Hindawi
Security and Communication Networks
Volume 2022, Article ID 4336526, 14 pages
https://doi.org/10.1155/2022/4336526

Research Article

High-Capacity Data Hiding Method Based on Two Subgroup Pixels-Value Adjustment Using Encoding Function

Jie Sun, ZhaoFang Yang, Yu Zhang, Teng Li, and Sha Wang

College of Computer and Information Science, Southwest University, Chongqing 400715, China

Correspondence should be addressed to ZhaoFang Yang; goodluck@swu.edu.cn

Received 9 March 2022; Revised 21 June 2022; Accepted 30 June 2022; Published 22 July 2022

Academic Editor: De Rosal Ignatius Moses Setiadi

Copyright © 2022 Jie Sun et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Confidential information can be hidden in digital images through data hiding technology. This has practical application value for copyright, intellectual property protection, public information protection, and so on. In recent years, researchers have proposed many schemes of data hiding. However, existed data hiding schemes suffer from low hiding capacity or poor stego-image quality. This paper uses a new method of multiple pixels-value adjustment with encoding function (MPA) to further improve the comprehensive performance, which is well in both hiding capacity and stego-image quality. The main idea is to divide n adjacent cover pixels into two sub-groups and implement multi-bit-based modulus operations in each group, respectively. The efficacy of this proposed is evaluated by peak signal-to-noise ratio (PSNR), embedding payload, structural similarity index (SSIM), and quality index (QI). The recorded PSNR value is 30.01 dB, and embedding payload is 5 bpp (bits per pixel). In addition, the steganalysis tests do not detect this steganography technique.

1. Introduction

With the rapid development of the Internet, more and more people transmit messages through the network. But at the same time, they also face the risk of information leakage. Hence, information security is getting high attention. In recent years, many techniques for information security have been researched such as cryptography and data hiding. Cryptography transforms secret information into an incomprehensible form, and the encrypted message looks messy and meaningless. The shortcoming of cryptography is that it is easily detected and then destroyed by the attacker. Unlike cryptography, data hiding ensures that there is no easily detectable change in the carrier of the hidden secret message. This technology can ensure the security of confidential information. One method of data hiding is to embed secret information in the cover image and then generate a stego-image that prevents attackers from stealing it [1]. There are various ways to classify data hiding, which can be divided into spatial domain and frequency domain. In spatial domain-based techniques, the pixel values of the image are directly modified to embed secret information, whereas in frequency domain-based techniques, pixels are transformed into coefficients after changes (e.g., Fourier transform, Laplace transform, and Z-transform). These coefficients are modified to embed secret data. Among these two domains, the spatial domain-based techniques are simple and less time consuming. In this domain, according to whether the stego-image can be recovered after embedding secret data, data hiding can be divided into reversible data hiding [2–6] and irreversible data hiding. Extracting secret data in stego-images also allow recovery of the cover image without distortion, and this method is suitable for some fields with high image requirements, such as medical and military. In contrast, irreversible data hiding cannot restore the cover image, but this method can embed more secret data and is easy to understand and implement.

In this paper, we focus on irreversible data hiding. In the past decades, many data hiding methods have been proposed. These methods can be divided into three categories, LSB-based, PVD-based, and EMD-based. Firstly, least significant bit (LSB)-based method embeds secret data through the least significant bits of cover pixels, which is the simplest and most direct approach. LSB is simple to implement, but

the generated stego-images may draw suspicion or be easily detected. In an image, the texture region can hide more secret data compared to the smooth region. Therefore, the goal of the pixel value difference (PVD)-based method is to hide less secret data in the smooth region and embed more secret data in the texture region. For this target, scholars have proposed many variants of PVD [7, 8] or combined PVD with methods such as LSB [9, 10] or modulus functions. The last type is exploiting modified direction (EMD)-based method, using special equations or modulus functions to embed secret data. In 2006, Zhang and Wang [11] proposed EMD by making full use of the modulus function to change the directional characteristics. The EMD method uses a function to embed secret data transformed in the $(2n+1)$-ary notational system in n cover pixels. When $n=2$, EMD can achieve the maximum embedding payload of 1.16 bpp. To improve the embedding capability, many data hiding techniques based on EMD methods have been proposed [12]. In 2007, Lee et al. [13] proposed the improved EMD (IEMD) method to embed secret data using modulus function. The embedding payload of IEMD is improved from 1.16 bpp to 1.5 bpp. However, the fixed number of cover pixels and weights in each group leads to this scheme being less flexible, which is weak against steganalysis. In 2009, Jung [14] proposed JY, which uses function to embed n secret data transformed in the $(2n+1)$-ary notational system into 1 cover pixel, increasing the single-pixel embedding payload to 2 bpp. In 2013, Kuo and Wang [15] proposed generalized exploiting modification direction (GEMD) method. The scheme is an extension of IEMD. Extending the fixed weights to vary with n, the method has an embedding payload of 2 bpp. In 2015, Kuo et al. [16] proposed the KKWW scheme to optimize the parameters and modulus function of GEMD. The simulation results showed that the scheme is feasible. In 2019, Sairam and Boopathybagan [17] proposed SB algorithm to carry secret data using other notational system. The maximum embedding payload of this method is 4 bpp. To improve the embedding capacity, in 2020, Zhang et al. [18] proposed the modulus calculations on prime number algorithm (MOPNA). This method is based on the modulo operation of prime numbers. The embedding payload of this algorithm is increased to 3.5 bpp.

To further increase the embedding payload of per pixel while ensuring the quality of stego-image, this paper proposes a high-capacity data hiding scheme MPA. The main contributions of this paper are summarized as follows.

(1) In this paper, we propose a new scheme that adopts the grouping technique of cover pixel groups and uses modulus function for secret data embedding which improves the capacity of data hiding. With this method, the embedding payload can reach 5 bpp, while the quality of the stego-image is still acceptable (PSNR > 30 dB).

(2) The correctness of the MPA scheme was demonstrated with mathematical and the experimental results which proved the performance and safety of the MPA scheme.

(3) To make it easier for other scholars to verify the work, we have uploaded all the code and other materials at https://github.com/SunJie0916/MPA.

The rest of this paper is organized as follows. Section 2 will show the related work. The new algorithm MPA and its math proofs are presented in Section 3. Section 4 provides the experimental details and results of MPA. Finally, a conclusion is given in Section 5.

2. Related Work

2.1. Exploiting Modified Direction (EMD) Algorithm. Zhang et al. proposed EMD, which embeds the secret digit $S_{(2n+1)}$ transformed in the $(2n+1)$-ary notational system into n cover pixels. The payload of EMD is $\log_2(2n+1)/n$, up to 1.16 bpp when $n=2$. There is room to improve. And the algorithm is as follows. The inputs of EMD are n cover pixels $G(g_1, g_2, \ldots, g_n)$ and $S_{(2n+1)}$. The output is $G'(g_1', g_2', \ldots, g_n')$.

Step 1. Calculate the extraction function in $f_{\mathrm{EMD}}(G) = (\sum_{i=1}^{n} i \times g_i) \bmod (2n+1)$.

Step 2. G' is obtained as follows.

$G' = GD = (S_{(2n+1)} - f_{\mathrm{EMD}}(G)) \bmod (2n+1)$

if $D > 0$ and $D \le n$, then $g_D' = g_D + 1$

else $g_D' = g_{2n+1-D} - 1$

Step 3. Calculate $f_{EM\,D}(G')$ to get the secret data.

2.2. Improved EMD (IEMD) Algorithm. In 2007, Lee et al. proposed a data hiding algorithm IEMD. Its embedding payload of IEMD is 3 bpp. More detailed steps are shown below. The inputs of IEMD are a pixel pair $G(g_1, g_2)$ and decimal digit $S_{(10)}$ which is transformed from 3 binary bit secret data. The output is $G'(g_1', g_2')$.

Step 1. Calculate using $f_{\mathrm{IEMD}}(G) = (1 \times g_1 + 3 \times g_2) \bmod 8$.

Step 2. $G'(g_1', g_2')$ is obtained as follows.

if $S_{(10)} = f_{\mathrm{IEMD}}(g_1, g_2)$, then $g_1' = g_1$ and $g_2' = g_2$

else if $S_{(10)} = f_{\mathrm{IEMD}}(g_1 + 1, g_2)$, then $g_1' = g_1 + 1$ and $g_2' = g_2$

else if $S_{(10)} = f_{\mathrm{IEMD}}(g_1 - 1, g_2)$then $g_1' = g_1 - 1$ and $g_2' = g_2$

else if $S_{(10)} = f_{\mathrm{IEMD}}(g_1, g_2 + 1)$, then $g_1' = g_1$ and $g_2' = g_2 + 1$

else if $S_{(10)} = f_{\mathrm{IEMD}}(g_1, g_2 - 1)$, then $g_1' = g_1$ and $g_2' = g_2 - 1$

else if $S_{(10)} = f_{\mathrm{IEMD}}(g_1 + 1, g_2 - 1)$, then $g_1' = g_1 + 1$ and $g_2' = g_2 - 1$

else if $S_{(10)} = f_{\mathrm{IEMD}}(g_1 - 1, g_2 + 1)$, then $g_1' = g_1 - 1$ and $g_2' = g_2 + 1$

Step 3. Calculate $f_{\mathrm{IEMD}}(G')$ to get the secret data.

2.3. JY. Jung et al. proposed the JY algorithm. The payload of the algorithm is up to 2 bpp. The details of JY are shown as

follows. The inputs of JY are 1 cover pixel $G(g)$ and n binary bit secret data. The output is stego-pixel $G'(g')$.

Step 1. Calculate $f_{\mathrm{JY}}(G) = g \bmod (2n+1)$.

Step 2. Based on the value of $g$choose the range of x that satisfies the requirements of the following equation and bring it in order to calculate $d = (g+x) \bmod (2n+1)$. And choose the x that satisfies $f_{\mathrm{JY}}(G) = d$, and then $g' = g + x$.

if $0 \le g \le 1$then $x\varepsilon[0, (2n+1)]$

else if $254 \le g \le 255$, $x\varepsilon[-(2n+1), 0]$

else $x\varepsilon[-(2n+1), (2n+1)]$

Step 3. Calculate $f_{\mathrm{JY}}(G')$ to get the secret data.

2.4. Generalized Exploiting Modification Direction (GEMD) Algorithm. In 2013, Kuo and Wang proposed the GEMD algorithm, which is an extended version of the IEMD scheme. The embedding payload of GEMD can reach 1.5 bpp. The inputs of GEMD are n adjacent pixels $G(g_1, g_2, \ldots, g_n)$ and decimal digit $S_{(10)}$ which is transformed from $n+1$ binary bit secret data. The output is n stego-pixels $G'(g_1', g_2', \ldots, g_n')$.

Step 1. Calculate $f_{\mathrm{GEMD}}(\mathrm{G}) = [\sum_{i=1}^{n} (2^i - 1) \times g_i] \bmod 2^{n+1}$.

Step 2.$G'(g_1', g_2', \ldots, g_n')$ is obtained as follows.

$G' = \mathrm{G}$, $D = (S_{(10)} - f_{\mathrm{GEMD}}(G)) \bmod 2^{n+1}$.

if $\mathrm{D} == 2^n$, then $g_1' = g_1 + 1$, $g_n' = g_n + 1$

else if $0 < \mathrm{D} < 2^n$, transform D to $(d_n \ldots d_1 d_0)_2$ and for $i = n$ down to 1 do

if $d_i == 0$ and $d_{i-1} == 1$, then $g_i' = g_i + 1$

if $d_i == 1$ and $d_{i-1} == 0$, then $g_i' = g_i - 1$

else if $2^n < \mathrm{D} < 2^{n+1}$, transform $(2^{n+1} - \mathrm{D})$ to $(d_n \ldots d_1 d_0)_2$, for $i = n$ down to 1 do

if $d_i == 0$ and $d_{i-1} == 1$, then $g_i' = g_i - 1$

if $d_i == 1$ and $d_{i-1} == 0$, then $g_i' = g_i + 1$

Step 3. Calculate $f_{\mathrm{GEMD}}(G')$ to get the secret data.

2.5. KKWW. KKWW was proposed by Kuo et al. in 2015. It is a high-capacity data hiding scheme based on multi-bit encoding function. The embedding payload of this scheme is up to 4.25 bpp. The details of KKWW are shown as follows. The inputs of KKWW are n adjacent pixels $G(g_1, g_2, \ldots, g_n)$ and decimal data $S_{(10)}$ which is transformed from $nk+1$ binary bit secret data. The output is n stego-pixels $G'(g_1', g_2', \ldots, g_n')$.

Step 1. Calculate $f_{\mathrm{KKWW}}(G) = [\sum_{i=1}^{n} c_i \times g_i] \bmod 2^{nk+1}$, where

$$c_i = \begin{cases} 1, i = 1 \\ 2^k c_{i-1} + 1, i \neq 1 \mathrm{and} i > 0 \end{cases}.$$

Step 2. $G'(g_1', g_2', \ldots, g_n')$ is obtained as follows.

$G' = \mathrm{G}$, $D = (S_{(10)} - f_{\mathrm{KKWW}}(G)) \bmod 2^{nk+1}$

if $D = 0$, $G' = G$

if $D = 2^{nk}$, $g_n' = g_n + (2^k - 1)$, $g_1' = g_1 + 1$

if $D < 2^{nk}$

transform D to $(d_{n-1} \ldots d_1 d_0)_{(2^k)}$ and for each i in $n(n-1, n-2, \ldots, 0)$ do

$g_{i+1}' = g_{i+1} + d_i$; $g_i' = g_i - d_i$, if $i > 0$

if $D > 2^{nk}$

transform $(2^{nk+1} - \mathrm{D})$ to $(d_{n-1} \ldots d_1 d_0)_{(2^k)}$ and for each i in $n(n-1, n-2, \ldots, 0)$ do

$g_{i+1}' = g_{i+1} - d_i$; $g_i' = g_i + d_i$, if $i > 0$

Step 3. Calculate $f_{\mathrm{KKWW}}(G')$ to get the secret data.

2.6. SB. In 2019, Sairam proposed a high-capacity information hiding scheme SB. The embedding payload of SB can reach 3 bpp. More detailed steps are shown below. The inputs of SB are pixel $G(g)$ and n^2-ary notational system digit $S_{(n^2)}$ which is transformed from n binary bit secret data. The output is stego-pixels $G'(g')$.

Step 1. By the steps below to find the variable x and obtain g' for $(-\lfloor n^2/2 \rfloor \le x \le \lfloor n^2/2 \rfloor)$ {

$f = (g + x) \bmod n^2$

if $(f == S_{(n^2)})$ {

$g' = g + x$

break}}

Step 2. Calculate $S_{(n^2)} = g' \bmod n^2$ to get the secret data.

2.7. Modulus Calculations on Prime Number (MOPNA) Algorithm. In 2020, Zhang et al. proposed MOPNA, a method for modulo computation using prime numbers $C(c_0, c_1)$. This method uses a function (changes in pixel values, CPV) to measure the effect of embedding. The maximum embedding payload of MOPNA is 3.5 bpp. More details are shown as follows. The inputs of MOPNA are a pixel pair $G(g_1, g_2)$ and decimal digit data $S_{(10)}$ which is transformed from $2n+1$ binary bit secret data. The output is stego-pixels $G'(g_1', g_2')$.

Step 1. Calculate $f_{\mathrm{MOPNA}}(\mathrm{G}) = (C * G^T) \bmod 2^{2n+1}$.

Step 2. Choose the variable X that satisfies the following equation and $G' = G + X$:

$X = \{(x_0, x_1)|$
$d = f(G+X) \mathrm{and} \forall X', X' \neq X$: $CPV(G + X, G) \ge CPV(G + X', G) \mathrm{and} x_0, x_1 \epsilon [-255, 255]\}$

Step 3. Calculate $f_{\mathrm{MOPNA}}(G')$ to get the secret data.

3. Proposed Method

3.1. Multiple Pixels-Value Adjustment with Encoding Function (MPA) Algorithm. Inspired by EMD, IEMD, and KKWW, we design a new scheme MPA to embed secret data into cover images. MPA divides cover image into pixel groups that include n pixels. In the specific embedding process, n pixels are regrouped to get two sub-groups. After that, $nk+2$ bits of secret data converted to decimal are grouped accordingly using (1) and embedded into the subgroups. The MPA algorithm can embed up to 5-bit secret data in each

pixel and ensure the quality of the stego-image. The detail of the algorithm is described as follows. Figure 1 shows the flowchart of the proposed Algorithm 1 for embedding process. The inputs of MPA are n adjacent pixels $G(g_1, g_2, \ldots, g_n)$ and decimal data $S_{(10)}$ which is transformed from $nk+2$ binary bit secret data s, where n is the number of pixels in a group and k is used to adjust the degree of secret data embedding. The output is n stego-pixels $G'(g_1', g_2', \ldots, g_n')$.

According to embedding algorithm, the secret data are embedded in the cover image to get the stego-image. Stego-image is transmitted from the sender to the receiver. The receiver extracts the secret information from the stego-image by the following extraction process. Figure 2 shows flowchart of data extracting process.

The inputs are n stego-pixels $G_1'(g_1', g_2', \ldots, g_{m1}')$ and $G_2'(i_1', i_2', \ldots, i_{m2}')$. And the output is secret data s.

Step 1. Compute the secret $\beta' = f(G_1') = f(g_1', g_2', \ldots, g_{m1}')$ and $\alpha' = f(G_2') = f(i_1', i_2', \ldots, i_{m2}')$.

Step 2. Convert the value of β' and α' to binary and combine them to get s.

Step 3. Repeat from steps 30 to step 31 until all secret data are extracted.

We use an example to illustrate the embedding process and the extraction process. The specific process is described below (Algorithms 2 and 3).

3.2. Math Proof: The MPA Can Embed $nk+2$-Bit Secret Data into n Cover Pixels. In this section, we prove that $nk+2$-bit secret data can be split according to the grouping of n cover pixels. In our scheme, the n cover pixels are divided into two groups $G_1(g_1, g_2, \ldots, g_{m_1})$, $G_2(i_1, i_2, \ldots, i_{m_2})$, and $m_1 + m_2 = n$.

$\because$ When s has $nk+2$ secret data, the decimal $S_{(10)} \in (0, 2^{nk+2})$.

$\because 2^{nk+2} = 2^{(m_1+m_2)k+2} = 2^{m_1k+1} + 2^{m_2k+1}$, $\alpha < 2^{m_2k+1}$.

$\therefore$ It is easy to know that $S_{(10)}$ can be uniquely represented, $S_{(10)} = \alpha \times 2^{m_1k+1} + \beta$, where α, β are integers, and $\beta < 2^{m_1k+1}$.

$\therefore \alpha$, β can be embedded into m_2k+1 and m_1k+1 cover pixels, respectively.

3.3. Math Proof: MPA Can Embed Secret Data into Cover Image Correctly. In this section, we need to prove that regardless of the values of β and G_1, the values of D must satisfy the four embedding conditions in Step 6. When $m_1 = 2$ and $k = 3$, the secret data $s \in [0000000, 1111111]_2$ and convert to decimal $\beta \in [0, 127]_{10}$.

$\because$ With the modulus operator, $(a+b)\text{mod}c = ((amo\ dc) + (bmo\ dc))\text{mod}c$.

$\therefore D = (\beta - f(G_1))\text{mod}2^{m_1k} + 1 = ((\beta\text{mod}128) - (f(G_1)\text{mod}128))\text{mod}128$.

$\therefore$ Let $y = \beta\text{mod}128$ and $z = f(G_1)\text{mod}128$, and the original equation can be changed to $D = (y - z)\text{mod}128$.

$\because \beta \in [0, 127]$.

$\therefore y \in [0, 127]$, which is an uncertain variable.

$\because f(G_1) = [\sum_{i=1}^{n} c_i \times g_i]\text{mod}2^{m_1k+1} = [1 \times g_1 + 9 \times g_2]\text{mod}128$, where g_1, $g_2 \in [0, 255]$, and they are two pixels with definite values.

$\therefore f(G_1)$ is a constant value and $f(G_1) \in [0, 127]$.

$\therefore z \in [0, 127]$, and z is a definite fixed value. $\because D = (y - z)mo\ d128$, y is an uncertain variable taking values from [0, 127], and z is a definite fixed value taking values in the range [0, 127]. D must take values in the range [0, 127]. The range of D must satisfy the four embedding conditions in step 6, so all the secret data must be able to be embedded in the cover image.

3.4. Math Proof: MPA Can Extract Secret Data from Stego-Image Correctly. In the previous subsection, we have proved that the range of D must satisfy the four embedding conditions in step 6. In this section, it is proved that the secret data extracted β' from the equation must be equal to the embedded secret data β. That is, $\beta' = f(G_1') = f(g_1', g_2', \ldots, g_m 1') = \beta$. When $m_1 = 2$ and $k = 3$,

(1) if $D = 0$, then $g_i' = g_i$

$$\therefore \beta' = f(G_1') = f(g_1, g_2) = \beta, \tag{1}$$

(2) if $D = 2^{m_1k} = 64$, then $g_2' = g_2 + (2^3 - 1)$, $g_1' = g_1 + 1$

$$\begin{aligned}\therefore \beta' &= f(G_1') = (1 \times g_1' + 9 \times g_2')\text{mod}2^7 \\ &= \left(1 \times (g_1 + 1) + 9 \times \left(g_2 + \left(2^3 - 1\right)\right)\right)\text{mod}2^7 \\ &= \left((1 \times g_1 + 9 \times g_2)\text{mod}2^7 + 64\text{mod}2^7\right)\text{mod}2^7 \\ &= \left(f(G_1)\text{mod}2^7 + 64\right)\text{mod}2^7\end{aligned}$$

$$\begin{aligned}&\because f(G_1) \in [0, 127] \\ &\therefore \beta' = f(G_1') = f(G_1) + 64 = f(G) + D \\ &\quad = f(G) + (\beta - f(G)) = \beta,\end{aligned} \tag{2}$$

(3) if $D < 2^{m_1k}$, then transform D to $(d_1d_0)_8$, and $g_1' = g_1 - d_1 + d_0$, $g_2' = g_2 + d_1$

$$\begin{aligned}\therefore \beta' &= f(G_1') = (1 \times g_1' + 9 \times g_2')\text{mod}2^7 \\ &= (1 \times (g_1 - d_1 + d_0) + 9 \times (g_2 + d_1))\text{mod}2^7 \\ &= \left((1 \times g_1 + 9 \times g_2)\text{mod}2^7 + (8d_1 + d_0)\right)\text{mod}2^7,\end{aligned} \tag{3}$$

$\because (8d_1 + d_0)$ is to convert (d_1d_0) from octal to decimal, and $f(G_1') \in [0, 127]$

$$\begin{aligned}\therefore \beta' &= f(G_1') = (f(G_1) + D)\text{mod}2^7 \\ &= f(G_1) + (\beta - f(G)) = \beta,\end{aligned} \tag{4}$$

(4) if $D > 2^{m_1k}$, then transform $D' = (2^{m_1k+1} - D)$ to $(d_1d_0)_8$ and $g_1' = g_1 + d_1 - d_0$, $g_2' = g_2 - d_1$

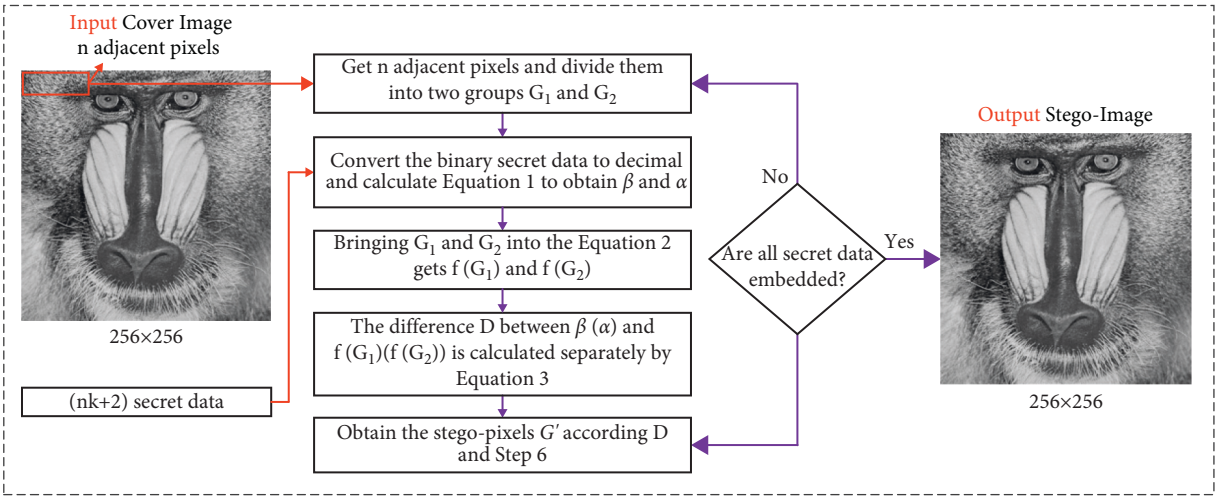

Figure 1: Flowchart of the proposed algorithm for data embedding process.

Step 1. Divide n adjacent pixels into two groups $G_1(g_1, g_2, \ldots, g_{m_1})$ and $G_2(I_1, I_2, \ldots, I_{m_2})$, where $m_1 + m_2 = n$.
Step 2. Transform s into decimal $S_{(10)}$ by formula $S_{(10)} = (2^{nk+1} \times s_{nk+1} + 2^{nk} \times s_{nk} + \cdots + 2^1 \times s_1 + s_0)$.
Step 3. Construct the Algorithm 1, where α, β are integers and $\beta < 2^{m_1 k+1}$.

$$S_{(10)} = \alpha \times 2^{m_1 k+1} + \beta$$

Step 4. Bring G_1 into Algorithm 1 and obtain $f(G_1)$

$$f(G_1) = [\sum_{i=1}^{m_1} c_i \times g_i] \bmod 2^{m_1 k+1}$$

$$\text{Where } c_i = \begin{cases} 1, i = 1 \\ 2^k c_{i-1} + 1, i \neq 1 \text{and} i > 0 \end{cases}$$

Step 5. Calculate the difference D between β into $f(G_1)$ by Algorithm 1.

$$D = (\beta - f(G_1)) \bmod 2^{m_1 k+1}$$

Step 6. $G_1'(g_1', g_2', \ldots, g_{m1}')$ is obtained by embedding β into G_1 in the following steps.

$G_1' = G_1$
if $D = 0$
  $G_1' = G_1'$
if $D = 2^{m_1 k}$
  $g_{m_1}' = g_{m_1} + (2^k - 1)$, $g_1' = g_1 + 1$
if $D < 2^{m_1 k}$
  transform D to $(d_{m_1-1} \ldots d_1 d_0)_{(2^k)}$, for each i in $(m_1 - 1, m_2 - 2, \ldots, 0)$ do
    $g_{i+1}' = g_{i+1} + d_i$
    $g_i' = g_i - d_i$, if $i > 0$
if $D > 2^{m_1 k}$
  transform $(2^{m_1 k+1} - D)$ to $(d_{m_1-1} \ldots d_1 d_0)_{(2^k)}$, for each i in $(m_1 - 1, m_2 - 2, \ldots, 0)$
    $g_{i+1}' = g_{i+1} - d_i$
    $g_i' = g_i + d_i$, if $i > 0$

Step 7. Bring G_2 into Algorithm 1 to calculate $f(G_2)$ and get the difference D between α and $f(G_2)$ by $D = (\alpha - f(G_2)) \bmod 2^{m_2 k+1}$. Finally, repeat step 6 to obtain $G_2'(i_1', i_2', \ldots, i_{m2}')$.
Step 8. Merge G_1' and G_2' to get G'.
Step 9. Repeat from steps 20 to 27 until all secret data are embedded.

Algorithm 1: Embedding algorithm of MPA.

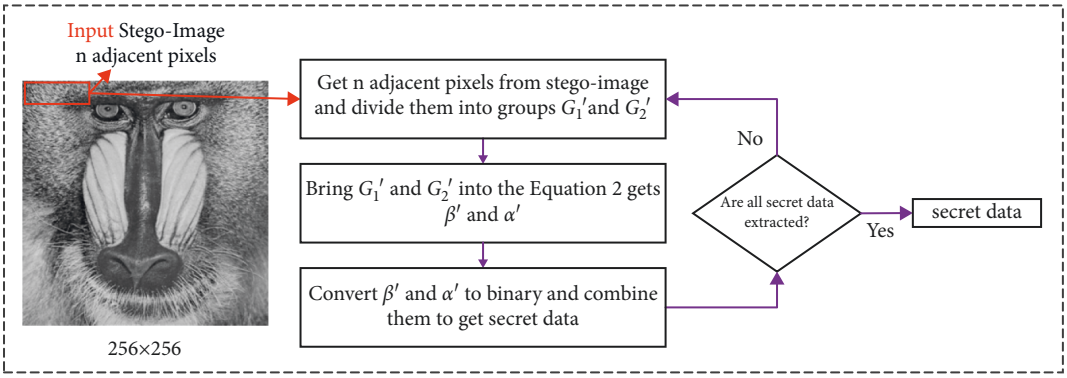

Figure 2: Flowchart of the proposed algorithm for data extracting process.

Input 4 adjacent pixels $G(10, 19, 5, 9)$ and $nk+2 = 14$ binary secret data s $(10101011011011)_2$.
Output 4 stego-pixels $G'(12, 23, 4, 9)$.
Step 1. Divide 4 adjacent pixels into two groups $G_1(10, 19)$ and $G_2(5, 9)$.
Step 2. Transform s into decimal $S_{(10)} = 10971$.
Step 3. Construct the $10971 = 85 \times 2^7 + 91$ to get $\alpha = 85$ and $\beta = 91$.
Step 4. Bring G_1 and G_2 into Equation 2 to calculate $f(G_1) = f(10, 19) = (1 \times 10 + 19 \times 9) \bmod 2^7 = 53$, $f(G_2) = f(5, 9) = (1 \times 5 + 9 \times 9)\, mo\, d\, 2^7 = 86$.
Step 5. Calculate $D = \beta - f(G_1) = (91 - 53) \bmod 2^7 = 38$ by Algorithm 1.
Step 6. $G_1'(12, 23)$ is obtained by embedding β into G_1 in the following steps.
$D = 38 < 2^{m_1 k} = 64$,
transform 38 to $(46)_{(8)}$, $(d_1, d_0) = (4, 6)$,
for $d_1 = 4$, $g_2' = g_2 + d_1 = 19 + 4 = 23$, $g_1' = g_1 - d_1 = 10 - 4 = 6$, $(g_2, g_1) = (23, 6)$,
for $d_0 = 6$, $g_1' = g_1 + d_0 = 6 + 6 = 12$,
get $G_1'(12, 23)$.
Step 7. Bring G_2 into Algorithm 1 to calculate $f(G_2) = 127$ and get the difference D between α and $f(G_2)$ by $D = \alpha - f(G_2) = (85 - 86) \bmod 2^7 = 127$.
$D = 127 > 2^{m_1 k} = 64$,
transform $(2^{m_1 k+1} - D) = 128 - 127 = 1$ to $(1)_{(8)}$, $(d_1, d_0) = (0, 1)$,
for $d_1 = 0$, $i_2' = 9$, $i_1' = 5$,
for $d_0 = 1$, $i_1' = 4$,
get $G_2'(4, 9)$.
Step 8. Merge G_1' and G_2' to get $G'(12, 23, 4, 9)$.

Algorithm 2: Embedding example of MPA.

$$
\begin{aligned}
\therefore \beta' &= f(G_1') = f(g_1', g_2') = (1 \times g_1' + 9 \times g_2') \bmod 2^7 \\
&= (1 \times (g_1 + d_1 - d_0) + 9 \times (g_2 - d_1)) \bmod 2^7 \\
&= \left((1 \times g_1 + 9 \times g_2) \bmod 2^7 - (8d_1 + d_0)\right) \bmod 2^7 \\
\therefore \beta' &= f(G_1') = (f(G_1) - D') \bmod 2^7 \\
&= \left(f(G_1) - 2^7 + D\right) \bmod 2^7 \\
&= \left((f(G_1) + D) \bmod 2^7 - 2^7 \bmod 2^7\right) \bmod 2^7 \\
&= (f(G_1) + D) \bmod 2^7 \\
&= (f(G_1) + (\beta - f(G_1))) \bmod 2^7 = \beta.
\end{aligned} \tag{5}
$$

Therefore, the algorithm can recover the secret data from stego-images.

3.5. Experimental Results and Security Analysis. In this section, we test the proposed algorithm and present the results. Our method and other algorithms are implemented in Python. And the hardware conditions for the experiments are based on a personal PC with Intel(R) Core(TM) i7-10700 CPU @ 2.90 GHz and 16-GB RAM. The operating system is Windows 10 Professional 64 bits, and the experiment software is PyCharm. The images used for the experiments are a series of grayscale images of standard size $256 * 256$ (Baboon, Barbara, Boat, F-16, Goldhill, Lena, Pepper, and Tiffany) [19], which are shown in Figure 3.

The performance of method in this paper includes stego-image quality, payload, PSNR, SSIM, QI, and several steganalysis test.

3.6. The Quality of Stego-Image. In this experiment, the number of secret data (NCD) bits is set as 262144 bits and 327680 bits. When $n = 2$ and $k = 3$, the payload of MPA is 4 bpp. There are $256 \times 256 = 65536$ pixels available for

Input Stego-pixels $G_1'(12, 23)$ and $G_2'(9, 4)$.
Output Secret data s(10101011011011).
Step 1. Compute the secret $\beta' = f(G_1') = f(12, 23) = 91$ and $\alpha' = f(G_2') = f(4, 9) = 85$.
Step 2. Convert the value of β' and α' to binary and combine them to get s(10101011011011).

Algorithm 3: Extraction example of MPA.

Figure 3: Eight $256 * 256$ grayscale images (Baboon, Barbara, Boat, F-16, Goldhill, Lena, Pepper, and Tiffany).

Figure 4: The eight stego-images under the numbers of secret data bits are 262144 bits (bpp = 4).

embedding secret data. The largest number of secret data bits that can be embedded on a cover image is $65536 \times 4 = 262144$. Another set of NCD data is chosen with bpp = 5. Figure 4 shows the stego-images when the length of secret data is 262144 bits. Figure 5 shows the stego-images when the length of secret data is 327680 bits.

It is clearly shown that it is difficult to detect the difference between the cover images and the stego-images with

Figure 5: The eight stego-images under the numbers of secret data bits are 327680 bits (bpp = 5).

the human visual system. Hiding the information in the image to ensure that it will not be detected by the attacker is exactly what data hiding techniques do.

3.7. Payload. Payload is the number of secret data bits that can be embedded into per pixel (bpp). The straightforward formula is defined in

$$\text{Payload} = \frac{\text{allsecredatabits}}{\text{allcoverpixels}}\,(bpp). \tag{6}$$

A higher bpp represents more secret data that can be embedded in a pixel. On the contrary, a low bpp represents poor embedding efficiency. The embedding scheme's characteristics including adjacent pixel in a group, number of embedding bits for each group, and payload (bpp) are summarized in Table 1. In this table, n is the number of pixels in a group and k is used to adjust the degree of secret data embedding. For MPA, $n = 2$, $k = 2,3,4$, and $n = 4$, $k = 2,3,4$ in experiments.

The number of adjacent pixels in a group is different for each data hiding scheme. Three are n, 2, 1, n, n, 1, 2, n adjacent pixels in a group of EMD, IEMD, JY, GEMD, KKWW, SB, MOPNA, and MPA, respectively. EMD, IEMD, JY, GEMD, KKWW, SB, MOPNA, and MPA can carry $\log_2(2n+1)$, 3, 2, $n+1$, $nk+1$, k, $2k+1$, and $nk+2$ bits. And then, the specific payload of the above algorithm is shown in Table 1.

We calculate their specific values and express them in Figure 6. In Figure 6, it can be observed that the bpp of the algorithms tends to increase as n or k decreases. And we can find that the MPA algorithm can achieve a maximum embedding payload of 5 bpp.

3.8. PSNR, SSIM, and QI. PSNR is a major metric in the field of information security, and many studies use it to evaluate the performance of information hiding methods. PSNR > 40 dB means the difference between cover image and stego-image is small and the secret information hidden in it is not easily detected. 40 dB > PSNR > 30 dB means that the quality of stego-image is acceptable. PSNR < 30 dB means that the quality of stego-image is poor and the secret information hidden in it has been detected and attacked. In the field of information hiding, the secret data hidden in the stego-image is not perceived by the human visual system when PSNR > 30 dB. The equation of PSNR is shown in equations (5) and (6).

$$\begin{aligned}\text{PSNR} &= 10\,\log_{10}\left(\frac{255^2}{\text{MSE}}\right)(\text{dB})\\ \text{MSE} &= \frac{1}{M\times N}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(g_{ij}-g_{ij}'\right),\end{aligned} \tag{7}$$

where PSNR is measured in dB, M and N represent the length and width of the image, g_{ij} represents the pixel value at position (i, j) of the cover image, and g_{ij}' represents the pixel value at position (i, j) of the stego-image. SSIM is another metric used to show the similarity between the cover image and the stego-image. The value of SSIM is between 0 and 1. The closer to 1 means that the stego-image is similar to the cover image. Its calculation is presented in equation (7).

$$\text{SSIM}(x, y) = \frac{\left(2\mu_o\mu_s + c1\right)\left(2\sigma_{os} + c2\right)}{\left(\mu_o^2 + \mu_s^2 + c1\right)\left(\sigma_o^2 + \sigma_s^2 + c2\right)}, \tag{8}$$

Table 1: Eight algorithms' payload.

	Number of adjacent pixels in a group	Embedding bits for each group	Payload (bpp)
EMD	n	$\log_2(2n+1)$	$\log_2(2n+1)/n$
IEMD	2	3	1.5
JY	1	2	2
GEMD	n	$n+1$	$(n+1)/n$
KKWW	n	$nk+1$	$(nk+1)/n$
SB	1	K	k
MOPNA	2	$2k+1$	$(2k+1)/2$
MPA	n	$nk+2$	$(nk+2)/n$

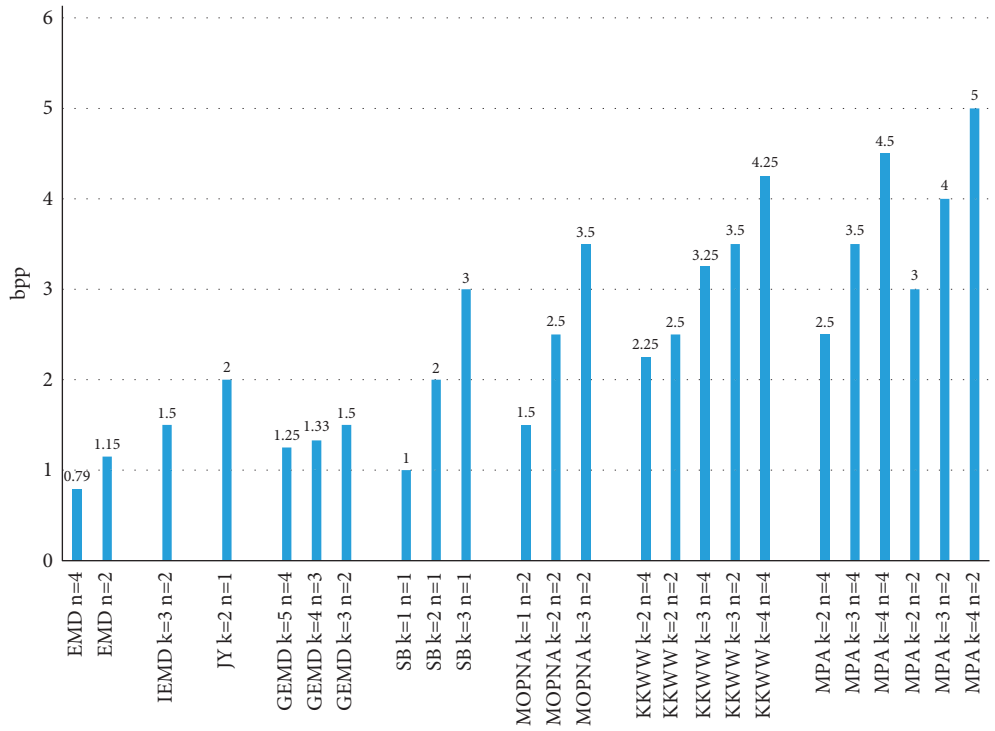

Figure 6: Compare eight algorithms' payload.

where o and s represent the cover image and the stego-image. $c1$ and $c2$ are the constants. μ_o and μ_s are the mean. σ_{os} is the covariance. μ_o^2 and μ_s^2 are variances, and σ_o^2 and σ_s^2 are the standard deviations for the corresponding o and s. QI is used to measure the equivalence of the stego-image with the cover image, and its value is estimated using equation (8) [20].

$$\mathrm{QI} = \frac{4 \times \overline{P} \times \overline{Q} \times \left\{\sum_{i=1}^{m}\sum_{j=1}^{n}\left(g_{ij}-\overline{P}\right)\times\left(g'_{ij}-\overline{Q}\right)\right\}}{\left\{\sum_{i=1}^{m}\sum_{j=1}^{n}\left(g_{ij}-\overline{P}\right)^2+\sum_{i=1}^{m}\sum_{j=1}^{n}\left(g'_{ij}-\overline{Q}\right)^2\right\}\times\left\{(\overline{P})^2+(\overline{Q})^2\right\}}, \tag{9}$$

where $\overline{P}$ represents the average pixel value of the cover image and $\overline{Q}$ represents the average pixel value of the stego-image.

In order to compare and analyze the comprehensive performance of the algorithms proposed, we compare MPA with a total of seven other algorithms. In our experiments, the NCD is set as 49000 bits and 98305 bits, respectively. Among all the compared algorithms, EMD has the lowest embedding capacity. The payload of the EMD is 0.79 bpp when there are 4 adjacent pixels in each group ($n = 4$). There are $256 \times 256 = 65536$ pixels available for embedding secret data. The maximal number of bits of secret data which can be embedded is $\lfloor 65536/2 \rfloor \times \lfloor 0.79 \times 4 \rfloor = 49152$. For comparison with EMD, NCD = 49000 is selected for comparison.

When NCD = 49000 bits, the PSNR, SSIM, and QI of the eight algorithms are shown in Table 2. It is obvious that when $n = 4$, EMD has the highest PSNR = 54.66 dB, but it has the lowest payload. IEMD (1.5 bpp), JY (2 bpp), and GEMD (highest payload 1.5 bpp) have better PSNR performance, but the highest payload is lower than MPA. For methods such as KKWW, SB, MOPNA, and MPA, lower k values bring lower embedded payload but good PSNR performance; on the contrary, higher k values bring higher embedded payload but poor PSNR performance. When $n = 1$, $k = 3$, SB has the highest payload of 3 bpp, but PSNR = 39.89 dB performs less than MPA (3 bpp, PSNR = 42.54 dB). When $n = 2$, $k = 2$, MOPNA has a payload of 2.5 bpp, but PSNR = 42.79 dB performs less than MPA

Table 2: Experimental results when NCD = 49000 bits.

Method	Baboon	Barbara	Boat	F-16	Goldhill	Lena	Pepper	Tiffany	Average	SSIM (Average)	QI (Average)
EMD $n=2$	52.12	52.12	52.11	52.10	52.10	52.10	52.10	52.12	52.11	0.9999	0.9999
EMD $n=4$	54.67	54.64	54.65	54.64	54.64	54.68	54.65	54.67	54.66	0.9999	0.9999
IEMD	50.16	50.21	50.17	50.17	50.15	50.20	50.18	50.19	50.18	0.9999	0.9999
JY	41.43	41.45	41.42	41.43	41.46	41.46	41.44	41.45	41.44	0.9993	0.9992
GEMD	50.99	51.01	51.02	50.99	51.04	51.01	51.02	50.99	51.01	0.9982	0.9999
KKWW $n=2$ $k=2$	43.15	43.15	43.16	43.09	43.14	43.23	43.11	43.16	43.15	0.9993	0.9997
KKWW $n=2$ $k=3$	37.29	37.27	37.20	37.29	37.18	37.18	37.25	37.22	37.24	0.9974	0.9992
KKWW $n=4$ $k=2$	43.79	43.78	43.73	43.68	43.73	43.79	43.71	43.73	43.74	0.9991	0.9997
KKWW $n=4$ $k=3$	37.29	37.27	37.2	37.29	37.18	37.18	37.25	37.22	37.24	0.9973	0.9993
KKWW $n=4$ $k=4$	30.99	31.07	31.03	31.03	31.06	31.02	30.92	30.98	31.01	0.9877	0.9977
SB $n=1$ $k=2$	46.33	46.36	46.36	46.39	46.41	46.36	46.31	46.34	46.36	0.9970	0.9998
SB $n=1$ $k=3$	39.90	39.9	39.82	39.92	39.83	39.86	39.96	39.91	39.89	0.9898	0.9996
MOPNA $n=2$ $k=2$	43.94	43.97	43.89	43.93	43.96	43.95	34.94	34.76	42.79	0.9950	0.9998
MOPNA $n=2$ $k=3$	37.98	37.94	37.93	37.95	37.97	37.99	37.98	37.72	37.93	0.9868	0.9994
MPA $n=2$ $k=2$	42.57	42.55	42.52	42.54	42.54	42.51	42.57	42.49	42.54	0.9944	0.9998
MPA $n=2$ $k=3$	35.70	35.68	35.74	35.65	35.74	35.60	35.64	35.67	35.80	0.9819	0.9992
MPA $n=2$ $k=4$	30.01	30.05	30.02	29.99	30.00	30.01	30.01	30.02	30.01	0.9580	0.9972
MPA $n=4$ $k=2$	43.15	43.15	43.16	43.09	43.14	42.23	43.11	43.16	43.15	0.9941	0.9998
MPA $n=4$ $k=3$	36.60	36.61	36.63	36.55	36.60	36.59	36.55	36.53	36.58	0.9827	0.9993
MPA $n=4$ $k=4$	30.38	30.36	30.27	30.31	30.30	30.26	30.29	30.30	30.31	0.9597	0.9976

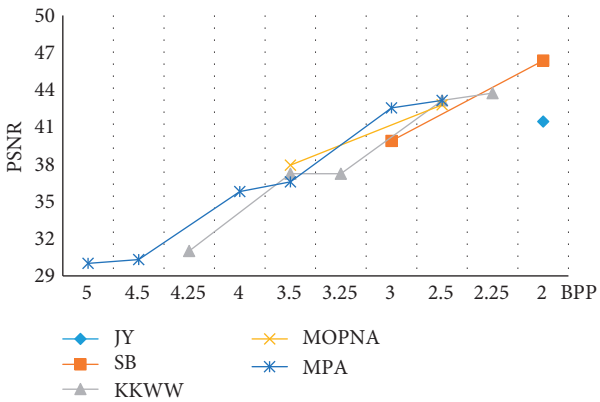

Figure 7: Compare PSNR of stego-image when NCD = 49000 bits.

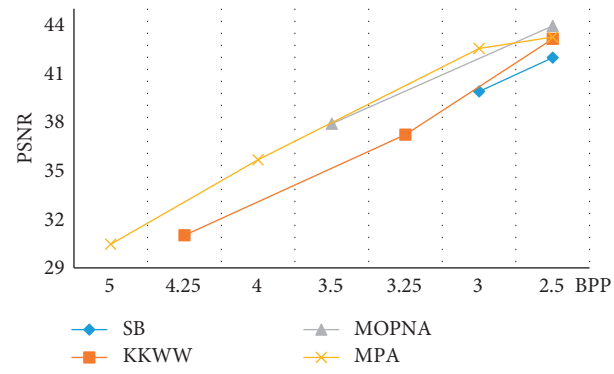

Figure 8: Compare PSNR of stego-image when NCD = 98305 bits.

Table 3: Experimental results when NCD = 98305 bits.

Method	Baboon	Barbara	Boat	F-16	Goldhill	Lena	Pepper	Tiffany	Average	SSIM (Average)	QI (Average)
JY	37.70	37.72	37.76	37.73	37.72	37.71	37.73	37.73	37.73	0.9994	0.9990
KKWW $n=2$ $k=2$	43.13	43.16	43.15	43.11	43.16	43.19	43.11	43.12	43.14	0.9994	0.9995
KKWW $n=4$ $k=3$	37.22	37.25	37.20	37.22	37.19	37.22	37.23	37.22	37.22	0.9976	0.9986
KKWW $n=4$ $k=4$	30.99	31.07	31.03	31.03	31.06	31.02	30.92	30.98	31.01	0.9891	0.9956
SB $n=1$ $k=2$	46.36	46.38	46.38	46.39	46.37	46.36	46.34	46.34	46.37	0.9951	0.9997
SB $n=1$ $k=3$	39.88	39.89	39.81	39.91	39.87	39.88	39.97	39.88	39.89	0.9819	0.9992
MOPNA $n=2$ $k=2$	43.92	43.94	43.93	43.94	43.97	43.95	43.93	43.92	43.94	0.9915	0.9996
MOPNA $n=2$ $k=3$	37.96	37.93	37.95	37.96	37.95	37.98	37.95	37.55	37.90	0.9769	0.9989
MPA $n=2$ $k=2$	42.57	42.55	42.52	42.54	42.54	42.51	42.57	42.49	42.54	0.9899	0.9996
MPA $n=2$ $k=3$	35.70	35.68	35.74	35.65	35.74	35.60	35.64	35.67	35.68	0.9655	0.9984
MPA $n=2$ $k=4$	30.50	30.49	30.51	30.20	30.45	30.56	30.50	30.52	30.46	0.9437	0.9944
MPA $n=4$ $k=2$	43.15	43.15	43.16	43.09	43.14	43.24	43.11	43.16	43.25	0.9898	0.9996
MPA $n=4$ $k=3$	36.59	36.62	36.60	36.57	36.61	36.57	36.56	36.56	36.59	0.9536	0.9991
MPA $n=4$ $k=4$	30.36	30.37	30.25	30.33	30.31	30.28	30.30	30.29	30.31	0.9432	0.9937

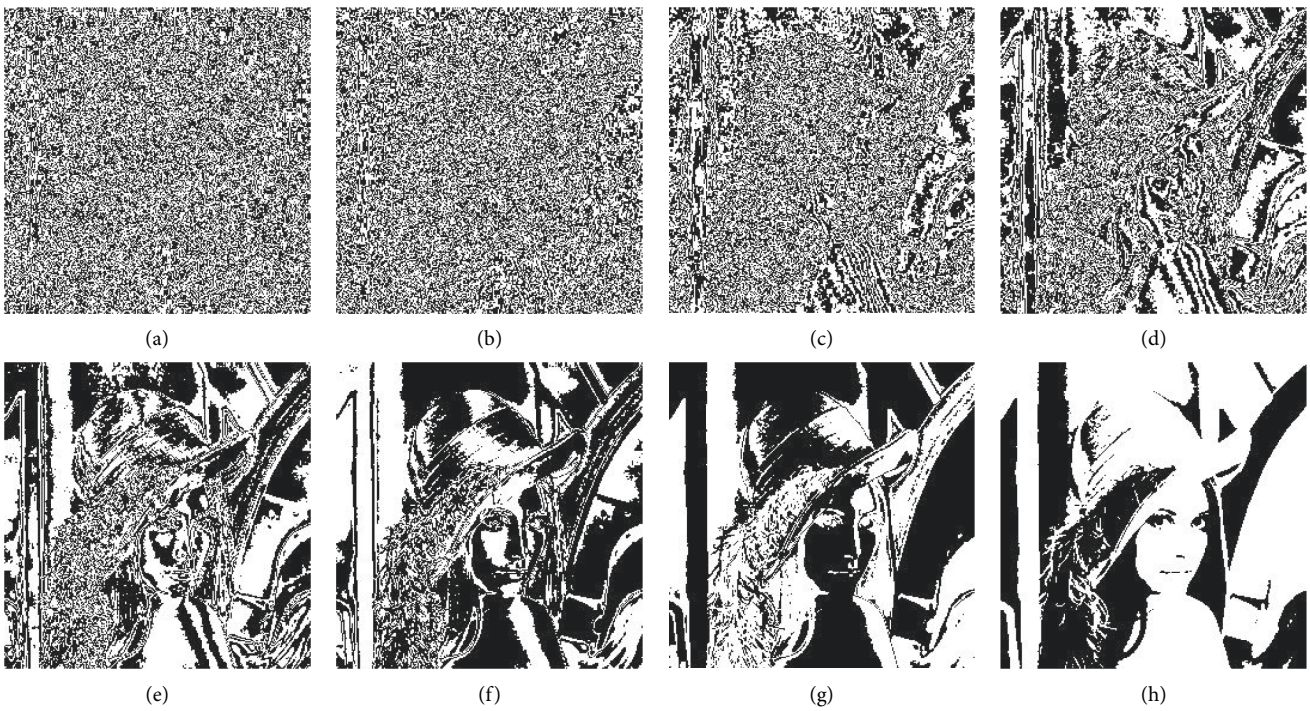

Figure 9: Bit-plane attack for cover image. (a) 0th bit. (b) 1th bit. (c) 2th bit. (d) 3th bit. (e) 4th bit. (f) 5th bit. (g) 6th bit. (h) 7th bit.

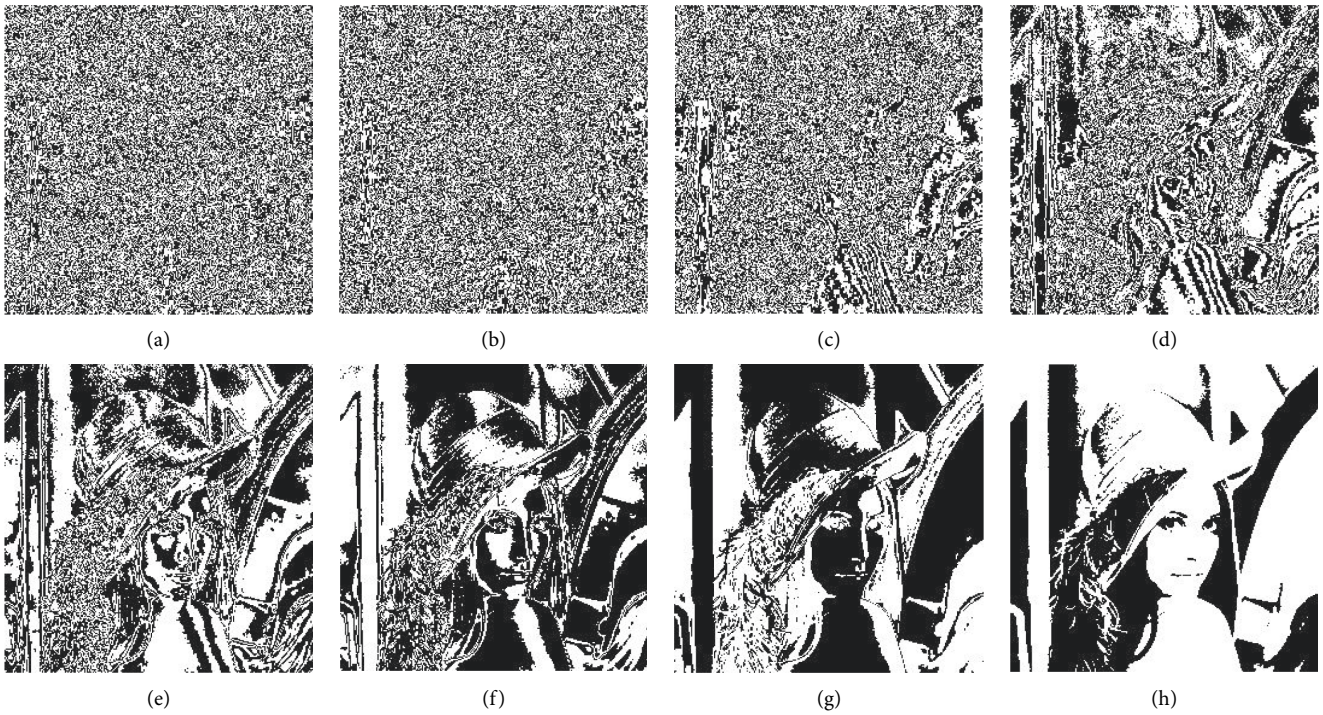

Figure 10: Bit-plane attack for stego-image when the number of secret data is 49000 bits. (a) 0th bit. (b) 1th bit. (c) 2th bit. (d) 3th bit. (e) 4th bit. (f) 5th bit. (g) 6th bit. (h) 7th bit.

(2.5 bpp, PSNR = 43.15 dB). When $n = 2$, $k = 3$, the highest payload of MOPNA is 3.5 bpp and PSNR = 37.93 dB. MPA performs slightly lower than MOPNA and KKWW (PSNR = 36.58 dB), but the highest payload of MPA is higher than MOPNA. The payload of KKWW is different from MPA when n is different from k. For a more intuitive comparison, we plot Figure 7. It can be observed that the PSNR curve of MPA generally performs better than that of KKWW from Figure 7. And only when MPA reaches the highest payload of 5 bpp, it still has good stego-image quality

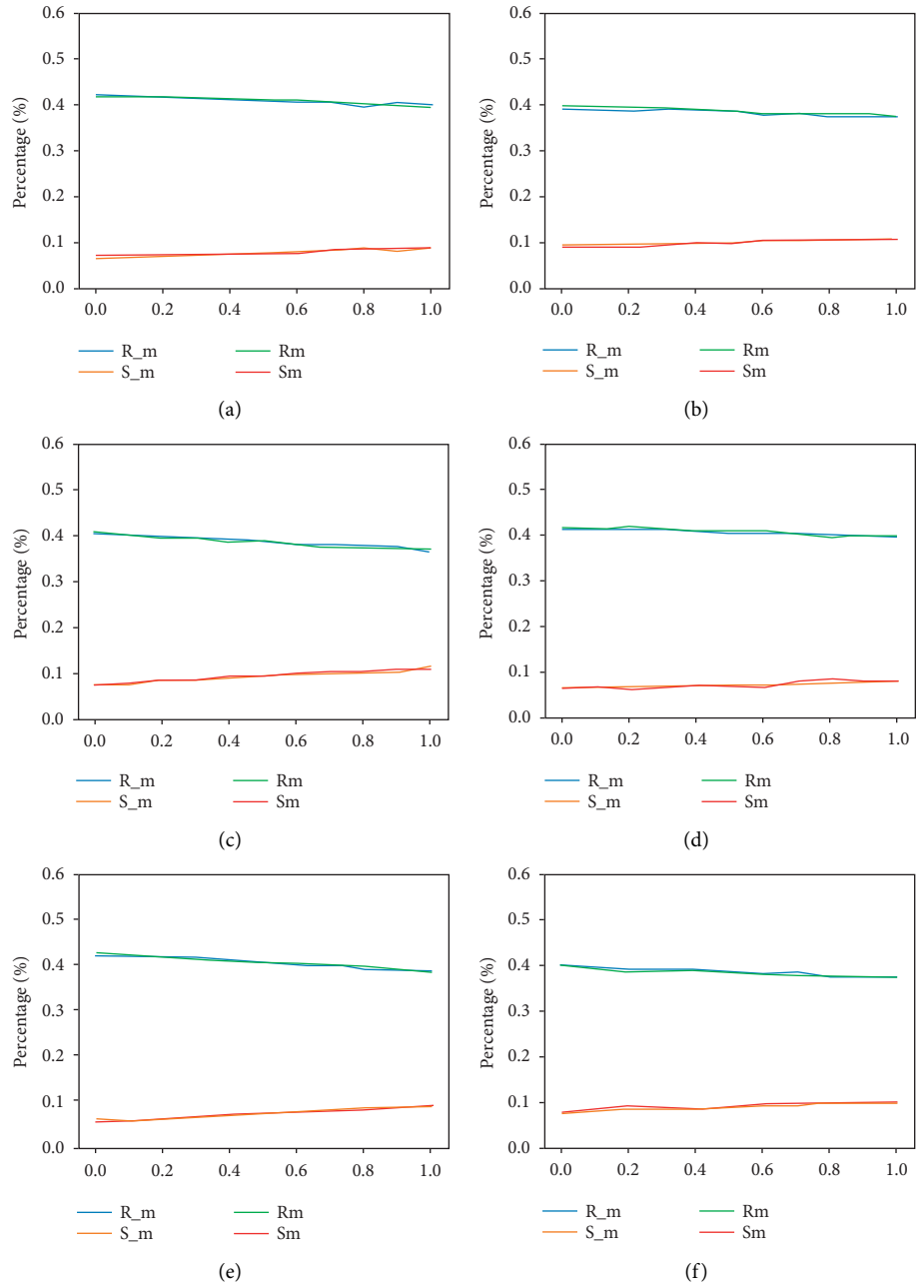

Figure 11: RS test to MPA with $n = 2,4$ and $k = 2, 3, 4$. (a) F-16 ($n = 2$, $k = 2$). (b) Barbara ($n = 2$, $k = 3$). (c) Boat ($n = 2$, $k = 4$). (d) Pepper ($n = 4$, $k = 2$). (e) Lena ($n = 4$, $k = 3$). (f).Tiffany ($n = 4$, $k = 4$).

(PSNR = 30.01 dB). In conclusion, MPA does better when the embedded payload and stego-image quality metrics are combined.

In addition to PSNR, the performance of MPA on SSIM and QI metrics also indicates that the quality of stego-images seems to be acceptable even though a large amount of secret data is hidden in the cover images. When $n = 2$, the SSIM values are 0.9944, 0.9819, and 0.9580 for $k = 2$ to 4, respectively, and the QI values are all around 0.99, which indicate that the quality of the stego-images is good. When $n = 4$, the SSIM values are 0.9941, 0.9827, and 0.9597 for $k = 2$ to 4, respectively, and the values of QI are around 0.99, so the stego-images of MPA are imperceptible in nature. No change can be found in the cover image and the stego-image.

When NCD = 98305 bits, the simulation results are shown in Table 3. In this case, 98305 bits of secret data cannot be embedded by IEMD, GEMD, and EMD. Only the scheme with a payload of at least 2bpp can embed 98305-bit secret data. As observed in Table 3, the highest payload of JY is 2 bpp with PSNR = 37.73 dB lower than the performance of MPA (when 2.5 bpp, PSNR = 43.25 dB). When $k = 2,3$, the PSNR of SB is lower than that of MPA. The PSNR of MOPNA is slightly higher than MPA when $k = 2,3$, but the maximum payload of this scheme is 3.5 bpp, which is lower

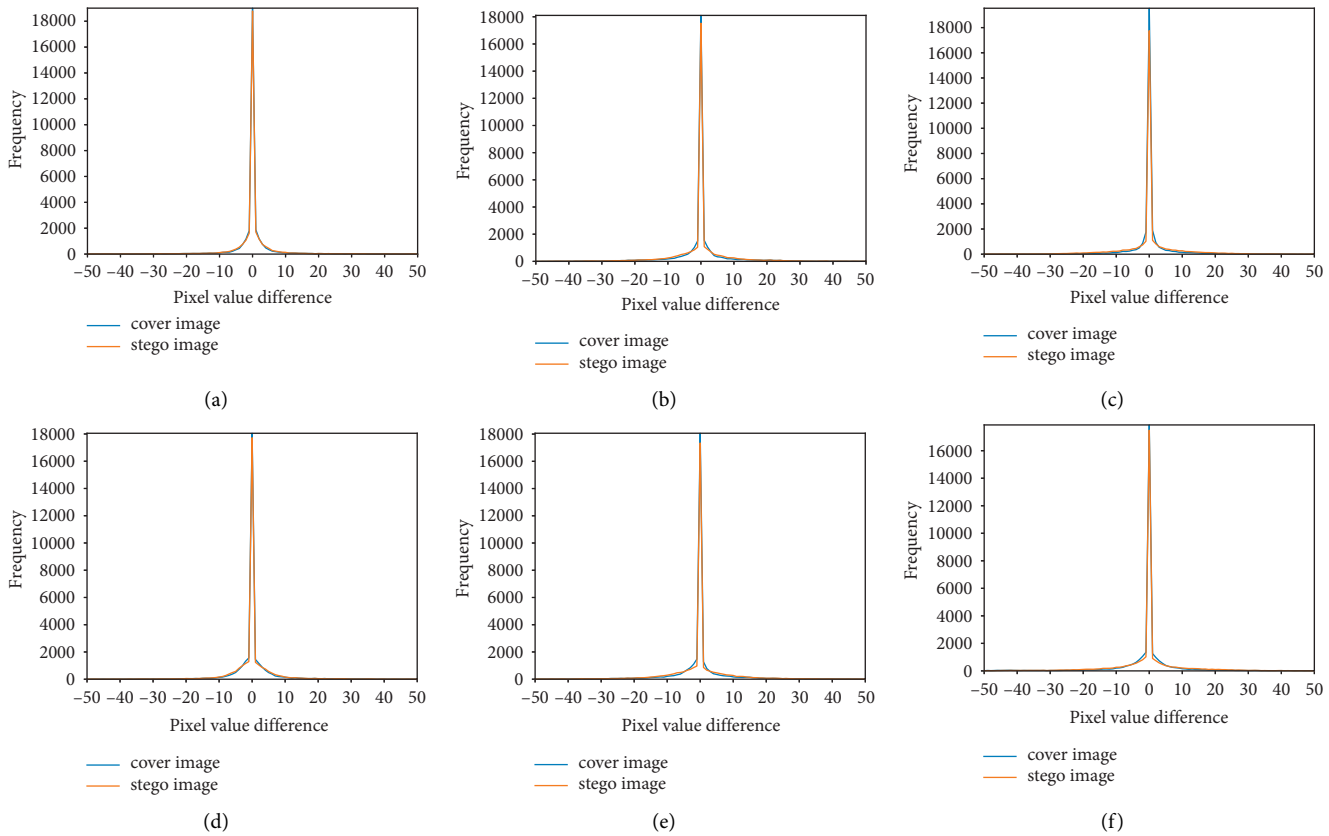

FIGURE 12: The PDH test to MPA on different images with $n = 2,4$ and $k = 2, 3, 4$. (a) F-16 ($n = 2$, $k = 2$). (b) Barbara ($n = 2$, $k = 3$). (c) Boat ($n = 2$, $k = 4$). (d) Pepper ($n = 4$, $k = 2$). (e) Lena ($n = 4$, $k = 3$). (f) Tiffany ($n = 4$, $k = 4$).

than MPA (5 bpp). To compare KKWW and MPA more intuitively, we plot Figure 8. From the figure, it can be observed that the PSNR curve of MPA performs better than KKWW in general. And only the MPA payload can reach 5 bpp, the quality of the stego-image is still good (PSNR = 30.31 dB). In summary, MPA does better when both payload and stego-image quality are considered.

When$n = 2$, the SSIM values of MPA are 0.9899, 0.9655,and 0.9437 for$k = 2$ to 4, respectively, and the values of QI are all around 0.99. When$n = 4$, the SSIM values of MPA were 0.9898, 0.9763,and 0.9567 for$k = 2$ to 4, respectively, and the values of QI were around 0.99. These indicate that it is difficult to find artificial traces in thestego-images generated by MPA.

3.9. Steganalysis. To further illustrate the security of the MPA scheme, we apply a bit-plane attack for testing. Bit-plane attack is a visual attack method to steal information by extracting the bit-plane of an image. In this section, bit-plane attack experiments are performed on cover image Lena and stego-image Lena embedded with 49000-bit secret data.

Comparing Figures 9 and 10, there is no difference between each bit plane of the stego-image and cover image after the bit-plane attack. That is, the attacker cannot steal the secret data through the bit-plane attack, which shows that the MPA scheme ensures the security of information hiding.

In RS steganalysis, n adjacent pixels $(g_1, g_2, \ldots, g_n)$ are as a group. Then, the discrimination function F, defined as $F(g_1, g_2, \ldots, g_n) = \sum_{i=1}^{n} |g_{i+1} - g_i|$, is used to quantify the smoothness or regularity of each pixel group. Finally, the flip function is applied to define three types of pixel groups: regular, singular, and unusable, represented by R, S, and U, respectively. The percentages of all groups of regular and singular with masks $M = [1\ 0\ 0\ 1]$ and $-M = [-1\ 0\ 0\ -1]$ are represented as R_m, R_{-m}, S_m, and S_{-m}. According the RS attack principle, if $R_{-m} - S_{-m} > R_m - S_m$, then steganography is detected. If $R_m \approx R_{-m} > S_m \approx S_{-m}$, then RS test fails to detect the steganography. Figure 11(a) depicts RS cures for F-16 image ($n = 2$, $k = 2$), Figure 11(b) depicts RS cures for Barbara image ($n = 2, k = 3$), Figure 11(c) depicts RS cures for Boat image ($n = 2$, $k = 4$), Figure 11(d) depicts RS cures for Pepper image ($n = 4$, $k = 2$), Figure 11(e) depicts RS cures for Lena image ($n = 4$, $k = 3$), and Figure 11(f) depicts RS cures for Tiffany image ($n = 4$, $k = 4$). The x-axis represents the rate of the total amount of data that MPA can embed under different conditions. For example, when $n = 4$, $k = 2$, and bpp = 2.5, the total amount of data that MPA can embed is NCD = $\lfloor (256 \times 256)/2 \rfloor \times \lfloor 2.5 \times 2 \rfloor = 163840$.

The y-axis represents the percentage of regular (R_m and R_{-m}) and singular (S_m and S_{-m}) groups. It can be observed from all the subfigures that $R_m \approx R_{-m} > S_m \approx S_{-m}$. Thus, the conclusion can be drawn that the RS attack fails to break the proposed scheme.

PDH analysis is performed by plotting the PDH graphs. As shown in Figure 12, PDH is a two-dimensional curve, where the x-axis represents the difference between two adjacent pixels in each group and the y-axis represents the incidence (frequency) of difference values. In general, the PDH plot of the cover image is kept smooth. If the PDH image of the stego-image has any zigzag shape, then steganography is detected; if not, steganography is not detected. Figure 12 depicts the PDH test of MPA on six cover images, (a) Airplane, (b) Barbara, (c) Boat, (d) Pepper, (e) Lena, and (f) Tiffany. In each of the six images, there are two curves. The blue curve is the cover image, and the orange curve is the stego-image. It can be seen from all the images that the orange curve does not have any zigzag nature. Therefore, the PDH test failed to reveal this proposed technique.

4. Conclusions

In this paper, a new data hiding algorithm MPA is proposed. It groups n adjacent pixels and completes the secret data grouping accordingly. The secret data are embedded into the pixels using a multi-bit encoding function in each group to achieve high embedding payload and high-quality stego-images. It can be observed from the experiments that the payload and PSNR perform better than the compared techniques. The SSIM and QI are acceptable, which implies that the stego-image generated by MPA is similar to the cover image. This does not make the attacker suspicious of the stego-image. From the results of the bit-plane attack, it is not possible to detect the secret information. 4 curves from the RS test fit the conditions of the attack. Similarly, the PDH curves of the stego-image do not appear zigzag and are consistent with the cover image, which does not disclose the steganographic technique. In the future, we will try to use this algorithm to investigate more applications such as watermarking and image authentication.

Data Availability

The data and code used to support MPA scheme are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] A. K. Sahu and M. Sahu, "Digital image steganography and steganalysis: a journey of the past three decades," *Open Computer Science*, vol. 10, no. 1, pp. 296–342, 2020.

[2] C. Yu, X. Zhang, X. Zhang, G. Li, and Z. Tang, "Reversible data hiding with hierarchical embedding for encrypted images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 2, pp. 451–466, 2022.

[3] C. Yu, X. Zhang, G. Li, S. Zhan, and Z. Tang, "Reversible data hiding with adaptive difference recovery for encrypted images," *Information Sciences*, vol. 584, no. 15, pp. 89–110, 2022.

[4] Z. Tang, M. Pang, C. Yu, G. Fan, and X. Zhang, "Reversible data hiding for encrypted image based on adaptive prediction error coding," *IET Image Processing*, vol. 15, no. 11, pp. 2643–2655, 2021.

[5] A. Malik, S. Singh, and R. Kumar, "Recovery based high capacity reversible data hiding scheme using even-odd embedding," *Multimed. Tools Appl.*vol. 77, no. 12, pp. 15803–15827, 2018.

[6] R. Kumar, S. Chand, and S. Singh, "An Improved Histogram-Shifting-Imitated reversible data hiding based on HVS characteristics," *Multimed. Tools Appl.*vol. 77, no. 11, pp. 13445–13457, 2018.

[7] R. Sonar and G. Swain, "A hybrid steganography technique based on RR, AQVD, and QVC," *Information Security Journal: A Global Perspective*, vol. 31, no. 4, pp. 479–498, 2022.

[8] R. Sonar and G. Swain, "Steganography based on quotient value differencing and pixel value correlation," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 4, pp. 504–519, 2021.

[9] G. Swain, "Two new steganography techniques based on quotient value differencing with addition-subtraction logic and PVD with modulus function," *Optik*, vol. 180, pp. 807–823, 2019.

[10] S. Singh, "Adaptive PVD and LSB based high capacity data hiding scheme," *Multimed. Tools Appl.*vol. 79, no. 25–26, pp. 18815–18837, 2020.

[11] X. Zhang and S. Wang, "Efficient steganographic embedding by exploiting modification direction," *IEEE Commun. Lett.*vol. 10, no. 11, pp. 781–783, 2006.

[12] R. Atta, M. Ghanbari, and I. Elnahry, "Advanced image steganography based on exploiting modification direction and neutrosophic set," *Multimed. Tools Appl.*vol. 80, no. 14, pp. 21751–21769, 2021.

[13] C. F. Lee, Y. R. Wang, and C. C. Chang, "A steganographic method with high embedding capacity by improving exploiting modification direction,"vol. 1, pp. 497–500, in *Proceedings of the Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007)*, vol. 1, IEEE, Kaohsiung, Taiwan, November. 2007.

[14] K. Jung, "Improved exploiting modification direction method by modulus operation," *Image Processing*, vol. 2, no. 1, pp. 79–88, 2009.

[15] W. C. Kuo and C. C. Wang, "Data hiding based on generalised exploiting modification direction method," *The Imaging Science Journal*, vol. 61, no. 6, pp. 484–490, 2013.

[16] W. C. Kuo, S. H. Kuo, C. C. Wang, and L. C. Wuu, "High capacity data hiding scheme based on multi-bit encoding function," *Optik*, vol. 127, no. 4, pp. 1762–1769, 2016.

[17] T. D. Sairam and K. Boopathybagan, "An improved high capacity data hiding scheme using pixel value adjustment and modulus operation," *Multimed. Tools Appl.*vol. 79, no. 23–24, Article ID 17003, 2020.

[18] Y. Zhang, S. Wang, T. Li, B. Liu, and D. B. Pan, "Modulus calculations on prime number algorithm for information hiding with high comprehensive performance," *IEEE Access*, vol. 8, Article ID 85309, 2020.

[19] USC-SIPI Image database, "USC-SIPI Image database," 1977, https://sipi.usc.edu/database/.

[20] D. B. Khadse and G. Swain, "Data hiding using quotient value differencing and remainder value substitution avoiding incorrect extraction problem," *Sens. Imaging*, vol. 22, no. 1, p. 39, 2021.