Modulus Calculations on Prime Number Algorithm for Information Hiding With High Comprehensive Performance

YU ZHANG, SHA WANG, TENG LI, BO LIU, AND DONG-BO PAN

1 College of Computer and Information Science, Southwest University, Chongqing 400715, China
2 College of Artificial Intelligence, Southwest University, Chongqing 400715, China
3 Chongqing Xin’an Classified Network Security Testing and Evaluation Co., Ltd., Chongqing 400715, China

Corresponding author: Dong-Bo Pan (pandb@swu.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB2004100, in part by the Special Foundation for Basic Science and Frontier Technology Research Program of Chongqing under Grant cstc2017jcyjAX0295, in part by the Capacity Development Grant of Southwest University under Grant SWU116007, and in part by the National Natural Science Foundation of China under Grant 61732019, Grant 61672435, and Grant 61811530327.

ABSTRACT Information hiding is an important technique for information security, which is wildly studied by researchers. Recently embedding methods are proposed in spatial, frequency and other domains. After investigating previous literatures, we find that there is still room for embedding performance improvement. Inspired by some literatures, we propose a new method (Modulus Calculations on Prime Number Algorithm, MOPNA) for embedding secret data into cover-images. The main idea of MOPNA is hiding confidential data in paired cover-pixels with modulus calculation based on weight parameters consisting of prime numbers. MOPNA improves the embedding capacity while maintaining good stego-image quality. The correctness of MOPNA method is proved by a combination of mathematical and programming proof. The experimental results prove that the proposed method has high embedding capacity and achieves better comprehensive performance than existing methods.

INDEX TERMS Comprehensive performance, data hiding, information hiding, MOPNA, steganography.

I. INTRODUCTION

Information hiding, also known as steganography or data hiding, is an important information security technology [1]–[3]. It is an effective technique to hide secret information into the medium. It is similar to encrypted communication in that they are used to transmit messages from one party to another in a confidential manner. Encrypted communication means that the message is encrypted and transmitted in the communication channel. Even if the adversaries obtained the cipher text, it cannot be restored to the original information. Information hiding is to embed secret data to the image, video or other public carriers, and makes them less susceptible and difficult for adversaries to detect. This paper studies and proposes a novel method of information hiding, which has practical application value for copyright and intellectual property protection [4], labeling digital images [5], company information exchange, public information protection, online elections, internet banking, medical-imaging [3] etc.

The paper focuses on embedding information into mediums like images which are largely available in the network. With proposed hiding method, confidential data can be concealed in cover-images. The stego-images outputted can be directly transferred across an unsecured public network [6]. Now for digital images, information hiding is performed mainly in the spatial domains and frequency/transform domains [7]. In the frequency domain, the cover-image is transformed into frequency coefficients through various discrete transform functions, such as discrete Fourier transform [8], discrete cosine transform [9], and discrete wavelet transform [10], etc. The frequency domain methods have characteristic of anti-attack ability, but their disadvantage is low embedding capacity. Therefore this paper designs a high-capacity embedding method in the spatial domain.

Least significant bit (LSB) method is a common and important method in the spatial domain, in which the least
significant bits inside an image is replaced with confidential bits [11], [12]. LSB is simple to implement, but stego-images generated may draw suspicion or be easily detected [13]. For better embedding quality, in 2006 Zhang and Wang [14] provided a method fully exploits the modification directions (EMD06). The EMD06 method uses a pixel group (n cover pixels) to carry a secret digit in a \((2n+1)\)-ary notational system. It is based on the theory that the \((2n+1)\) different ways of modification to the cover pixels map to \((2n+1)\) possible values of a secret digit. EMD06 can give better embedding quality, but has low embedding capacity.

In 2007 Lee et al. [15] proposed a data hiding method (LWC07) based on EMD06 with the aim of enhancing the embedding capacity. LWC07 method converts confidential bits into a sequence of secret digits in an 8-ary notational system. Each secret digit is embedded into a pixel group including 2 cover-pixels. Jung [16] also tried to improve embedding capability and proposed an improved method (JY09) of the EMD06 method. The method uses one cover pixel to carry a secret digit in \((2n+1)\)-ary notational system, and achieves a capacity double that of EMD06 method. Kuo and Wang [17] proposed a method (GEMD13) eliminating the need to transform the secret message into a special number like in EMD06 and LWC07. Higher embedding capacity is achieved with GEMD13 method. Later Liu et al. [18] proposed an enhanced GEMD13 method by dividing a group of \(n\) cover-pixels into multiple groups, which improved the embedding capacity further. A method for information hiding using difference expansion and modulus function is presented in [6], which obtains good embedding quality with embedding capacity about 1.4 bpp.

However, these methods cannot embed more than two secret bits in one pixel on average. A hiding method based on adaptive difference expansion is presented in [19] which has embedding capacity of 3 bpp. In 2016, Kuo et al. [20] proposed a method (KKWW16) based on a multi-bit encoding function. They use a geometric series with common ratio \(r = 2\) to generate the extraction function coefficient. The modulus of \(2^{nk+1}\) is adopted to increase embedding capacity to \((nk+1)\) bits for \(n\) pixels, where the variable \(k\) determines how many bits can be embedded per pixel. Similar work is also done by Sairam and Boopathiyagan [21] who proposed an improved high capacity data hiding method (SB19). SB19 uses \(k\)-ary notational system to represent the confidential data. A desired value of \(x\) is chosen in \([-\lceil k^2/2 \rceil, \lceil k^2/2 \rceil]\), and the stego-pixel is obtained by adding \(x\) to original pixel value.

After our investigation and analysis of existing literatures, we find that there is still room to improve the embedding capacity and quality of information hiding. Inspired by some of these literatures, in this paper we propose MOPNA (Modulo Operation of Prime Numbers Algorithm) for embedding confidential data in images with high capacity and quality. Our innovation and contribution are threefold.

1) A novel information hiding method (MOPNA) based on modulo operation of prime numbers. The embedding capacity is improved to \((2k+1)/2\) bpp (bits/pixel). The new performance index is defined for evaluating hiding methods, and MOPNA shows best comprehensive performance than others;

2) The correctness of proposed method is proved by a combination of mathematical and programming proof. The effectiveness and the superiority of MOPNA are verified by experiments;

3) All methods including MOPNA are implemented in Python, and we make it available to all researchers at https://github.com/ppppp-cn/MOPNA

This paper is organized as follows. Methods of related works from EMD06, LWC07, JY09, GEMD13, KKWW16 and SB19 are described in Section II. The proposed novel method (MOPNA) is presented in Section III. Section IV provides experiment details and experimental results of MOPNA and the comparison of embedding capacity and quality between our method and methods in Section II. The conclusion and major contributions are highlighted in Section V.

II. RELATED WORKS

A. EMD06 METHOD

Zhang and Wang [14] introduced the Exploiting Modification Direction (EMD06) method that uses \(n\) cover-pixels to carry one secret digit which is in \((2n+1)\)-ary notational systems. The EMD06 embedding algorithm is shown as follows.

**Input:** cover-image pixels \(G = (g_1, g_2, \ldots, g_n)\) and secret data \(d, d \in [0, 2n]\)

**Output:** stego-image pixels \(G' = (g'_1, g'_2, \ldots, g'_n)\)

**Step 1:** Compute the extraction function given in (1).

\[
f(G) = \left[\sum_{i=1}^{n} g_i \cdot i \right] \mod (2n + 1) \tag{1}
\]

**Step 2:** The secret digit \(d\) is embedded into the cover-image pixels \(G\) as follows.

\[
G' = G
\]

\[
j = (d - f(G)) \mod (2n + 1)
\]

\[
\text{if } j > 0 \text{ and } j \leq n \text{ then } g'_j = g_j + 1
\]

\[
\text{else } g'_j = g_{2n+1-j} - 1
\]

The secret data \(d\) is extracted with the extraction function in (1) from stego-pixels \(G'\). The embedding capacity achieved by this method is \(\text{bpp} = k/n = \lceil \log_2 (2n+1) \rceil / n\), and the maximal value is 1 bpp when \(n = 1\), or \(n = 2\). The embedding capacity of EMD06 method has room to be improved [22].

B. LWC07 METHOD

Lee et al. [15] proposed a method (LWC07) based on EMD06 embedding method to enhance the embedding capacity. They fixed two pixels for each pixel group and gave the modified extraction function as (2) [23].
Input: A cover-pixel pair \((x_1, x_2)\) and the secret digit data \(d\) which is transformed from 3 confidential bits.

Output: A stego-pixel pair \((x'_1, x'_2)\)

Step 1: Calculate \(f_e(x_1, x_2)\) using (2).

\[
f_e(x_1, x_2) = (1 \times x_1 + 3 \times x_2) \mod 8 \tag{2}\]

Step 2: Adjust \((x_1, x_2)\) to obtain \((x'_1, x'_2)\) using following rules:

1. If \(d = f_e(x_1, x_2)\) then \(x'_1 = x_1 \) and \(x'_2 = x_2\).
2. Else if \(d = f_e(x_1 + 1, x_2)\) then \(x'_1 = x_1 + 1\) and \(x'_2 = x_2\).
3. Else if \(d = f_e(x_1 - 1, x_2)\) then \(x'_1 = x_1 - 1\) and \(x'_2 = x_2\).
4. Else if \(d = f_e(x_1, x_2 + 1)\) then \(x'_1 = x_1 \) and \(x'_2 = x_2 + 1\).
5. Else if \(d = f_e(x_1, x_2 - 1)\) then \(x'_1 = x_1 \) and \(x'_2 = x_2 - 1\).
6. Else if \(d = f_e(x_1 + 1, x_2 + 1)\) then \(x'_1 = x_1 + 1\) and \(x'_2 = x_2 + 1\).
7. Else if \(d = f_e(x_1 + 1, x_2 - 1)\) then \(x'_1 = x_1 + 1\) and \(x'_2 = x_2 - 1\).
8. Else if \(d = f_e(x_1 - 1, x_2 + 1)\) then \(x'_1 = x_1 - 1\) and \(x'_2 = x_2 + 1\).

C. JY09 METHOD

Jung [16] designed an improved method on EMD06 to hide more secret bits.

Input: A cover-pixel and the secret digit data \(d\) which is transformed from \(k\) confidential bits

Output: A stego-pixel pair \(g\)

Step 1: For a pixel \(g\) in cover-image, compute the extraction function \(f\), given in (3).

\[
f = g \mod (2k + 1) \tag{3}\]

Step 2: Choose the \(x\) value in a specific Range for satisfying following equation.

\[
d = (g + x) \mod (2k + 1)\]

1. If \(0 \leq g \leq 1\) then Range = \([0, (2k + 1)]\)
2. Else if \(254 \leq g \leq 255\) then Range = \([- (2k + 1), 0]\)
3. Else Range = \([- (2k + 1), (2k + 1)]\)

Step 3: Output \(g'\) as \(g' = g + x\)

With the extraction function in (3), secret data embedded in the pixels of stego-image can be extracted. The method can embed \(k\) secret bits on every pixel of cover data. The embedding capacity is \(k\) bpp, and \(k\) can only take 1 or 2.

D. GEMD13 METHOD

Kuo and Wang [17] presented a data-hiding method which does not require the sender to transform the secret message into a special number form. The major property of the GEMD13 method is maintaining embedding capacity of more than 1 bpp [23].

Input: \(n\) adjacent pixels \(G = (g_1, g_2, \ldots, g_n)\) and \(n + 1\) binary bits secret data \((s_n, s_{n-1}, \ldots, s_0)\), \(d = (2^n \times s_n + 2^{n-1} \times s_{n-1} + \ldots + 2^1 \times s_1 + s_0)\)

Output: A stego-pixel \(G' = (g'_1, g'_2, \ldots, g'_n)\)

Step 1: Compute variable \(\text{diff}\) using (4).

\[
f_{\text{GEMD}}(G) = \left[ \sum_{i=1}^{n} (2^i - 1) \times g_i \right] \mod 2^{n+1} \tag{4}\]

\[
\text{diff} = [d - f_{\text{GEMD}}(G)] \mod 2^{n+1}, \text{diff} \in [0, 2^{n+1} - 1] \tag{5}\]

Step 2: Generate \(G'\) with following rules.

\[
G' = \begin{cases} 
G & \text{if diff} \leq 2^n \text{ then} \\
G' & \text{else if diff} > 2^n \text{ then} \\
\end{cases}
\]

For \(i = n \text{ down to } 1 \text{ do} \)

\[
\begin{aligned}
g'_i &= g_i + 1 \\
g'_i &= g_i - 1 \\
\end{aligned}
\]

The secret data \(d\) can be extracted through calculating \(f_{\text{GEMD}}(G')\) in (4). The embedding capacity is \((n+1)/n\) bpp. Later, Wang et al. [23] modified GEMD13 and got embedding capacity of 2 bpp when \(n\) increases.

E. KKWW16 METHOD

Kuo et al. [20] proposed a high hiding capacity and acceptable stego-image quality steganography method based on multi-bit encoding function.

Input: \(n\) adjacent pixels \(G\) and \(nk + 1\) binary bits secret data \((s_{nk}, s_{nk-1}, \ldots, s_0)\)

Output: A stego-pixels \(G'\)

Step 1: Compute the extraction function using (5)

\[
f(G) = \left[ \sum_{i=1}^{n} c_i \times g_i \right] \mod 2^{nk+1} \tag{5}\]
where \( c_i = \begin{cases} 1, & i = 1 \\ 2^i c_{i-1} + 1, & i \neq 1 \text{ and } i > 0 \end{cases} \)

Step 2: Transform confidential bits into secret value \( S \). The transformation of each secret value is done by sum of product.

\[
S = (2^n \times s_n + 2^{n-1} \times s_{n-1} + \ldots + 2^1 \times s_1 + s_0)
\]

Step 3: Compute the difference \( D \) between \( f(G) \) and secret value \( S \) using the equation given below

\[
D = (S - f(G)) \mod 2^{nk+1}
\]

Step 4: Embed the secret data into cover-pixels as follows. \( G' = G \)

If \( D = 0 \)

\[
g'_n = g_n
\]

If \( D = 2^{nk} \)

\[
g'_n = g_n + (2^k - 1)
\]

\[
g'_1 = g_1 + 1
\]

If \( D < 2^{nk} \)

transform \( D \rightarrow (d_n-1 \ldots d_1 d_0)_{2^k} \)

for each \( i \)

\[
g'_{i+1} = g_{i+1} + d_i
\]

\[
g'_i = g_i - d_i, \text{ if } i > 0
\]

If \( D > 2^{nk} \)

transform \( (2^{nk+1} - D) \rightarrow (d_n-1 \ldots d_1 d_0)_{2^k} \)

for each \( i \)

\[
g'_{i+1} = g_{i+1} - d_i
\]

\[
g'_i = g_i + d_i, \text{ if } i > 0
\]

The secret data \( S = f(G') \) can be recovered from each block by using (5). The embedding capacity is \((nk+1) / n \text{bpp}\).

**F. SB19 METHOD**

Recently, Sairam and Boopathybagan [21] proposed a data hiding method, which uses \( k^2 \)-ary notational system and aims to achieve higher embedding capability.

**Input:** A cover-pixel \( g \) and the secret digit data \( d \) which is transformed from \( k \) confidential bits

**Output:** A stego-pixel \( g' \)

Step 1: Convert the secret bits into a digit in \( k^2 \)-ary notational system before applying the algorithm, and represent the secret digit as \( d \).

Step 2: For cover-pixel value \( g \) and secret value \( d \), search the value for variable \( x \) which makes \( d = (g + x) \mod k^2 \) hold. Then use variable \( x \) to generate stego-pixel \( g' \) as follows.

\[
f(x) = \begin{cases} (g + x) \mod k^2 & \text{iff } d = \lfloor x \rfloor \\ g' = g + x & \text{break} \end{cases}
\]

where \( \lfloor \cdot \rfloor \) is the flooring operation.

All the embedded values, from the stego-image with adjusted pixel value \( g' \), are extracted by applying the extraction function in (6).

\[
f'(G') = g' \mod k^2
\]

The extracted values \( f' \) is the secret information \( d \) embedded in stego-pixel.

**III. OUR PROPOSED METHOD: MOPNA**

**A. MOPNA METHOD**

Inspired by some literatures [15], [16], [20], [21], we design a new method for embedding secret data using modulus calculations with prime numbers. MOPNA divides cover-image into pixel groups that includes two pixels (a pixel pair). Then a decimal value calculated from \( 2k + 1 \) bits of confidential data is embedded into a pixel pair.

Let \( G = (g_0, g_1) \) be a pixel pair of the cover-image. The pixel value is \( g_i, i = [0, 1] \), and typically \( g_i \in [0, 255] \) for an 8-bit grayscale image. Let \( G' \) be the pixel pair of stego-image, \( G' = (g'_0, g'_1) \). \( C = (c_0, c_1) \) is the weight parameter for modulus calculation, and \( c_i, i = [0, 1] \), is a prime number, for example \( C = (3, 11) \). A modulus calculation function \( f(G) \) is defined in (7) and a function \( CPV \) changes in pixel values in (8) is proposed to measure the effect of embedding. The \( f(G) \) and \( CPV \) functions are used to search the embedding variable \( X \), with which \( CPV(G + X, G) \) reaches maximal value. Finally after adding the variant \( X \) and the original pixel pair \( G \) together, the stego-pixel pair \( G' \) is obtained.

**Input:** A pixel pair \( G = (g_0, g_1) \) and the 2k + 1 bits of confidential data

**Output:** A stego-pixel pair \( G' = (g'_0, g'_1) \)

Step 1: Compute the function \( f(G) \) with (7).

\[
f(G) = C \cdot G^T \mod 2^{2k+1}
\]

Step 2: Transform the confidential binary bits \((b_{2k} b_{2k-1} \ldots b_1 b_0)_{2} \), \( b_i \in (0, 1) \), into decimal valued = \( 2^{2k} b_{2k} + 2^{2k-1} b_{2k-1} + \ldots + 2^1 b_1 + 2^0 b_0 \).

Step 3: Search variable \( X \)

\[
X = \{(x_0, x_1) | d = f(G + X) \quad \text{and} \quad \forall X' \neq X : CPV(G + X, G) \geq CPV(G + X', G) \}
\]

\[x_0, x_1 \in [-255, 255]\]

Step 4: Generate stego-pixel pair.

\[
G' = G + X
\]

The function of \( CPV(G, G') \) in Step 3 is defined in (8).

\[
CPV(G, G') = \begin{cases} 10 \log_{10} \frac{||G|| - ||G' - G||}{255}, & ||G|| \neq 0, ||G - G'|| \neq 0 \\
10 \log_{10} \frac{||G||}{255}, & ||G|| = 0, ||G - G'|| \neq 0 \\
100, & ||G - G'|| = 0
\end{cases}
\]
where $\| G \| = g_0^2 + g_1^2$ and $\| G - G' \| = (g_0 - g_0')^2 + (g_1 - g_1')^2$. The CPV function is used to calculate magnitude of changes in pixel values. The larger the CPV value is, the smaller the change in pixel value, or the closer the two pixels are.

**B. PROOF OF CORRECTNESS**

We use mathematical and programming methods to prove the correctness of the proposed method. Now the following proof applies only to 8-bit grayscale images. Two specific prime numbers $(3, 11)$ are selected for weight parameter $C$. Following theorems are all proved under the condition that $C = (3, 11)$.

**Theorem 1:** When $k \in [1, 5]$, the value of $f(G)$ in (7) may take any one of the values in $[0, 2^{2k+1} - 1]$.

**Proof:** A Python program is used to prove that $f(G)$ can take any one of the values in $[0, 2^{2k+1} - 1]$. If the program is executed correctly, the theorem is proved. Conversely, if the program stops on the assert statement, the theorem does not hold.

| for $g_0$ in range $(0, 256)$: |
| for $g_1$ in range $(0, 256)$: |
| $d = (c_0 \ast g_0 + c_1 \ast g_1) \% (2 \ast \ast (2k+1))$ |
| if $d$ not in outlist: |
| outlist.append $(d)$ |
| assert (len(outlist) == $m$) |

The program runs successfully without stopping, and it proves the theorem holds. □

With similar proof process, it can be proved that Theorem 1 also holds when $C = (3, 7), (3, 13), (5, 9), (5, 11), (5, 13), (7, 9), (7, 11), (9, 17), (9, 19), (11, 17), (11, 19) \ldots$ We have not tested all possible values of $C$, nor do we intend to do so. Because in the MOPNA design process, we find that with $C = (3, 11)$ MOPNA achieves good enough results. Therefore, this article is only carrying out MOPNA correctness verification and experiments for the case of $C = (3, 11)$.

Note that when $k \geq 6$ or $C$ takes other values, for example $C = (4, 12)$, Theorem 1 may not hold.

**Theorem 2:** When $k \in [1, 5]$, it can be guaranteed that Step 3 can find a variable $X$ such that $d = f(G + X)$ holds.

**Proof:** $\because x_0, x_1 \in [-255, 255], g_0, g_1 \in [0, 255]$. Let $G' = G + X, G' = (g_0', g_1')$.

$\because g_0', g_1' \in [-255, 255]$.

According to Theorem 1, we can find a pair $(x_0, x_1)$ such that $g_0', g_1' \in [0, 255]$ and $d = f(G')$ hold. □

Theorem 2 shows that $X$ can always be found and used for embedding secret data.

**Theorem 3:** When $k \in [1, 3]$, there is more than one value for $X$ that make $d = f(G + X)$ hold.

**Proof:** Let $X = (X_0, x_1)$ that make $d = f(G + X)$ hold. $\because k \in [1, 3], m = 2^{2k+1} \leq 128$

If $x_0 < 127$ then construct $X' = (x_0 + m, x_1)$, else construct $X' = (x_0 - m, x_1)$.

$f(G + X') = f(G + (x_0 \pm m, x_1)) = f(G + X) \pm c_0 m = f(G + X) = d \% (mod m)$.

$\therefore$ Both $X$ and $X'$ may make $d = f(G + X)$ hold. □

Theorem 3 means that multiple different values can be found for $G'$ to hide information. We can choose the one yielding maximum CPV value. Therefore, according to the above theorems, for 8-bit grayscale images, under the condition that $C = (3, 11)$ and $k \in [1, 3]$, MOPNA method proposed in the paper can successfully embed and recover the secret data.

**IV. EXPERIMENTAL RESULTS AND DISCUSSION**

In this section, the experiments are carried out to evaluate the performance of MOPNA method. Our proposed method and several other methods are implemented in Python and run in a PC with an Intel i7-7700 CPU @ 3.6 GHz and 8-GB RAM. The operating system is Windows 10 Professional 64-bit. The performance evaluation of information hiding methods includes image quality, embedding capacity/embedding payload and peak signal to noise ratio.

**A. THE QUALITY OF STEGO-IMAGE**

MOPNA method together with other previous methods are tested on some $256 \times 256$ grayscale images (Baboon, Barbara, Boat, F-16, Goldhill, Lena, Pepper and Tiffany [18]). In our experiments, the numbers of confidential bits (NCB) are set as $NCB = 49k$ and $NCB = 98k$ respectively. The stego-images of Baboon and Boat generated by different methods are shown in Fig. 1 to Fig. 4. More stego-images can be found at https://github.com/ppppp-cn/MOPNA.

It is difficult to see the difference between the original cover-image and the stego-image with naked eyes in Figures 1-4, and that is why the information hiding works. However, there is a difference between the cover-image and the stego-images, and between the stego-images generated by different information hiding methods. In order to observe this difference with naked eyes, we convert stego-images into videos. By recording the switching of pictures as a video, we can observe the minor changes between stego-images with naked eyes. Since the video cannot be shown in this paper, we uploaded the video file to GitHub. One may notice the small pixel changes during the switching of stego-images in a video file available at https://github.com/ppppp-cn/MOPNA/blob/master/diff.mp4.

From Fig. 1 to Fig. 4, we can see MOPNA gives better or similar embedding quality when comparing with previous embedding methods. It is not easy to notice the hidden information existed in these stego-images, and this demonstrates that our method has high embedding capacity while still produces stego-images with acceptable quality.

**B. THE EMBEDDING CAPACITY**

The embedding capacity is defined as the number of secret bits that can be hidden in every cover pixel (bits/pixel, bpp)
A larger value of bpp represents that the method has better embedding performance, that is, a pixel in the cover-image can carry more confidential bits. On the contrary, a smaller value of bpp means a worse embedding performance [24].

The embedding methods’ characteristics including number of pixels in a pixel group, number of bits embedded in each group and embedding capacity (bpp) are summarized in Table 1. In the table, \( n \) is a variable for number of pixels in one group and \( k \) is used to adjust the degree of embedding capacity. For MOPNA, \( n = 2 \) and \( k \in [1, 3] \) in experiments.

The pixels of cover-image are divided into groups, in which the confidential data will be embedded. The number of pixels is different for each embedding method. There are \( n, 2, 1, n, n, 1 \) and 2 pixels in one pixel group of EMD06, LWC07, JY09, GEMD13, KKWW16, SB19 and MOPNA respectively. Each group in EMD06, LWC07, JY09, GEMD13, KKWW16, SB19 and MOPNA methods can carry \( \log_2(2n + 1) \), 3, 2, \( n+1 \), \( nk+1 \), \( k \) and \( 2k+1 \) confidential bits respectively. So the embedding capacity of each method is obtained through dividing number of pixels in one group by bits embedded in the group. The final bpps for EMD06, LWC07, JY09, GEMD13, KKWW16, SB19 and MOPNA method are \( \frac{\log_2(2n+1)}{n}, 1.5, 2, \frac{n+1}{n}, \frac{nk+1}{n}, k \) and \( \frac{2k+1}{2} \) respectively.

It is not easy to see which bpp in the table is better. Therefore, we calculate their specific values and express them with a graph. As shown in Fig. 5, it is obvious that MOPNA method and KKWW16 method have best embedding capacity, which is 3.5 bpp when \( k = 3 \) (and \( n = 2 \) for KKWW16).

### Table 1. Embedding capacity of methods.

|                | EMD06 | LWC07 | JY09 | GEMD13 | KKWW16 | SB19 | MOPNA |
|----------------|-------|-------|------|--------|--------|------|-------|
| No. of pixels  | \( n \) | 2     | 1    | \( n \) | \( n \) | 1    | 2     |
| Embedding bits | \( \log_2(2n+1) \) | 3     | 2    | \( n+1 \) | \( nk+1 \) | \( k \) | \( 2k+1 \) |
| bpp            | \( \frac{\log_2(2n+1)}{n} \) | 1.5   | 2    | \( \frac{n+1}{n} \) | \( \frac{nk+1}{n} \) | \( k \) | \( \frac{2k+1}{2} \) |
C. PEAK SIGNAL TO NOISE RATIO

We investigate quality of stego-images with Peak Signal to Noise Ratio (PSNR) [25] defined in (9). The formula of MSE (Mean Square Error) [5] calculation is defined in (10).

\[
PSNR = 10 \log_{10} \frac{255^2}{MSE} \text{ dB} \quad (9)
\]

\[
MSE = \frac{\sum_{x=1}^{M} \sum_{y=1}^{N} [I(x,y) - I'(x,y)]^2}{MN} \quad (10)
\]

where \( M \) and \( N \) represent the length and width of the image, and \( I(x,y), I'(x,y) \) respectively stand for the original pixel value and the stego-pixel value at position \((x,y)\).

In our experiments, the numbers of confidential bits (NCB) are set as \( NCB = 49k \) and \( NCB = 98k \) respectively. Among these embedding methods, EMD06 has lowest embedding capacity. When the number of cover-pixels in one group is set as \( n = 2 \), the embedding capacity of EMD06 is 1.16 bpp. There are \( 256 \times 256 = 65536 \) pixels available for carrying confidential data. The maximal number of bits of confidential data which can be embedded is \( \lfloor 65536 \div 2 \rfloor \times \lfloor 1.16 \times 2 \rfloor = 65536 \). When \( n=4 \), the embedding capacity of EMD06 is 0.79 bpp. The maximal number of bits can be embedded is \( \lfloor 65536 \div 4 \rfloor \times \lfloor 0.79 \times 4 \rfloor = 49152 \). That is why \( NCB=49k \) confidential bits is selected in experiments.

When the numbers of confidential bits is \( NCB = 49k \), the result of simulation is shown in Table 2. The data in columns from 2 to 9 is the PSNR for 8 images. The average value of PSNR and the standard deviation (stdev) value are given in the following two columns. The value in NoP column is the number of pixels in cover-image used to embed the confidential data. It is obvious that EMD06 gives highest PSNR with \( n=4 \), but it has lowest embedding capacity among the methods; it uses 65336 cover pixels to carry the confidential data. For methods of KKWW16, SB19 and MOPNA, a lower value of \( k \) may introduce lower embedding capacity, but higher PSNR. On the contrary, when using a higher value of \( k \), they give higher embedding capacity but lower PSNR. When \( k=3 \), both MOPNA and KKWW16 (at \( n=2 \)) reach the highest embedding capacity. Both of them use 14000 cover-pixels to carry the 49k confidential bit, but MOPNA has better PSNR.
The method SB19 can be used as a high embedding capacity method. When $k$ is set to 2 and 3, average PSNR values are 46.36 and 39.89 respectively, which is a little higher than PSNR values of our proposed method. That is because SB19 has lower embedding capacity than MOPNA, when the number of bits to be embedded is set as $k = 2$ or $k = 3$. Other methods like LWC07, JY09 and GEMD13 show better performance in either embedding quality or capacity, but not at the same time. When combined embedding quality and capacity together, MOPNA does better.

We define a comprehensive index to represent the performance combining both the embedding capacity and embedding quality. The new index is $RPN$ (ratio of PSNR and number of cover pixels used to carry the confidential data) as in (11). The higher value of $RPN$ means the higher comprehensive embedding performance.

$$RPN = \frac{PSNR \times 1000}{\text{number of cover pixels used}} \quad (11)$$

The $RPN$ values from different methods are calculated and shown in Fig. 6. From the experimental result in the figure,
it is easy to notice the embedding capacity and quality values from KKWW16, SB19 and MOPNA are better than others. However, our proposed method MOPNA performs best (at \( k = 3 \)).

When the numbers of confidential bits is \( NCB = 98k \), the result of simulation is shown in Table 3. At this case, when the number of cover pixels in one group of EMD06 is set as \( n = 2 \), EMD06 reaches its maximal embedding capacity i.e. 1.16 bpp. Total 65536 confidential bits can be carried by a 256 × 256 grayscale image. So 98k bits of confidential data cannot be embedded in the images with EMD06. This also applies to LWC07, JY09 (with \( k = 1 \)) and GEMD13 methods. Only the methods with embedding capacity of at least 2 bpp can be used to carry the 98k confidential bits.

From the experimental data in Table 3, we can see KKWW16, SB19 and MOPNA have higher embedding capacity when the number of confidential bits to be embedded is set as \( k = 3 \), but have lower embedding qualities. When \( k \)
is set to 2, these methods show higher embedding quality values, while the embedding capacity values are lower than in the case of \( k = 3 \). Among the methods SB19 (\( k = 2 \)) gives the highest average PSNR at 46.37, with the cost of lower embedding capacity. It needs more cover pixels (49000) to carry the load than methods like KKWW16(39200 cover pixels) to have the highest PSNR.
pixels at \( n = 2 \) and \( k = 2 \), 28000 cover pixels at \( n = 2 \) and \( k = 3 \), 43556 cover pixels at \( n = 4 \) and \( k = 2 \), and 30156 cover pixels at \( n = 4 \) and \( k = 3 \), MOPNA(39200 cover pixels at \( k = 2 \), 28000 cover pixels at \( k = 3 \)).

It is also easy to see these methods show certain advantages in some aspects, but MOPNA method shows higher comprehensive performance. The result with comprehensive performance index (RPN) as defined in (11) is shown in Fig. 7. The RPN value of MOPNA (at \( k = 3 \)) is 1.35 which is the highest among the methods discussed. That means our proposed method has best comprehensive performance at this case.

![FIGURE 6. Comprehensive performance under NCB= 49k.](image)

![FIGURE 7. Comprehensive performance under NCB= 98k.](image)

V. CONCLUSION

Information hiding is an important technique for information security. It is wildly studied by researchers and many embedding methods are introduced. After carefully investigating of the existing literatures, we find that there is still room for embedding performance improvement in spatial domain of information hiding.

In this paper, methods like EMD06 [14], LWC07 [15], JY09 [16], GEMD13 [17], KKW16 [20] and SB19 [21] is studied in detail and implemented in Python. Inspired by some of these literatures, we propose a new method (Modulus Calculations on Prime Number Algorithm, MOPNA) for embedding secret data into cover-images. The main idea of MOPNA is hiding confidential data in paired cover-pixels with modulus calculation based on weight parameters consisting of prime numbers. It searches for a variable \( X \), which will be added to cover pixel pair \( G \) to form the stego-pixel pair \( G' \). The correctness of MOPNA method is proved by a combination of mathematical and programming proof.

Experiments are carried out to verify MOPNA method, and the result shows the effeciveness and the superiority of the proposed method. With our method, embedding capacity is improved to \((2k + 1)/2 \) bpp, \( k \in [1, 3] \), without decreasing the embedding quality. MOPNA shows best comprehensive performance among methods this paper involved, when embedding high payload into cover-images.

In future studies, different prime number pairs will be adopted to improve the performance of MOPNA. Since MOPNA has higher computational complexity than other methods, we will continue to study how to reduce its complexity. It should be noted that it is necessary to prove its correctness before using MOPNA with \( k \geq 4 \). We will also prove this in the subsequent work. In addition, the method will be evaluated and improved by using different structural analysis methods, such as histogram analysis and non-structural steganalysis methods.

REFERENCES

[1] F. A. Jassim, “A novel Steganography algorithm for hiding text in image using five modulus method,” Int. J. Comput. Appl., vol. 72, no. 17, pp. 975–8887, 2013.

[2] N. Wang, Y. Chen, Y. Yang, Z. Fang, and Y. Sun, “Blockchain private key storage algorithm based on image information hiding,” in Artificial Intelligence and Security (Lecture Notes in Computer Science), vol. 11635, 2019, pp. 542–552.

[3] C. P Sumathi, T. Santanam, and G. Umamaheswari, “A study of various steganographic techniques used for information hiding,” Int. J. Comput. Sci. Eng. Surv., vol. 4, no. 6, pp. 9–25, Dec. 2013.

[4] M. Hussain, A. W. A. Wahab, N. Javed, and K.-H. Jung, “Recursive information hiding scheme through LSB, PVD shift, and MPE,” IETE Tech. Rev., vol. 35, no. 1, pp. 53–63, Jan. 2018.

[5] Q. Ying, Z. Qian, X. Zhang, and D. Ye, “Reversible data hiding with image enhancement using histogram shifting,” IEEE Access, vol. 7, pp. 46506–46521, 2019.

[6] P. Manirho and T. Ahmad, “Information hiding scheme for digital images using difference expansion and modulus function,” J. King Saud Univ. Comput. Inf. Sci., vol. 31, no. 3, pp. 335–347, Jul. 2019.

[7] S. A. Parah, J. A. Sheikh, J. A. Akhoon, N. A. Loan, and G. M. Bhat, “Information hiding in edges: A high capacity information hiding technique using hybrid edge detection,” Multimedia Tools Appl., vol. 77, no. 1, pp. 185–207, Jan. 2018.

[8] D. Bhattacharyya and T. H. Kim, “Image data hiding technique using discrete Fourier transformation,” in Ubiquitous Computing and Multime dia Applications (Communications in Computer and Information Science), vol. 151. 2011, pp. 315–323.

[9] K. Wang, Z. M. Lu, and Y. J. Hu, “A high capacity lossless data hiding method for JPEG images,” J. Syst. Softw., vol. 86, no. 7, pp. 1965–1975, 2013.

[10] P. Patil and D. S. Bormane, “DWT based invisible watermarking technique for digital images,” Int. J. Eng. Adv. Technol., vol. 2, no. 4, pp. 603–605, 2013.

[11] C.-K. Chan and L. M. Cheng, “Hiding data in images by simple LSB substitution,” Pattern Recognit., vol. 37, no. 3, pp. 469–474, Mar. 2004.

[12] A. G. S. Gupta and B. Bluhshan, “Information hiding using least significant bit steganography and cryptography,” Int. J. Modern Edu. Comput. Sci., vol. 4, no. 6, pp. 27–34, Jun. 2012.
YU ZHANG was born in Banan, Chongqing, China, in 1979. He received the B.S. and M.S. degrees in computer science and technology from the Southwest Normal University of China, in 2002 and 2005, and the Ph.D. degree in communication engineering from Chongqing University, China, in 2016.

From 2005 to 2012, he was a Lecturer with the Department of Automation of China, Southwest University. Since 2012, he has been an Associate Professor with the College of Computer and Information Science, Chongqing. He is the author of five books, more than 30 articles, and more than ten inventions. His research interests include optimization in automation systems, machine learning, intelligent control, integration of control systems, and industrial communications. He drafted an international standard about device integration and more than ten national standards.

Dr. Zhang was a member of IEC and SC2/TC124/S/AC. He was a recipient of the Standard Innovation Contribution Award, in 2007, the Chongqing Science and Technology Progress Award (Second Class), in 2014, and the Chongqing Science and Technology Achievements Award, in 2015.

SHA WANG was born in Chongqing, China, in 1995. She received the B.S. degree in computer science and technology from the Shenyang University of Technology of China, in 2018. She is currently pursuing the master’s degree in computer science and technology with Southwest University, China.

Her research interests include machine learning and network security.

Ms. Wang is a member of CCF. She won the Third Prize in the Chinese Computer Competition, in 2017.

TENG LI was born in Xucheng, Anhui, China, in 1997. He received the B.E. degree in computer science and technology from the Southwest University, China, in 2019, where he is currently pursuing the master’s degree.

His research interests include machine learning and information hiding technology.

Mr. Li is a member of CCF. In 2017, he won the Third Prize in Mathematical Modeling Competition of Southwest University.

BO LIU was born in Guang’an, Sichuan, China, in 1981. He received the B.S. degree in software engineering from Wuhan University, Wuhan, China, in 2004, and the Ph.D. degree in computer science and technology from Chongqing University, Chongqing, China, in 2012.

Since 2013, he has been a Lecturer with the Department of Software Engineering, Southwest University, Chongqing. He is the author of more than fifteen articles and a PI or a member in more than five scientific projects in cyber-physical systems, model-driven development, software system trustworthiness, service computing, and intelligent algorithms.

Dr. Liu was a member of ACM/CCF/CCF YOCSEF (China Computer Federation Young Computer Scientists and Engineers Forum). He was a recipient of Excellent Academic Secretary Award of CCF YOCSEF, Chongqing, in 2017. He was also the Local Chair of SETSS 2016/2017/2018/2019 and the China Annual Conference on Formal Methods, in 2018.

DONG-BO PAN was born in Chongqing, China, in 1977. He received the B.S. degree in electrical automation from Shenyang Polytechnic University, in 1997, and the M.S. degree in control theory and control engineering from Chongqing University, China, in 2003. He is currently pursuing the Ph.D. degree in mathematics with Southwest University, China.

He is currently an Associate Professor with the College of Artificial Intelligence, Southwest University. His research interests include industrial control systems, network security, advanced manufacture, and system control.

* * *