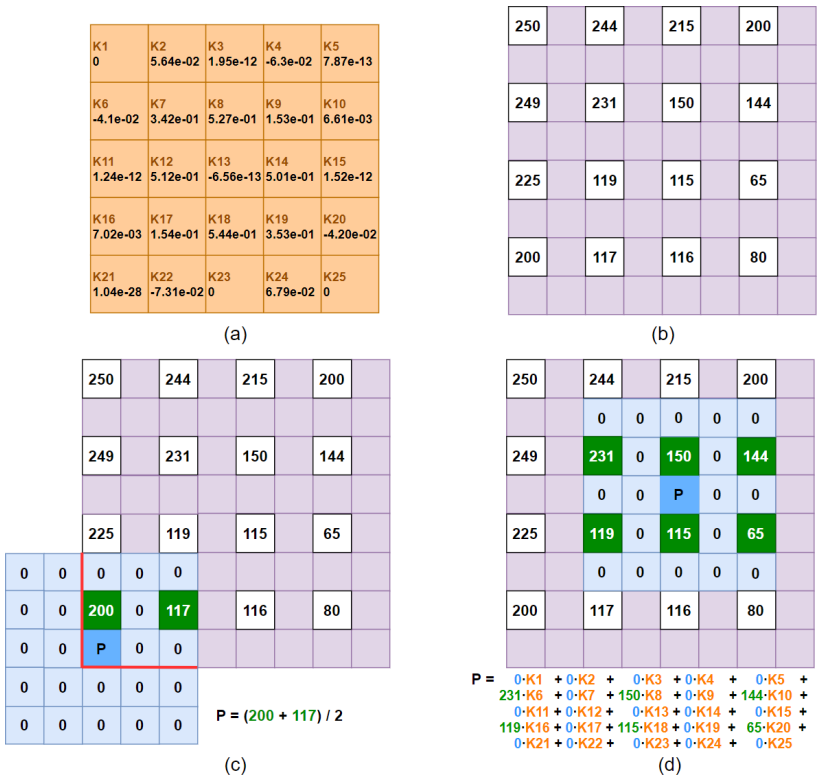

Fig. 2. Non-reference pixel prediction: (a) a 5 × 5 kernel with trained weights, (b) an 8 × 8 image with applied mask, (c) border case prediction using averaging and (d) interior case prediction using the dot product.

Table 1 presents predicted PSNR and SSIM results for 2000 images (the first 1000 from BOSSbase and the first 1000 from BOWS2), comparing different kernel sizes to determine the optimal preprocessing configuration. The 3 × 3 kernel proves insuffi-

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

cient for accurate prediction, yielding poor average PSNR (10.37 dB) and SSIM (0.134) values, which leads to large compressed error maps (on average 1.3591 Mb) and a reduced embedding rate. The 5 × 5, 7 × 7, and 9 × 9 kernels achieve comparable performance, with average PSNR around 33 dB, SSIM around 0.92, and error map sizes of approximately 0.797–0.8 Mb. The 5 × 5 kernel was selected as optimal due to its superior computational efficiency (0.081 s training time, compared to 0.223 s and 0.440 s for larger kernels) and lower storage requirement (only 25 64-bit float values compared to 49 or 81 for larger variants), improving the overall embedding rate.

Table 1. Prediction, error compression and kernel training time results for different kernel sizes.

Kernel	PSNR [dB]			SSIM			Avg error map	Avg kernel train-
size	Min	Avg	Max	Min	Avg	Max	size [Mb]	ing time [s]
3 × 3	3.89	10.37	31.01	0.005	0.134	0.791	1.3591	0.007
5 × 5	21.38	32.65	52.78	0.515	0.918	0.996	0.7967	0.081
7 × 7	21.33	32.83	52.13	0.708	0.919	0.996	0.7975	0.223
9 × 9	21.29	32.75	52.18	0.705	0.917	0.996	0.8002	0.440

Table 2 presents the prediction accuracy and training time for different regression methods for a 5 × 5 kernel. Ridge (SVD) is considerably faster than Lasso across all tested iteration limits (100, 200, 400, and 800). Comparable prediction performance between the two methods is achieved only at 400 or 800 Lasso iterations, which substantially increases average training time to 0.535 s and 0.798 s, respectively, compared to 0.081 s for Ridge. This confirms the suitability of L2 regularization in the proposed method. Additional tests with different α values (0.01, 0.1, and 0.5) for both methods produced nearly identical results and were therefore omitted from the table.

Table 2. Prediction performance and kernel training time ($K = 5$) results for Ridge and Lasso regression methods.

Metrics	PSNR [dB]			SSIM			Kernel training time [s]		
	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
Ridge (SVD)	21.38	32.65	52.78	0.515	0.918	0.996	0.064	0.081	0.113
Lasso (100 it)	21.33	31.15	49.18	0.513	0.901	0.994	0.069	0.199	0.288
Lasso (200 it)	21.33	31.93	49.59	0.513	0.912	0.995	0.074	0.311	0.410
Lasso (400 it)	21.37	32.34	49.62	0.513	0.916	0.995	0.071	0.535	0.785
Lasso (800 it)	21.37	32.52	50.11	0.513	0.917	0.996	0.063	0.798	1.336

2.2 Reference pixels and error map compression

To maximize the embedding rate, both reference pixels and the error map undergo lossless compression before data embedding. The reference pixels are compressed using Delta-Huffman coding, which outperforms direct Huffman coding of raw pixel values due to its ability to effectively exploit spatial correlations. First, all reference

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

pixels are serialized in row-major order to form a reference map M_{ref} of size $\lceil H/2 \rceil \times \lceil W/2 \rceil$. Next, the values of M_{ref} are predicted using the MED predictor, while the first row and the first column are kept unchanged and treated as boundary references:

$$\hat{M}_{\text{ref}}(i,j) = \begin{cases} \min(L,T) & \text{if } TL \geq \max(L,T) \\ \max(L,T) & \text{if } TL \leq \min(L,T) \\ L+T-TL & \text{otherwise} \end{cases} \tag{2}$$

where $L = \hat{M}_{\text{ref}}(i, j-1)$, $T = \hat{M}_{\text{ref}}(i-1, j)$, and $TL = \hat{M}_{\text{ref}}(i-1, j-1)$ denote the Left, Top, and Top-Left neighbors of the predicted pixel $\hat{M}_{\text{ref}}(i,j)$, respectively. $M_{\text{ref-dif}}$ is then computed (excluding the first row and the first column) as the difference between M_{ref} and its prediction $\hat{M}_{\text{ref}}$. Subsequently, the first row and the first column of $M_{\text{ref-dif}}$ are encoded using signed delta values, computed with respect to the preceding pixel within the corresponding row or column. The top-left reference pixel (i.e., the first element of $M_{\text{ref-dif}}$) is preserved in its original form. To handle potential negative values, an offset of +255 is added to all elements of $M_{\text{ref-dif}}$ except the top-left one. Finally, $M_{\text{ref-dif}}$ is flattened in row-major order and Huffman encoded.

In contrast, the error map is compressed using standard Huffman coding. Since error values are typically concentrated around zero and exhibit a strongly peaked distribution, they are inherently well suited to entropy coding. Prior to Huffman compression, each error value is offset by +255 to map the range to [0, 510], in the same manner as the reference pixels.

2.3 Auxiliary data formulation and encryption

In the proposed method, after compressing the reference pixels, generating and compressing the error map, the auxiliary data (AD) is formed (cf. Figure 3) to enable lossless recovery of the carrier. AD is positioned at the beginning of the reserved embedding region of size $N \cdot 8$ bits within the carrier, where N denotes the total number of pixels in the image. The AD is structured as follows:

- **Image dimensions.** Two 10-bit fields that store the carrier image height H and width W. The bit length of both fields was chosen primarily to handle the tested images but it can be easily increased if necessary.
- **AD length.** A field of $\lceil \log_2(H \cdot W \cdot 8) \rceil$ bits specifying the total length of the AD binary sequence. It allows both the data hider and the receiver to precisely determine the end of the entire AD segment, implying that the remaining embedding region for secret data begins immediately afterward—one bit later.
- **Kernel weights.** A field of $64 \cdot K^2$ bits, where K is the kernel size, containing serialized kernel weights stored as 64-bit floats. The weights are used to predict non-reference pixel values from neighboring reference pixels.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

— **Compressed reference pixels.** A section that begins with a $\lceil \log_2 (N_{\text{ref}} \cdot b_{\text{sym}}) \rceil$ bits field, where N_{ref} is the number of reference pixels and b_{sym} is the number of bits needed to store a single symbol ($b_{\text{sym}} = \lceil \log_2 S \rceil$ for S possible symbol values). This field indicates the bit length of the following Delta-Huffman codebook segment, which starts immediately afterwards. In the codebook, each entry includes a symbol value (b_{sym} bits) representing delta-offsets in the range [0, 510], a code length of $\lceil \log_2 (L_{\text{max}}) \rceil$ bits, where L_{max} is the maximum Huffman code length, and the Huffman code itself, stored as a variable-length binary string. Following the codebook, another $\lceil \log_2 (N_{\text{ref}} \cdot b_{\text{sym}}) \rceil$ bits field specifies the bit length of the compressed reference pixel data, which is stored directly afterwards.

— **Compressed error map.** Similar to reference pixels, this section starts with a $\lceil \log_2 (N_{\text{non-ref}} \cdot b_{\text{sym}}) \rceil$ bit codebook length field, where $N_{\text{non-ref}}$ is the number of non-reference pixels and b_{sym} is the number of bits needed to store an error map symbol ($b_{\text{sym}} = \lceil \log_2 S \rceil$ for S possible symbols). In the codebook, each entry includes a symbol value (b_{sym} bits), a code length of $\lceil \log_2 (L_{\text{max}}) \rceil$ bits, where L_{max} is the maximum Huffman code length, and the Huffman code itself, stored as a variable-length binary string. Lastly, $\lceil \log_2 (N_{\text{non-ref}} \cdot b_{\text{sym}}) \rceil$ bits field specifies the bit length of the compressed error map.

To secure the carrier image with embedded auxiliary data, the entire binary sequence of the image—excluding the embedded AD length field—is encrypted using a bitwise XOR operation with a pseudo-random bitstream derived from the encryption key K_e. The bitstream is generated using a ChaCha20-based Cryptographically Secure Pseudo-Random Number Generator (CSPRNG), which provides a high-quality pseudorandom sequence commonly employed in practical encryption settings and mitigates straightforward statistical dependencies between the encrypted and original image.

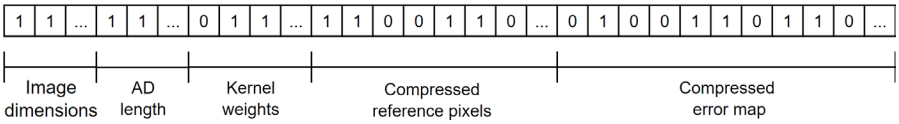

Fig. 3. Auxiliary data structure.

2.4 Data hiding

The space following the embedded AD sequence within the encrypted carrier is reserved for embedding additional secret data. The starting point is determined by the value of the AD length field. Secret data is encrypted via a bitwise XOR operation

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

with a pseudo-random bitstream derived from the data hiding key K_h, generated by a ChaCha20-based CSPRNG to ensure strong cryptographic security. The resulting encrypted data string is then inserted into the remaining embedding region of the encrypted image, directly after the identified AD sequence. Finally, the encrypted and marked image—containing both the embedded AD and secret data—is ready to be sent to the receiver (cf. Figure 1).

2.5 Carrier image recovery and secret data extraction

Upon receiving the encrypted and marked image, the recipient can recover the original carrier and extract the embedded secret data through a fully reversible routine. The process begins by reading the unencrypted Image dimensions and AD length fields to determine dimensions of the carrier as well as the exact boundary between auxiliary data and secret payload. The receiver is then able to isolate the AD bitstream and decrypt it using encryption key K_e. From the decrypted auxiliary data, the receiver sequentially extracts the kernel weights stored as 64-bit floats each, followed by reference pixels (obtained by first reading the corresponding codebook specification and then decoding the Delta-Huffman compressed data) and error map (obtained by first reading the corresponding codebook specification and then decoding the Huffman compressed data). Both the reference pixels as well as error map values are offset-corrected (by subtracting 255) to recover their original form. Additionally, the reference-pixel stream is de-vectorized to reconstruct $M_{\text{ref-dif}}$ in row-major order (note that its dimensions are obtained from the Image dimensions filed). The original boundary elements are first restored iteratively, pixel by pixel, by cumulatively summing the signed deltas along the first row and the first column, while keeping the top-left (initial) element from $M_{\text{ref-dif}}$ unchanged. Next, the remaining pixels are recovered by inverting the MED-based delta representation: the MED predictor is recomputed using already reconstructed neighbors, and the corresponding prediction residuals from $M_{\text{ref-dif}}$ are added to obtain the correct reference map M_{ref}.

The original image is reconstructed by placing the recovered reference pixels in their designated positions using the predefined binary mask and predicting non-reference pixels using the recovered kernel weights, in the same manner as described in Section 2.1. Finally, the predicted values are adjusted by adding corresponding error values from the recovered error map.

To extract the embedded secret data, the receiver locates the bitstream immediately following the auxiliary data region, whose boundary was already determined by the AD length field. The identified segment contains the encrypted secret payload, which the receiver decrypts using the data hiding key K_h.

3 Experimental setup

All experiments were conducted on a Windows 11 workstation equipped with an AMD Ryzen 5 7600X processor and 32 GB of DDR5 RAM. The proposed RDHEI algorithm was implemented in Python 3.12.2 using the scikit-learn 1.6.0 library. Ker-

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

nel training was performed on the CPU using scikit-learn's Ridge class, as this implementation does not currently support CUDA-based GPU acceleration. For final validation, 20,000 8-bit grayscale images of size 512 × 512 pixels were used from the BOSSbase [23] and BOWS2 [24] datasets. Additionally, experiments were conducted on four standard benchmark images: Baboon, Jetplane, Man, and Lena.

4 Results

This section presents the evaluation results of the proposed method, including embedding rate, prediction and reconstruction metrics, error map statistics, and encryption effectiveness. For all testing, the optimal variant (5 × 5 kernel and Ridge regression) was chosen based on the preprocessing evaluation presented in Section 2.1.

4.1 Image reconstruction and data embedding effectiveness

High effectiveness of the proposed RDHEI method in both image reconstruction and data embedding is demonstrated by the results in Table 3. Across all datasets and individual test images, the reconstructed carriers achieved SSIM = 1 and PSNR of $+\infty$ dB, since MSE = 0, confirming that the reconstructed image is bit-for-bit identical to the original. The average predicted PSNR was 33.29 dB for BOSSbase and 32.35 dB for BOWS2, indicating a highly accurate prediction stage. Among standard images, the highest predicted PSNR was 33.18 dB (Lena), and the lowest was 24.08 dB (Baboon), which is expected due to the latter's complex texture. The method achieved an average embedding rate of 3.885 bpp for BOSSbase and 3.861 bpp for BOWS2, with maximum values reaching 6.873 bpp and 6.469 bpp, respectively. Error map statistics further indicate that residuals are concentrated around small values, which confirms the reliability of the proposed prediction approach.

Based on the comparative analysis presented in Table 4, the proposed RDHEI scheme demonstrates competitive performance against existing State-of-the-Art methods. For the BOSSbase dataset, the proposed method achieves higher average embedding rate than other methods, including Chen [15] (3.696 bpp) and Yao [17] (3.793 bpp). Similarly, for the BOWS2 dataset, the proposed method maintains consistent performance, also surpassing other approaches. Additionally, for standard test images, the proposed RDHEI scheme consistently achieves high embedding rates.

Figure 4 demonstrates the kernel-based prediction process performed on the Man test image, showing the original image and the predicted image. The error map visualized as a heatmap reveals that prediction errors are predominantly concentrated around edges and textural regions, with most areas showing minimal errors (white regions), while higher errors (red and blue regions) appear mainly at high-contrast boundaries. The corresponding error map histogram confirms the effectiveness of the prediction method, showing a highly peaked distribution centered around zero with the majority of prediction errors falling within a narrow range, which is ideal for efficient Huffman compression and contributes to the high embedding rates achieved by the proposed RDHEI scheme.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

Table 3. Evaluation results for the proposed RDHEI scheme.

Images		Predicted PSNR [dB]	Reconstructed PSNR [dB]	Predicted SSIM	Reconstructed SSIM	Embedding rate [bpp]	Error map avg values	Error map medians
BOSSbase	Min	20.90	+∞	0.515	1	1.073	-0.73	0
	Avg	33.29	+∞	0.916	1	3.885	0.27	0
	Max	53.06	+∞	0.998	1	6.873	2.35	1
BOWS2	Min	20.68	+∞	0.670	1	0.937	-0.77	0
	Avg	32.35	+∞	0.921	1	3.861	0.27	0
	Max	51.03	+∞	0.997	1	6.469	0.53	0
Baboon		24.08	+∞	0.749	1	1.739	0.24	0
Jetplane		29.90	+∞	0.944	1	3.499	0.08	0
Man		30.57	+∞	0.899	1	3.014	0.13	0
Lena		33.18	+∞	0.917	1	3.344	0.28	0

Table 4. Comparison of the average embedding rates for the proposed RDHEI scheme against other State-of-the-Art methods.

Images	This work	Method [9]	Method [13]	Method [14]	Method [15]	Method [16]	Method [17]
BOSSbase	3.885	3.597	3.461	2.667	3.696	3.195	3.793
BOWS2	3.861	3.482	3.302	2.637	3.543	3.134	3.705
Baboon	1.739	1.828	1.199	1.343	1.530	1.523	1.375
Jetplane	3.499	3.321	3.183	2.767	3.404	3.176	3.535
Man	3.014	2.610	2.285	2.644	2.819	2.566	2.647
Lena	3.344	2.989	2.821	2.760	3.175	3.014	2.859

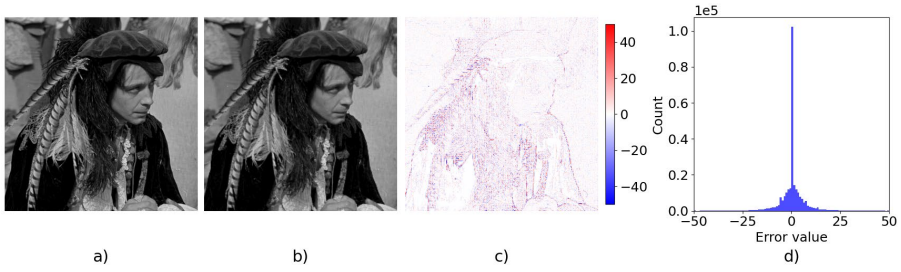

Fig. 4. Kernel-based prediction performed on the Man image: (a) original image, (b) predicted image (PSNR = 30.57 dB), (c) error map visualized as a heatmap, (d) histogram of the error map.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

4.2 Analysis of method's complexity and encryption effectiveness

The computational cost of the proposed RDHEI framework is primarily determined by operations that traverse the image domain exactly once. Feature extraction, Ridge-based kernel estimation with a fixed kernel size, Huffman encoding and decoding, and pixel-wise reconstruction all scale linearly with respect to the number of pixels N. In particular, for a fixed kernel dimension, the SVD-based Ridge regression has complexity $O(N)$, as the feature dimensionality remains constant. Consequently, both the content owner and the receiver incur a time complexity of $O(N)$. The data-hiding procedure, in contrast, processes only the embeddable portion of the bitstream. Since ChaCha20 bitstream generation and XOR operations scale with the size of this region, the data hider operates in $O(C_e)$, where $C_e < N$ denotes the number of bytes available for embedding (for 8-bit images, $C_e = N - AD_{length}/8$). The per-stage computational characteristics of the proposed and reference methods are summarized in Table 5.

The memory usage of all stages is proportional to the carrier size, as each stage stores only the image data along with the auxiliary information, without introducing any super-linear data structures. Therefore, the space complexity of the proposed method is $O(N)$. Overall, the per-image complexity is $O(N)$ in both time and space, providing the same asymptotic efficiency as existing linear-time schemes such as [10] and [17], while remaining more efficient than the quasi-linear method [19]. In addition, as a practical implementation-level observation, the encryption and decryption steps on the machine used in the experiments required only a few milliseconds per image. This remark is included for completeness, in order to complement the theoretical complexity discussion with an indication of the practical runtime under the adopted experimental conditions, while noting that the exact execution time depends on the hardware and software environment.

Table 5. Computational time and space complexities of the proposed method and other State-of-the-Art RDHEI schemes.

Complexity	This work				Method [10]	Method [17]	Method [19]
	Content owner	Data hider	Receiver	Total per image			
Time	$O(N)$	$O(C_e)$	$O(N)$	$O(N)$	$O(N)$	$O(N)$	$O(N \log N)$
Space	$O(N)$	$O(N)$	$O(N)$	$O(N)$	$O(N)$	$O(N)$	$O(N)$

To determine the effectiveness of the encryption routine that relies on the ChaCha20 stream cipher, three standard evaluation metrics were utilized, including histogram analysis (cf. Figure 5) for both original and encrypted images, as well as the Number of Pixel Change Rate (NPCR) and Unified Average Changing Intensity (UACI). On average, the NPCR values reached 99.66% and the UACI results averaged 32.8% for the tested images, which confirms that the implemented RDHEI algorithm ensures the desired level of security for encrypted data.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

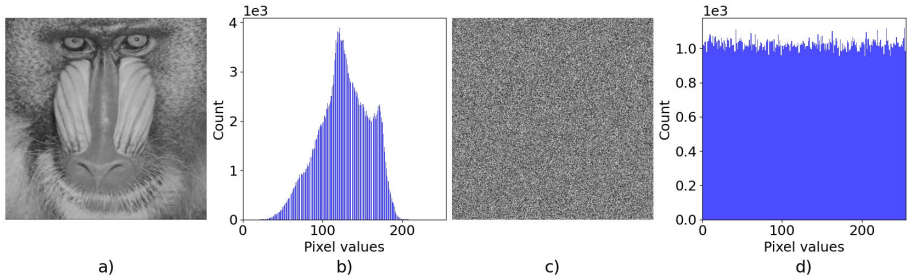

Fig. 5. Encryption of the Baboon image: (a) original image, (b) histogram of the original image, (c) encrypted image, and (d) histogram of the encrypted image.

5 Conclusions

A computationally efficient, fully reversible, and separable RRBE-type RDHEI framework has been proposed. The method combines kernel-based pixel prediction and Delta-Huffman coding to maximize embedding effectiveness, achieving an average rate of 3.885 bpp on BOSSbase and 3.861 bpp on BOWS2 datasets. With linear time and space complexity, it is well suited for large-scale scientific imaging requiring secure and reproducible data exchange. The integration of regression-based prediction and adaptive compression demonstrates the potential of machine learning–driven models to enhance prediction accuracy and embedding rate.

Future work will focus on improving error map compression, dynamic reference pixel selection, and integrating edge-preserving filters such as bilateral and anisotropic diffusion to refine prediction accuracy. Additional research will explore multiple adaptive kernels, GPU-accelerated kernel optimization, and extensions to 16-bit medical imagery (e.g., DICOM) to further enhance its applicability in computational science applications like sensitive clinical environments.

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.

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

References

1. Z. Ni, Y.Q. Shi, N. Ansari, W. Su, Reversible data hiding, IEEE Trans. Circuits Syst. Video Technol. 16 (3) (2006) 354–362.
2. X. Li, W. Zhang, X. Gui, B. Yang, Efficient reversible data hiding based on multiple histograms modification, IEEE Trans. Inf. Forensics Secur. 10 (9) (2015) 2016–2027.
3. J. Tian, Reversible watermarking by difference expansion, in: J. Dittmann, J. Fridrich, P. Wohlmacher (Eds.), Proceedings of the Workshop on Multimedia and Security: Authentication, Secrecy, and Steganalysis, Association for Computing Machinery, Inc, 2002, pp. 19–22.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44

4. A.M. Alattar, Reversible watermark using the difference expansion of a generalized integer transform, IEEE Trans. Image Process. 13 (8) (2004) 1147–1156.
5. Y. Hu, H.K. Lee, K. Chen, J. Li, Difference expansion based reversible data hiding using two embedding directions, IEEE Trans. Multimed. 10 (8) (2008) 1500–1512.
6. X. Li, J. Li, B. Li, B. Yang, High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion, Signal Process. 93 (1) (2013) 198–205.
7. X. Qu, H.J. Kim, Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding, Signal Process. 111 (2015) 249–260.
8. H. Zhang, L. Li, Q. Li, Reversible data hiding in encrypted images based on block-wise multi-predictor, IEEE Access 9 (2021) 61943–61954.
9. R. Martyniak, M. Dzwonkowski, Reversible data hiding in encrypted images with pixel prediction and ERLE compression, in: M.H. Lees, et al. (Eds.), Computational Science – ICCS 2025, Lecture Notes in Computer Science, vol. 15906, Springer, Cham, 2025.
10. C. Yu, X. Zhang, X. Zhang, G. Li, Z. Tang, Reversible data hiding with hierarchical embedding for encrypted images, IEEE Trans. Circuits Syst. Video Technol. 32 (2) (2022) 451–466.
11. Z. Yin, Y. Peng, Y. Xiang, Reversible data hiding in encrypted images based on pixel prediction and bit-plane compression, IEEE Trans. Dependable Secure Comput. 19 (2) (2022) 992–1002.
12. L. Sui, H. Li, J. Liu, Z. Xiao, A. Tian, Reversible data hiding in encrypted images based on hybrid prediction and Huffman coding, Symmetry 15 (6) (2023) 1222.
13. P. Ping, J. Huo, B. Guo, Novel asymmetric CNN-based and adaptive mean predictors for reversible data hiding in encrypted images, Expert Syst. Appl. 246 (2024) 123270.
14. H. Ren, Z. Yue, F. Gu, M. Li, T. Chen, G. Bai, A novel reversible data hiding method in encrypted images using efficient parametric binary tree labeling, Knowl.-Based Syst. 300 (2024) 112198.
15. F. Chen, Y. Yang, H. He, Y. Yuan, Adaptive coding and ordered-index extended scrambling based RDH in encrypted images, IEEE Trans. Multimed. 25 (2023) 2864–2875.
16. Z. Fu, X. Chai, Z. Tang, X. He, Z. Gan, G. Cao, Adaptive embedding combining LBE and IBBE for high-capacity reversible data hiding in encrypted images, Signal Process. 216 (2024) 109299.
17. Y. Yao, K. Wang, Q. Chang, S. Weng, Reversible data hiding in encrypted images using global compression of zero-valued high bit-planes and block rearrangement, IEEE Trans. Multimed. 26 (2024) 3701–3714.
18. C. Yu, X. Zhang, C. Qin, Z. Tang, Reversible data hiding in encrypted images with secret sharing and hybrid coding, IEEE Trans. Circuits Syst. Video Technol. 33 (11) (2023) 6443–6458.
19. Ankur, R. Kumar, P. Ranjan, K.-H. Jung, Leveraging rANS for synchronized high capacity reversible data hiding in encrypted image, Expert Syst. Appl. 267 (2025) 126181.
20. B. Huang, C. Wan, K. Chen, High-capacity reversible data hiding in encrypted images based on adaptive predictor and compression of prediction errors, Mathematics 9 (17) (2021) 2166.
21. F. Li, H. Zhu, J. Yu, et al., Double linear regression prediction based reversible data hiding in encrypted images, Multimed. Tools Appl. 80 (2021) 2141–2159.
22. A.E. Hoerl, R.W. Kennard, Ridge regression: Biased estimation for nonorthogonal problems, Technometrics 12 (1) (1970) 55–67.
23. BOSSbase 1.01 dataset, https://dde.binghamton.edu/download/, last accessed 11/12/2025.
24. BOWS2 dataset, https://web.archive.org/web/20221129163351/http://bows2.ec-lille.fr/, last accessed 11/12/2025.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29915-4_44